NVIDIA Performance Primitives (NPP)

Version 4.0

February 11, 2011

Contents

1 NVIDIA Performance Primitives **1**

1.1 What is NPP? 1

1.2 Documentation 1

1.3 Technical Specifications 1

1.4 Files 2

1.4.1 Header Files 2

1.4.2 Library Files 2

1.5 Supported NVIDIA Hardware 2

2 General API Conventions **3**

2.1 Memory Management 4

2.2 Function Naming 4

2.3 Integer Result Scaling 4

3 Signal-Processing Specific API Conventions **7**

3.1 Signal Data 8

3.1.1 Parameter Names for Signal Data 8

3.1.1.1 Source Signal Pointer 8

3.1.1.2 Destination Signal Pointer 8

3.1.1.3 In-Place Signal Pointer 8

3.1.2 Signal Data Alignment Requirements 9

3.1.3 Signal Data Related Error Codes 9

3.2 Signal Length 9

3.2.1 Length Related Error Codes 9

4 Imaging-Processing Specific API Conventions **11**

4.1 Function Naming 12

4.2 Image Data 12

4.2.1 Line Step 13

4.2.2 Parameter Names for Image Data 13
4.2.2.1 Passing Source-Image Data 13
4.2.2.2 Passing Destination-Image Data 14
4.2.2.3 Passing In-Place Image Data 14
4.2.3 Image Data Alignment Requirements 14
4.2.4 Image Data Related Error Codes 15
4.3 Region-of-Interest (ROI) 15
4.3.1 ROI Related Error Codes 15

5 Module Index **17**
5.1 Modules 17

6 Data Structure Index **19**
6.1 Data Structures 19

7 Module Documentation **21**
7.1 NPP Core 21
7.1.1 Detailed Description 21
7.1.2 Function Documentation 22
7.1.2.1 nppGetGpuComputeCapability 22
7.1.2.2 nppGetGpuName 22
7.1.2.3 nppGetGpuNumSMs 22
7.1.2.4 nppGetLibVersion 22
7.1.2.5 nppGetMaxThreadsPerBlock 22
7.1.2.6 nppGetStream 22
7.1.2.7 nppSetStream 23
7.2 NPP Type Definitions and Constants 24
7.2.1 Define Documentation 27
7.2.1.1 NPP_MAX_16S 27
7.2.1.2 NPP_MAX_16U 27
7.2.1.3 NPP_MAX_32S 27
7.2.1.4 NPP_MAX_32U 27
7.2.1.5 NPP_MAX_64S 27
7.2.1.6 NPP_MAX_8S 27
7.2.1.7 NPP_MAX_8U 28
7.2.1.8 NPP_MAXABS_32F 28
7.2.1.9 NPP_MAXABS_64F 28
7.2.1.10 NPP_MIN_16S 28

7.2.1.11 NPP_MIN_16U 28
7.2.1.12 NPP_MIN_32S 28
7.2.1.13 NPP_MIN_32U 28
7.2.1.14 NPP_MIN_64S 28
7.2.1.15 NPP_MIN_8S 28
7.2.1.16 NPP_MIN_8U 28
7.2.1.17 NPP_MINABS_32F 28
7.2.1.18 NPP_MINABS_64F 29
7.2.2 Enumeration Type Documentation 29
7.2.2.1 NppCmpOp 29
7.2.2.2 NppGpuComputeCapability 29
7.2.2.3 NppHintAlgorithm 29
7.2.2.4 NppiAxis 29
7.2.2.5 NppiInterpolationMode 30
7.2.2.6 NppRoundMode 30
7.2.2.7 NppStatus 30
7.3 Basic NPP Data Types 32
7.3.1 Typedef Documentation 33
7.3.1.1 Npp16s 33
7.3.1.2 Npp16u 33
7.3.1.3 Npp32f 33
7.3.1.4 Npp32s 33
7.3.1.5 Npp32u 33
7.3.1.6 Npp64f 33
7.3.1.7 Npp64s 33
7.3.1.8 Npp64u 33
7.3.1.9 Npp8s 33
7.3.1.10 Npp8u 33
7.4 NPP Image Processing 34
7.4.1 Function Documentation 73
7.4.1.1 nppiAbsDiff_32f_C1R 73
7.4.1.2 nppiAbsDiff_32s_C1R 73
7.4.1.3 nppiAbsDiff_8u_AC4R 74
7.4.1.4 nppiAbsDiff_8u_C1R 74
7.4.1.5 nppiAbsDiff_8u_C4R 75
7.4.1.6 nppiAbsDiffC_32f_C1R 75

7.4.1.7 nppiAdd_32f_C1R . . . 75
7.4.1.8 nppiAdd_32s_C1R . . . 76
7.4.1.9 nppiAdd_8u_AC4RSfs . . . 76
7.4.1.10 nppiAdd_8u_C1RSfs . . . 77
7.4.1.11 nppiAdd_8u_C4RSfs . . . 77
7.4.1.12 nppiAddC_32f_C1R . . . 77
7.4.1.13 nppiAddC_32fc_C1R . . . 78
7.4.1.14 nppiColorTwist32f_8u_AC4R . . . 78
7.4.1.15 nppiColorTwist32f_8u_C3R . . . 79
7.4.1.16 nppiColorTwist32f_8u_P3R . . . 79
7.4.1.17 nppiCompare_32f_C1R . . . 79
7.4.1.18 nppiCompare_8u_AC4R . . . 80
7.4.1.19 nppiCompare_8u_C4R . . . 80
7.4.1.20 nppiConvert_16s32f_C1R . . . 81
7.4.1.21 nppiConvert_16s32s_C1R . . . 81
7.4.1.22 nppiConvert_16s8u_AC4R . . . 82
7.4.1.23 nppiConvert_16s8u_C1R . . . 82
7.4.1.24 nppiConvert_16s8u_C4R . . . 82
7.4.1.25 nppiConvert_16u32f_C1R . . . 83
7.4.1.26 nppiConvert_16u32s_C1R . . . 83
7.4.1.27 nppiConvert_16u8u_AC4R . . . 83
7.4.1.28 nppiConvert_16u8u_C1R . . . 84
7.4.1.29 nppiConvert_16u8u_C4R . . . 84
7.4.1.30 nppiConvert_32f16s_C1R . . . 85
7.4.1.31 nppiConvert_32f16u_C1R . . . 85
7.4.1.32 nppiConvert_32f8u_C1R . . . 85
7.4.1.33 nppiConvert_8u16s_AC4R . . . 86
7.4.1.34 nppiConvert_8u16s_C1R . . . 86
7.4.1.35 nppiConvert_8u16s_C4R . . . 87
7.4.1.36 nppiConvert_8u16u_AC4R . . . 87
7.4.1.37 nppiConvert_8u16u_C1R . . . 87
7.4.1.38 nppiConvert_8u16u_C4R . . . 88
7.4.1.39 nppiConvert_8u32f_C1R . . . 88
7.4.1.40 nppiCopy_16s_AC4R . . . 88
7.4.1.41 nppiCopy_16s_C1R . . . 89
7.4.1.42 nppiCopy_16s_C4R . . . 89

7.4.1.43 nppiCopy_16u_AC4R 89
7.4.1.44 nppiCopy_16u_C1R 90
7.4.1.45 nppiCopy_16u_C4R 90
7.4.1.46 nppiCopy_32f_AC4R 90
7.4.1.47 nppiCopy_32f_C1R 91
7.4.1.48 nppiCopy_32f_C4R 91
7.4.1.49 nppiCopy_32s_AC4R 91
7.4.1.50 nppiCopy_32s_C1R 92
7.4.1.51 nppiCopy_32s_C4R 92
7.4.1.52 nppiCopy_8u_AC4R 92
7.4.1.53 nppiCopy_8u_C1R 93
7.4.1.54 nppiCopy_8u_C4R 93
7.4.1.55 nppiCopyConstBorder_32s_C1R 93
7.4.1.56 nppiCopyConstBorder_8u_AC4R 94
7.4.1.57 nppiCopyConstBorder_8u_C1R 94
7.4.1.58 nppiCopyConstBorder_8u_C4R 95
7.4.1.59 nppiDCTQuantFwd8x8LS_JPEG_8u16s_C1R 95
7.4.1.60 nppiDCTQuantInv8x8LS_JPEG_16s8u_C1R 96
7.4.1.61 nppiDilate_8u_C1R 96
7.4.1.62 nppiDilate_8u_C4R 97
7.4.1.63 nppiDiv_32f_C1R 97
7.4.1.64 nppiDiv_32s_C1R 98
7.4.1.65 nppiDiv_8u_AC4RSfs 98
7.4.1.66 nppiDiv_8u_C1RSfs 98
7.4.1.67 nppiDiv_8u_C4RSfs 99
7.4.1.68 nppiDivC_32f_C1R 99
7.4.1.69 nppiDivC_32fc_C1R 100
7.4.1.70 nppiErode_8u_C1R 100
7.4.1.71 nppiErode_8u_C4R 101
7.4.1.72 nppiEvenLevelsHost_32s 101
7.4.1.73 nppiExp_32f_C1R 101
7.4.1.74 nppiFilter_8u_C1R 102
7.4.1.75 nppiFilter_8u_C4R 102
7.4.1.76 nppiFilterBox_8u_C1R 103
7.4.1.77 nppiFilterBox_8u_C4R 103
7.4.1.78 nppiFilterColumn_8u_C1R 104

7.4.1.79 nppiFilterColumn_8u_C4R 104
7.4.1.80 nppiFilterMax_8u_C1R 105
7.4.1.81 nppiFilterMax_8u_C4R 105
7.4.1.82 nppiFilterMin_8u_C1R 106
7.4.1.83 nppiFilterMin_8u_C4R 106
7.4.1.84 nppiFilterRow_8u_C1R 107
7.4.1.85 nppiFilterRow_8u_C4R 107
7.4.1.86 nppiFree 108
7.4.1.87 nppiGetAffineBound 108
7.4.1.88 nppiGetAffineQuad 108
7.4.1.89 nppiGetAffineTransform 109
7.4.1.90 nppiGetPerspectiveBound 109
7.4.1.91 nppiGetPerspectiveQuad 110
7.4.1.92 nppiGetPerspectiveTransform 110
7.4.1.93 nppiGraphcut_32s8u 110
7.4.1.94 nppiGraphcutGetSize 111
7.4.1.95 nppiHistogramEven_16s_AC4R 112
7.4.1.96 nppiHistogramEven_16s_C1R 112
7.4.1.97 nppiHistogramEven_16s_C4R 113
7.4.1.98 nppiHistogramEven_16u_AC4R 113
7.4.1.99 nppiHistogramEven_16u_C1R 114
7.4.1.100 nppiHistogramEven_16u_C4R 114
7.4.1.101 nppiHistogramEven_8u_AC4R 114
7.4.1.102 nppiHistogramEven_8u_C1R 115
7.4.1.103 nppiHistogramEven_8u_C4R 115
7.4.1.104 nppiHistogramEvenGetBufferSize_16s_AC4R 116
7.4.1.105 nppiHistogramEvenGetBufferSize_16s_C1R 116
7.4.1.106 nppiHistogramEvenGetBufferSize_16s_C4R 116
7.4.1.107 nppiHistogramEvenGetBufferSize_16u_AC4R 117
7.4.1.108 nppiHistogramEvenGetBufferSize_16u_C1R 117
7.4.1.109 nppiHistogramEvenGetBufferSize_16u_C4R 117
7.4.1.110 nppiHistogramEvenGetBufferSize_8u_AC4R 118
7.4.1.111 nppiHistogramEvenGetBufferSize_8u_C1R 118
7.4.1.112 nppiHistogramEvenGetBufferSize_8u_C4R 118
7.4.1.113 nppiHistogramRange_16s_AC4R 118
7.4.1.114 nppiHistogramRange_16s_C1R 119

7.4.1.115 nppiHistogramRange_16s_C4R 119
7.4.1.116 nppiHistogramRange_16u_AC4R 120
7.4.1.117 nppiHistogramRange_16u_C1R 120
7.4.1.118 nppiHistogramRange_16u_C4R 121
7.4.1.119 nppiHistogramRange_32f_AC4R 121
7.4.1.120 nppiHistogramRange_32f_C1R 122
7.4.1.121 nppiHistogramRange_32f_C4R 122
7.4.1.122 nppiHistogramRange_8u_AC4R 122
7.4.1.123 nppiHistogramRange_8u_C1R 123
7.4.1.124 nppiHistogramRange_8u_C4R 123
7.4.1.125 nppiHistogramRangeGetBufferSize_16s_AC4R 124
7.4.1.126 nppiHistogramRangeGetBufferSize_16s_C1R 124
7.4.1.127 nppiHistogramRangeGetBufferSize_16s_C4R 124
7.4.1.128 nppiHistogramRangeGetBufferSize_16u_AC4R 125
7.4.1.129 nppiHistogramRangeGetBufferSize_16u_C1R 125
7.4.1.130 nppiHistogramRangeGetBufferSize_16u_C4R 125
7.4.1.131 nppiHistogramRangeGetBufferSize_32f_AC4R 125
7.4.1.132 nppiHistogramRangeGetBufferSize_32f_C1R 126
7.4.1.133 nppiHistogramRangeGetBufferSize_32f_C4R 126
7.4.1.134 nppiHistogramRangeGetBufferSize_8u_AC4R 126
7.4.1.135 nppiHistogramRangeGetBufferSize_8u_C1R 127
7.4.1.136 nppiHistogramRangeGetBufferSize_8u_C4R 127
7.4.1.137 nppiLn_32f_C1R 127
7.4.1.138 nppiLUT_Linear_8u_AC4R 128
7.4.1.139 nppiLUT_Linear_8u_C1R 128
7.4.1.140 nppiLUT_Linear_8u_C3R 129
7.4.1.141 nppiMagnitude_32fc32f_C1R 129
7.4.1.142 nppiMagnitudeSqr_32fc32f_C1R 129
7.4.1.143 nppiMalloc_16s_C1 130
7.4.1.144 nppiMalloc_16s_C2 130
7.4.1.145 nppiMalloc_16s_C4 130
7.4.1.146 nppiMalloc_16u_C1 131
7.4.1.147 nppiMalloc_16u_C2 131
7.4.1.148 nppiMalloc_16u_C3 131
7.4.1.149 nppiMalloc_16u_C4 132
7.4.1.150 nppiMalloc_32f_C1 132

7.4.1.151 nppiMalloc_32f_C2 132
7.4.1.152 nppiMalloc_32f_C3 132
7.4.1.153 nppiMalloc_32f_C4 133
7.4.1.154 nppiMalloc_32s_C1 133
7.4.1.155 nppiMalloc_32s_C3 133
7.4.1.156 nppiMalloc_32s_C4 134
7.4.1.157 nppiMalloc_8u_C1 134
7.4.1.158 nppiMalloc_8u_C2 134
7.4.1.159 nppiMalloc_8u_C3 134
7.4.1.160 nppiMalloc_8u_C4 135
7.4.1.161 nppiMean_StdDev_8u_C1R 135
7.4.1.162 nppiMinMax_8u_C1R 135
7.4.1.163 nppiMinMax_8u_C4R 136
7.4.1.164 nppiMinMaxGetBufferSize_8u_C1R 136
7.4.1.165 nppiMinMaxGetBufferSize_8u_C4R 136
7.4.1.166 nppiMirror_8u_C1R 137
7.4.1.167 nppiMirror_8u_C4R 137
7.4.1.168 nppiMul_32f_C1R 138
7.4.1.169 nppiMul_32s_C1R 138
7.4.1.170 nppiMul_8u_AC4RSfs 138
7.4.1.171 nppiMul_8u_C1RSfs 139
7.4.1.172 nppiMul_8u_C4RSfs 139
7.4.1.173 nppiMulC_32f_C1R 140
7.4.1.174 nppiMulC_32fc_C1R 140
7.4.1.175 nppiNormDiff_Inf_8u_C1R 141
7.4.1.176 nppiNormDiff_L1_8u_C1R 141
7.4.1.177 nppiNormDiff_L2_8u_C1R 141
7.4.1.178 nppiQuantFwdRawTableInit_JPEG_8u 142
7.4.1.179 nppiQuantFwdTableInit_JPEG_8u16u 142
7.4.1.180 nppiQuantInvTableInit_JPEG_8u16u 142
7.4.1.181 nppiRectStdDev_32s32f_C1R 142
7.4.1.182 nppiReductionGetBufferHostSize_8u_C1R 143
7.4.1.183 nppiReductionGetBufferHostSize_8u_C4R 143
7.4.1.184 nppiResize_8u_C1R 143
7.4.1.185 nppiResize_8u_C4R 144
7.4.1.186 nppiRGBToYCbCr420_8u_C3P3R 145

7.4.1.187 nppiRGBToYCbCr422_8u_C3C2R 145
7.4.1.188 nppiRGBToYCbCr_8u_AC4R 145
7.4.1.189 nppiRGBToYCbCr_8u_C3R 146
7.4.1.190 nppiRGBToYCbCr_8u_P3R 146
7.4.1.191 nppiRotate_8u_C1R 146
7.4.1.192 nppiRotate_8u_C4R 147
7.4.1.193 nppiSet_16s_AC4MR 148
7.4.1.194 nppiSet_16s_AC4R 148
7.4.1.195 nppiSet_16s_C1MR 148
7.4.1.196 nppiSet_16s_C1R 149
7.4.1.197 nppiSet_16s_C2R 149
7.4.1.198 nppiSet_16s_C4CR 149
7.4.1.199 nppiSet_16s_C4MR 150
7.4.1.200 nppiSet_16s_C4R 150
7.4.1.201 nppiSet_16u_AC4MR 151
7.4.1.202 nppiSet_16u_AC4R 151
7.4.1.203 nppiSet_16u_C1MR 151
7.4.1.204 nppiSet_16u_C1R 152
7.4.1.205 nppiSet_16u_C2R 152
7.4.1.206 nppiSet_16u_C4CR 152
7.4.1.207 nppiSet_16u_C4MR 153
7.4.1.208 nppiSet_16u_C4R 153
7.4.1.209 nppiSet_32f_AC4MR 154
7.4.1.210 nppiSet_32f_AC4R 154
7.4.1.211 nppiSet_32f_C1MR 154
7.4.1.212 nppiSet_32f_C1R 155
7.4.1.213 nppiSet_32f_C4CR 155
7.4.1.214 nppiSet_32f_C4MR 155
7.4.1.215 nppiSet_32f_C4R 156
7.4.1.216 nppiSet_32s_AC4MR 156
7.4.1.217 nppiSet_32s_AC4R 157
7.4.1.218 nppiSet_32s_C1MR 157
7.4.1.219 nppiSet_32s_C1R 157
7.4.1.220 nppiSet_32s_C4CR 158
7.4.1.221 nppiSet_32s_C4MR 158
7.4.1.222 nppiSet_32s_C4R 158

7.4.1.223 nppiSet_8u_AC4MR 159
7.4.1.224 nppiSet_8u_AC4R 159
7.4.1.225 nppiSet_8u_C1MR 160
7.4.1.226 nppiSet_8u_C1R 160
7.4.1.227 nppiSet_8u_C4CR 160
7.4.1.228 nppiSet_8u_C4MR 161
7.4.1.229 nppiSet_8u_C4R 161
7.4.1.230 nppiSetDefaultQuantTable 161
7.4.1.231 nppiSqrIntegral_8u32s32f_C1R 162
7.4.1.232 nppiSub_32f_C1R 162
7.4.1.233 nppiSub_32s_C1R 163
7.4.1.234 nppiSub_8u_AC4RSfs 163
7.4.1.235 nppiSub_8u_C1RSfs 163
7.4.1.236 nppiSub_8u_C4RSfs 164
7.4.1.237 nppiSubC_32f_C1R 164
7.4.1.238 nppiSubC_32fc_C1R 165
7.4.1.239 nppiSum_8u_C1R 165
7.4.1.240 nppiSum_8u_C4R 165
7.4.1.241 nppiSumWindowColumn_8u32f_C1R 166
7.4.1.242 nppiSumWindowRow_8u32f_C1R 166
7.4.1.243 nppiSwapChannels_8u_C4IR 167
7.4.1.244 nppiThreshold_32f_C1R 167
7.4.1.245 nppiThreshold_8u_AC4R 168
7.4.1.246 nppiTranspose_8u_C1R 168
7.4.1.247 nppiWarpAffine_16u_AC4R 168
7.4.1.248 nppiWarpAffine_16u_C1R 169
7.4.1.249 nppiWarpAffine_16u_C3R 170
7.4.1.250 nppiWarpAffine_16u_C4R 170
7.4.1.251 nppiWarpAffine_16u_P3R 170
7.4.1.252 nppiWarpAffine_16u_P4R 170
7.4.1.253 nppiWarpAffine_32f_AC4R 171
7.4.1.254 nppiWarpAffine_32f_C1R 171
7.4.1.255 nppiWarpAffine_32f_C3R 172
7.4.1.256 nppiWarpAffine_32f_C4R 172
7.4.1.257 nppiWarpAffine_32f_P3R 172
7.4.1.258 nppiWarpAffine_32f_P4R 172

7.4.1.259 nppiWarpAffine_32s_AC4R . . . 173
7.4.1.260 nppiWarpAffine_32s_C1R . . . 173
7.4.1.261 nppiWarpAffine_32s_C3R . . . 174
7.4.1.262 nppiWarpAffine_32s_C4R . . . 174
7.4.1.263 nppiWarpAffine_32s_P3R . . . 174
7.4.1.264 nppiWarpAffine_32s_P4R . . . 174
7.4.1.265 nppiWarpAffine_8u_AC4R . . . 175
7.4.1.266 nppiWarpAffine_8u_C1R . . . 175
7.4.1.267 nppiWarpAffine_8u_C3R . . . 176
7.4.1.268 nppiWarpAffine_8u_C4R . . . 176
7.4.1.269 nppiWarpAffine_8u_P3R . . . 176
7.4.1.270 nppiWarpAffine_8u_P4R . . . 177
7.4.1.271 nppiWarpAffineBack_16u_AC4R . . . 177
7.4.1.272 nppiWarpAffineBack_16u_C1R . . . 177
7.4.1.273 nppiWarpAffineBack_16u_C3R . . . 178
7.4.1.274 nppiWarpAffineBack_16u_C4R . . . 178
7.4.1.275 nppiWarpAffineBack_16u_P3R . . . 179
7.4.1.276 nppiWarpAffineBack_16u_P4R . . . 179
7.4.1.277 nppiWarpAffineBack_32f_AC4R . . . 179
7.4.1.278 nppiWarpAffineBack_32f_C1R . . . 179
7.4.1.279 nppiWarpAffineBack_32f_C3R . . . 180
7.4.1.280 nppiWarpAffineBack_32f_C4R . . . 180
7.4.1.281 nppiWarpAffineBack_32f_P3R . . . 181
7.4.1.282 nppiWarpAffineBack_32f_P4R . . . 181
7.4.1.283 nppiWarpAffineBack_32s_AC4R . . . 181
7.4.1.284 nppiWarpAffineBack_32s_C1R . . . 181
7.4.1.285 nppiWarpAffineBack_32s_C3R . . . 182
7.4.1.286 nppiWarpAffineBack_32s_C4R . . . 182
7.4.1.287 nppiWarpAffineBack_32s_P3R . . . 183
7.4.1.288 nppiWarpAffineBack_32s_P4R . . . 183
7.4.1.289 nppiWarpAffineBack_8u_AC4R . . . 183
7.4.1.290 nppiWarpAffineBack_8u_C1R . . . 183
7.4.1.291 nppiWarpAffineBack_8u_C3R . . . 184
7.4.1.292 nppiWarpAffineBack_8u_C4R . . . 185
7.4.1.293 nppiWarpAffineBack_8u_P3R . . . 185
7.4.1.294 nppiWarpAffineBack_8u_P4R . . . 185

7.4.1.295 nppiWarpAffineQuad_16u_AC4R 185
7.4.1.296 nppiWarpAffineQuad_16u_C1R 185
7.4.1.297 nppiWarpAffineQuad_16u_C3R 186
7.4.1.298 nppiWarpAffineQuad_16u_C4R 187
7.4.1.299 nppiWarpAffineQuad_16u_P3R 187
7.4.1.300 nppiWarpAffineQuad_16u_P4R 187
7.4.1.301 nppiWarpAffineQuad_32f_AC4R 187
7.4.1.302 nppiWarpAffineQuad_32f_C1R 187
7.4.1.303 nppiWarpAffineQuad_32f_C3R 188
7.4.1.304 nppiWarpAffineQuad_32f_C4R 188
7.4.1.305 nppiWarpAffineQuad_32f_P3R 189
7.4.1.306 nppiWarpAffineQuad_32f_P4R 189
7.4.1.307 nppiWarpAffineQuad_32s_AC4R 189
7.4.1.308 nppiWarpAffineQuad_32s_C1R 189
7.4.1.309 nppiWarpAffineQuad_32s_C3R 190
7.4.1.310 nppiWarpAffineQuad_32s_C4R 190
7.4.1.311 nppiWarpAffineQuad_32s_P3R 190
7.4.1.312 nppiWarpAffineQuad_32s_P4R 191
7.4.1.313 nppiWarpAffineQuad_8u_AC4R 191
7.4.1.314 nppiWarpAffineQuad_8u_C1R 191
7.4.1.315 nppiWarpAffineQuad_8u_C3R 192
7.4.1.316 nppiWarpAffineQuad_8u_C4R 192
7.4.1.317 nppiWarpAffineQuad_8u_P3R 192
7.4.1.318 nppiWarpAffineQuad_8u_P4R 193
7.4.1.319 nppiWarpPerspective_16u_AC4R 193
7.4.1.320 nppiWarpPerspective_16u_C1R 193
7.4.1.321 nppiWarpPerspective_16u_C3R 194
7.4.1.322 nppiWarpPerspective_16u_C4R 194
7.4.1.323 nppiWarpPerspective_16u_P3R 195
7.4.1.324 nppiWarpPerspective_16u_P4R 195
7.4.1.325 nppiWarpPerspective_32f_AC4R 195
7.4.1.326 nppiWarpPerspective_32f_C1R 195
7.4.1.327 nppiWarpPerspective_32f_C3R 196
7.4.1.328 nppiWarpPerspective_32f_C4R 196
7.4.1.329 nppiWarpPerspective_32f_P3R 197
7.4.1.330 nppiWarpPerspective_32f_P4R 197

7.4.1.331 nppiWarpPerspective_32s_AC4R 197
7.4.1.332 nppiWarpPerspective_32s_C1R 197
7.4.1.333 nppiWarpPerspective_32s_C3R 198
7.4.1.334 nppiWarpPerspective_32s_C4R 198
7.4.1.335 nppiWarpPerspective_32s_P3R 199
7.4.1.336 nppiWarpPerspective_32s_P4R 199
7.4.1.337 nppiWarpPerspective_8u_AC4R 199
7.4.1.338 nppiWarpPerspective_8u_C1R 199
7.4.1.339 nppiWarpPerspective_8u_C3R 200
7.4.1.340 nppiWarpPerspective_8u_C4R 201
7.4.1.341 nppiWarpPerspective_8u_P3R 201
7.4.1.342 nppiWarpPerspective_8u_P4R 201
7.4.1.343 nppiWarpPerspectiveBack_16u_AC4R 201
7.4.1.344 nppiWarpPerspectiveBack_16u_C1R 201
7.4.1.345 nppiWarpPerspectiveBack_16u_C3R 203
7.4.1.346 nppiWarpPerspectiveBack_16u_C4R 203
7.4.1.347 nppiWarpPerspectiveBack_16u_P3R 203
7.4.1.348 nppiWarpPerspectiveBack_16u_P4R 203
7.4.1.349 nppiWarpPerspectiveBack_32f_AC4R 203
7.4.1.350 nppiWarpPerspectiveBack_32f_C1R 204
7.4.1.351 nppiWarpPerspectiveBack_32f_C3R 205
7.4.1.352 nppiWarpPerspectiveBack_32f_C4R 205
7.4.1.353 nppiWarpPerspectiveBack_32f_P3R 205
7.4.1.354 nppiWarpPerspectiveBack_32f_P4R 205
7.4.1.355 nppiWarpPerspectiveBack_32s_AC4R 205
7.4.1.356 nppiWarpPerspectiveBack_32s_C1R 206
7.4.1.357 nppiWarpPerspectiveBack_32s_C3R 207
7.4.1.358 nppiWarpPerspectiveBack_32s_C4R 207
7.4.1.359 nppiWarpPerspectiveBack_32s_P3R 207
7.4.1.360 nppiWarpPerspectiveBack_32s_P4R 207
7.4.1.361 nppiWarpPerspectiveBack_8u_AC4R 207
7.4.1.362 nppiWarpPerspectiveBack_8u_C1R 208
7.4.1.363 nppiWarpPerspectiveBack_8u_C3R 209
7.4.1.364 nppiWarpPerspectiveBack_8u_C4R 209
7.4.1.365 nppiWarpPerspectiveBack_8u_P3R 209
7.4.1.366 nppiWarpPerspectiveBack_8u_P4R 209

7.4.1.367 nppiWarpPerspectiveQuad_16u_AC4R 210
7.4.1.368 nppiWarpPerspectiveQuad_16u_C1R 210
7.4.1.369 nppiWarpPerspectiveQuad_16u_C3R 211
7.4.1.370 nppiWarpPerspectiveQuad_16u_C4R 211
7.4.1.371 nppiWarpPerspectiveQuad_16u_P3R 211
7.4.1.372 nppiWarpPerspectiveQuad_16u_P4R 211
7.4.1.373 nppiWarpPerspectiveQuad_32f_AC4R 212
7.4.1.374 nppiWarpPerspectiveQuad_32f_C1R 212
7.4.1.375 nppiWarpPerspectiveQuad_32f_C3R 213
7.4.1.376 nppiWarpPerspectiveQuad_32f_C4R 213
7.4.1.377 nppiWarpPerspectiveQuad_32f_P3R 213
7.4.1.378 nppiWarpPerspectiveQuad_32f_P4R 213
7.4.1.379 nppiWarpPerspectiveQuad_32s_AC4R 213
7.4.1.380 nppiWarpPerspectiveQuad_32s_C1R 214
7.4.1.381 nppiWarpPerspectiveQuad_32s_C3R 214
7.4.1.382 nppiWarpPerspectiveQuad_32s_C4R 215
7.4.1.383 nppiWarpPerspectiveQuad_32s_P3R 215
7.4.1.384 nppiWarpPerspectiveQuad_32s_P4R 215
7.4.1.385 nppiWarpPerspectiveQuad_8u_AC4R 215
7.4.1.386 nppiWarpPerspectiveQuad_8u_C1R 215
7.4.1.387 nppiWarpPerspectiveQuad_8u_C3R 216
7.4.1.388 nppiWarpPerspectiveQuad_8u_C4R 217
7.4.1.389 nppiWarpPerspectiveQuad_8u_P3R 217
7.4.1.390 nppiWarpPerspectiveQuad_8u_P4R 217
7.4.1.391 nppiYCbCr420ToRGB_8u_P3C3R 217
7.4.1.392 nppiYCbCr420ToYCbCr411_8u_P3P2R 218
7.4.1.393 nppiYCbCr420ToYCbCr422_8u_P3R 218
7.4.1.394 nppiYCbCr422ToRGB_8u_C2C3R 218
7.4.1.395 nppiYCbCr422ToYCbCr411_8u_P3R 219
7.4.1.396 nppiYCbCr422ToYCbCr420_8u_P3R 219
7.4.1.397 nppiYCbCrToRGB_8u_AC4R 219
7.4.1.398 nppiYCbCrToRGB_8u_C3R 220
7.4.1.399 nppiYCbCrToRGB_8u_P3R 220
7.5 NPP Signal Processing 221
7.5.1 Function Documentation 258
7.5.1.1 npps10Log10_32s_ISfs 258

7.5.1.2 npps10Log10_32s_Sfs 258
7.5.1.3 nppsAbs_16s 259
7.5.1.4 nppsAbs_16s_I 259
7.5.1.5 nppsAbs_32f 259
7.5.1.6 nppsAbs_32f_I 259
7.5.1.7 nppsAbs_32s 260
7.5.1.8 nppsAbs_32s_I 260
7.5.1.9 nppsAbs_64f 260
7.5.1.10 nppsAbs_64f_I 260
7.5.1.11 nppsAdd_16s 261
7.5.1.12 nppsAdd_16s32f 261
7.5.1.13 nppsAdd_16s32s_I 261
7.5.1.14 nppsAdd_16s_I 262
7.5.1.15 nppsAdd_16s_ISfs 262
7.5.1.16 nppsAdd_16s_Sfs 262
7.5.1.17 nppsAdd_16sc_ISfs 263
7.5.1.18 nppsAdd_16sc_Sfs 263
7.5.1.19 nppsAdd_16u 263
7.5.1.20 nppsAdd_16u_ISfs 264
7.5.1.21 nppsAdd_16u_Sfs 264
7.5.1.22 nppsAdd_32f 264
7.5.1.23 nppsAdd_32f_I 265
7.5.1.24 nppsAdd_32fc 265
7.5.1.25 nppsAdd_32fc_I 265
7.5.1.26 nppsAdd_32s_ISfs 265
7.5.1.27 nppsAdd_32s_Sfs 266
7.5.1.28 nppsAdd_32sc_ISfs 266
7.5.1.29 nppsAdd_32sc_Sfs 267
7.5.1.30 nppsAdd_32u 267
7.5.1.31 nppsAdd_64f 267
7.5.1.32 nppsAdd_64f_I 268
7.5.1.33 nppsAdd_64fc 268
7.5.1.34 nppsAdd_64fc_I 268
7.5.1.35 nppsAdd_64s_Sfs 268
7.5.1.36 nppsAdd_8u16u 269
7.5.1.37 nppsAdd_8u_ISfs 269

7.5.1.38 nppsAdd_8u_Sfs 270
7.5.1.39 nppsAddC_16s_ISfs 270
7.5.1.40 nppsAddC_16s_Sfs 270
7.5.1.41 nppsAddC_16sc_ISfs 271
7.5.1.42 nppsAddC_16sc_Sfs 271
7.5.1.43 nppsAddC_16u_ISfs 271
7.5.1.44 nppsAddC_16u_Sfs 272
7.5.1.45 nppsAddC_32f 272
7.5.1.46 nppsAddC_32f_I 272
7.5.1.47 nppsAddC_32fc 273
7.5.1.48 nppsAddC_32fc_I 273
7.5.1.49 nppsAddC_32s_ISfs 273
7.5.1.50 nppsAddC_32s_Sfs 274
7.5.1.51 nppsAddC_32sc_ISfs 274
7.5.1.52 nppsAddC_32sc_Sfs 274
7.5.1.53 nppsAddC_64f 275
7.5.1.54 nppsAddC_64f_I 275
7.5.1.55 nppsAddC_64fc 275
7.5.1.56 nppsAddC_64fc_I 276
7.5.1.57 nppsAddC_8u_ISfs 276
7.5.1.58 nppsAddC_8u_Sfs 276
7.5.1.59 nppsAddProduct_16s32s_Sfs 277
7.5.1.60 nppsAddProduct_16s_Sfs 277
7.5.1.61 nppsAddProduct_32f 277
7.5.1.62 nppsAddProduct_32fc 278
7.5.1.63 nppsAddProduct_32s_Sfs 278
7.5.1.64 nppsAddProduct_64f 278
7.5.1.65 nppsAddProduct_64fc 279
7.5.1.66 nppsAddProductC_16s_ISfs 279
7.5.1.67 nppsAddProductC_16s_Sfs 280
7.5.1.68 nppsAddProductC_16sc_ISfs 280
7.5.1.69 nppsAddProductC_16sc_Sfs 280
7.5.1.70 nppsAddProductC_16u_ISfs 281
7.5.1.71 nppsAddProductC_16u_Sfs 281
7.5.1.72 nppsAddProductC_32f 281
7.5.1.73 nppsAddProductC_32f_I 282

7.5.1.74 nppsAddProductC_32s_ISfs . . . 282
7.5.1.75 nppsAddProductC_32s_Sfs . . . 282
7.5.1.76 nppsAddProductC_32sc_ISfs . . . 283
7.5.1.77 nppsAddProductC_32sc_Sfs . . . 283
7.5.1.78 nppsAddProductC_8u_ISfs . . . 283
7.5.1.79 nppsAddProductC_8u_Sfs . . . 284
7.5.1.80 nppsAnd_16u . . . 284
7.5.1.81 nppsAnd_16u_I . . . 284
7.5.1.82 nppsAnd_32u . . . 285
7.5.1.83 nppsAnd_32u_I . . . 285
7.5.1.84 nppsAnd_8u . . . 285
7.5.1.85 nppsAnd_8u_I . . . 286
7.5.1.86 nppsAndC_16u . . . 286
7.5.1.87 nppsAndC_16u_I . . . 286
7.5.1.88 nppsAndC_32u . . . 286
7.5.1.89 nppsAndC_32u_I . . . 287
7.5.1.90 nppsAndC_8u . . . 287
7.5.1.91 nppsAndC_8u_I . . . 287
7.5.1.92 nppsArctan_32f . . . 288
7.5.1.93 nppsArctan_32f_I . . . 288
7.5.1.94 nppsArctan_64f . . . 288
7.5.1.95 nppsArctan_64f_I . . . 288
7.5.1.96 nppsCauchy_32f_I . . . 289
7.5.1.97 nppsCauchyD_32f_I . . . 289
7.5.1.98 nppsCauchyDD2_32f_I . . . 289
7.5.1.99 nppsCopy_16s . . . 289
7.5.1.100 nppsCopy_16sc . . . 290
7.5.1.101 nppsCopy_32f . . . 290
7.5.1.102 nppsCopy_32fc . . . 290
7.5.1.103 nppsCopy_32s . . . 291
7.5.1.104 nppsCopy_32sc . . . 291
7.5.1.105 nppsCopy_64fc . . . 291
7.5.1.106 nppsCopy_64s . . . 291
7.5.1.107 nppsCopy_64sc . . . 292
7.5.1.108 nppsCopy_8u . . . 292
7.5.1.109 nppsCubrt_32f . . . 292

7.5.1.110 nppsCubrt_32s16s_Sfs 293
7.5.1.111 nppsDiv_16s_ISfs 293
7.5.1.112 nppsDiv_16s_Sfs 293
7.5.1.113 nppsDiv_16sc_ISfs 294
7.5.1.114 nppsDiv_16sc_Sfs 294
7.5.1.115 nppsDiv_16u_ISfs 294
7.5.1.116 nppsDiv_16u_Sfs 295
7.5.1.117 nppsDiv_32f 295
7.5.1.118 nppsDiv_32f_I 295
7.5.1.119 nppsDiv_32fc 296
7.5.1.120 nppsDiv_32fc_I 296
7.5.1.121 nppsDiv_32s16s_Sfs 296
7.5.1.122 nppsDiv_32s_ISfs 297
7.5.1.123 nppsDiv_32s_Sfs 297
7.5.1.124 nppsDiv_64f 297
7.5.1.125 nppsDiv_64f_I 298
7.5.1.126 nppsDiv_64fc 298
7.5.1.127 nppsDiv_64fc_I 298
7.5.1.128 nppsDiv_8u_ISfs 298
7.5.1.129 nppsDiv_8u_Sfs 299
7.5.1.130 nppsDiv_Round_16s_ISfs 299
7.5.1.131 nppsDiv_Round_16s_Sfs 300
7.5.1.132 nppsDiv_Round_16u_ISfs 300
7.5.1.133 nppsDiv_Round_16u_Sfs 300
7.5.1.134 nppsDiv_Round_8u_ISfs 301
7.5.1.135 nppsDiv_Round_8u_Sfs 301
7.5.1.136 nppsDivC_16s_ISfs 301
7.5.1.137 nppsDivC_16s_Sfs 302
7.5.1.138 nppsDivC_16sc_ISfs 302
7.5.1.139 nppsDivC_16sc_Sfs 302
7.5.1.140 nppsDivC_16u_ISfs 303
7.5.1.141 nppsDivC_16u_Sfs 303
7.5.1.142 nppsDivC_32f 303
7.5.1.143 nppsDivC_32f_I 304
7.5.1.144 nppsDivC_32fc 304
7.5.1.145 nppsDivC_32fc_I 304

7.5.1.146 nppsDivC_32s_ISfs 304
7.5.1.147 nppsDivC_32s_Sfs 305
7.5.1.148 nppsDivC_32sc_ISfs 305
7.5.1.149 nppsDivC_32sc_Sfs 306
7.5.1.150 nppsDivC_64f 306
7.5.1.151 nppsDivC_64f_I 306
7.5.1.152 nppsDivC_64fc 307
7.5.1.153 nppsDivC_64fc_I 307
7.5.1.154 nppsDivC_8u_ISfs 307
7.5.1.155 nppsDivC_8u_Sfs 308
7.5.1.156 nppsDivCRev_16s 308
7.5.1.157 nppsDivCRev_16s_I 308
7.5.1.158 nppsDivCRev_16u 309
7.5.1.159 nppsDivCRev_16u_I 309
7.5.1.160 nppsDivCRev_32f 309
7.5.1.161 nppsDivCRev_32f_I 310
7.5.1.162 nppsDivCRev_32s 310
7.5.1.163 nppsDivCRev_32s_I 310
7.5.1.164 nppsDivCRev_64f 310
7.5.1.165 nppsDivCRev_64f_I 311
7.5.1.166 nppsDivCRev_8u 311
7.5.1.167 nppsDivCRev_8u_I 311
7.5.1.168 nppsExp_16s_ISfs 312
7.5.1.169 nppsExp_16s_Sfs 312
7.5.1.170 nppsExp_32f 312
7.5.1.171 nppsExp_32f64f 312
7.5.1.172 nppsExp_32f_I 313
7.5.1.173 nppsExp_32s_ISfs 313
7.5.1.174 nppsExp_32s_Sfs 313
7.5.1.175 nppsExp_64f 314
7.5.1.176 nppsExp_64f_I 314
7.5.1.177 nppsExp_64s_ISfs 314
7.5.1.178 nppsExp_64s_Sfs 314
7.5.1.179 nppsFree 315
7.5.1.180 nppsIntegral_32s 315
7.5.1.181 nppsIntegralGetBufferSize_32s 315

7.5.1.182 nppsLn_16s_ISfs . . . 315
7.5.1.183 nppsLn_16s_Sfs . . . 315
7.5.1.184 nppsLn_32f . . . 316
7.5.1.185 nppsLn_32f_I . . . 316
7.5.1.186 nppsLn_32s16s_Sfs . . . 316
7.5.1.187 nppsLn_32s_ISfs . . . 316
7.5.1.188 nppsLn_32s_Sfs . . . 317
7.5.1.189 nppsLn_64f . . . 317
7.5.1.190 nppsLn_64f32f . . . 317
7.5.1.191 nppsLn_64f_I . . . 318
7.5.1.192 nppsLShiftC_16s . . . 318
7.5.1.193 nppsLShiftC_16s_I . . . 318
7.5.1.194 nppsLShiftC_16u . . . 318
7.5.1.195 nppsLShiftC_16u_I . . . 319
7.5.1.196 nppsLShiftC_32s . . . 319
7.5.1.197 nppsLShiftC_32s_I . . . 319
7.5.1.198 nppsLShiftC_32u . . . 320
7.5.1.199 nppsLShiftC_32u_I . . . 320
7.5.1.200 nppsLShiftC_8u . . . 320
7.5.1.201 nppsLShiftC_8u_I . . . 321
7.5.1.202 nppsMalloc_16s . . . 321
7.5.1.203 nppsMalloc_16sc . . . 321
7.5.1.204 nppsMalloc_16u . . . 321
7.5.1.205 nppsMalloc_32f . . . 322
7.5.1.206 nppsMalloc_32fc . . . 322
7.5.1.207 nppsMalloc_32s . . . 322
7.5.1.208 nppsMalloc_32sc . . . 322
7.5.1.209 nppsMalloc_32u . . . 323
7.5.1.210 nppsMalloc_64f . . . 323
7.5.1.211 nppsMalloc_64fc . . . 323
7.5.1.212 nppsMalloc_64s . . . 323
7.5.1.213 nppsMalloc_64sc . . . 324
7.5.1.214 nppsMalloc_8u . . . 324
7.5.1.215 nppsMax_16s . . . 324
7.5.1.216 nppsMax_32f . . . 324
7.5.1.217 nppsMax_32s . . . 325

7.5.1.218 nppsMax_64f . . . 325
7.5.1.219 nppsMin_16s . . . 325
7.5.1.220 nppsMin_32f . . . 326
7.5.1.221 nppsMin_32s . . . 326
7.5.1.222 nppsMin_64f . . . 326
7.5.1.223 nppsMinMax_16s . . . 327
7.5.1.224 nppsMinMax_16u . . . 327
7.5.1.225 nppsMinMax_32f . . . 327
7.5.1.226 nppsMinMax_32s . . . 328
7.5.1.227 nppsMinMax_32u . . . 328
7.5.1.228 nppsMinMax_64f . . . 328
7.5.1.229 nppsMinMax_8u . . . 329
7.5.1.230 nppsMinMaxGetBufferSize_16s . . . 329
7.5.1.231 nppsMinMaxGetBufferSize_16u . . . 329
7.5.1.232 nppsMinMaxGetBufferSize_32f . . . 329
7.5.1.233 nppsMinMaxGetBufferSize_32s . . . 330
7.5.1.234 nppsMinMaxGetBufferSize_32u . . . 330
7.5.1.235 nppsMinMaxGetBufferSize_64f . . . 330
7.5.1.236 nppsMinMaxGetBufferSize_8u . . . 330
7.5.1.237 nppsMul_16s . . . 331
7.5.1.238 nppsMul_16s32f . . . 331
7.5.1.239 nppsMul_16s32s_Sfs . . . 331
7.5.1.240 nppsMul_16s_I . . . 332
7.5.1.241 nppsMul_16s_ISfs . . . 332
7.5.1.242 nppsMul_16s_Sfs . . . 332
7.5.1.243 nppsMul_16sc_ISfs . . . 333
7.5.1.244 nppsMul_16sc_Sfs . . . 333
7.5.1.245 nppsMul_16u16s_Sfs . . . 333
7.5.1.246 nppsMul_16u_ISfs . . . 334
7.5.1.247 nppsMul_16u_Sfs . . . 334
7.5.1.248 nppsMul_32f . . . 334
7.5.1.249 nppsMul_32f32fc . . . 335
7.5.1.250 nppsMul_32f32fc_I . . . 335
7.5.1.251 nppsMul_32f_I . . . 335
7.5.1.252 nppsMul_32fc . . . 336
7.5.1.253 nppsMul_32fc_I . . . 336

7.5.1.254 nppsMul_32s32sc_ISfs 336
7.5.1.255 nppsMul_32s32sc_Sfs 337
7.5.1.256 nppsMul_32s_ISfs 337
7.5.1.257 nppsMul_32s_Sfs 337
7.5.1.258 nppsMul_32sc_ISfs 338
7.5.1.259 nppsMul_32sc_Sfs 338
7.5.1.260 nppsMul_64f 338
7.5.1.261 nppsMul_64f_I 339
7.5.1.262 nppsMul_64fc 339
7.5.1.263 nppsMul_64fc_I 339
7.5.1.264 nppsMul_8u16u 339
7.5.1.265 nppsMul_8u_ISfs 340
7.5.1.266 nppsMul_8u_Sfs 340
7.5.1.267 nppsMul_Low_32s_Sfs 341
7.5.1.268 nppsMulC_16s_ISfs 341
7.5.1.269 nppsMulC_16s_Sfs 341
7.5.1.270 nppsMulC_16sc_ISfs 342
7.5.1.271 nppsMulC_16sc_Sfs 342
7.5.1.272 nppsMulC_16u_ISfs 342
7.5.1.273 nppsMulC_16u_Sfs 343
7.5.1.274 nppsMulC_32f 343
7.5.1.275 nppsMulC_32f16s_Sfs 343
7.5.1.276 nppsMulC_32f_I 344
7.5.1.277 nppsMulC_32fc 344
7.5.1.278 nppsMulC_32fc_I 344
7.5.1.279 nppsMulC_32s_ISfs 344
7.5.1.280 nppsMulC_32s_Sfs 345
7.5.1.281 nppsMulC_32sc_ISfs 345
7.5.1.282 nppsMulC_32sc_Sfs 346
7.5.1.283 nppsMulC_64f 346
7.5.1.284 nppsMulC_64f64s_ISfs 346
7.5.1.285 nppsMulC_64f_I 347
7.5.1.286 nppsMulC_64fc 347
7.5.1.287 nppsMulC_64fc_I 347
7.5.1.288 nppsMulC_8u_ISfs 347
7.5.1.289 nppsMulC_8u_Sfs 348

7.5.1.290 nppsMulC_Low_32f16s . . . 348
7.5.1.291 nppsNormalize_16s_Sfs . . . 349
7.5.1.292 nppsNormalize_16sc_Sfs . . . 349
7.5.1.293 nppsNormalize_32f . . . 349
7.5.1.294 nppsNormalize_32fc . . . 350
7.5.1.295 nppsNormalize_64f . . . 350
7.5.1.296 nppsNormalize_64fc . . . 350
7.5.1.297 nppsNot_16u . . . 351
7.5.1.298 nppsNot_16u_I . . . 351
7.5.1.299 nppsNot_32u . . . 351
7.5.1.300 nppsNot_32u_I . . . 351
7.5.1.301 nppsNot_8u . . . 352
7.5.1.302 nppsNot_8u_I . . . 352
7.5.1.303 nppsOr_16u . . . 352
7.5.1.304 nppsOr_16u_I . . . 352
7.5.1.305 nppsOr_32u . . . 353
7.5.1.306 nppsOr_32u_I . . . 353
7.5.1.307 nppsOr_8u . . . 353
7.5.1.308 nppsOr_8u_I . . . 354
7.5.1.309 nppsOrC_16u . . . 354
7.5.1.310 nppsOrC_16u_I . . . 354
7.5.1.311 nppsOrC_32u . . . 354
7.5.1.312 nppsOrC_32u_I . . . 355
7.5.1.313 nppsOrC_8u . . . 355
7.5.1.314 nppsOrC_8u_I . . . 355
7.5.1.315 nppsReductionGetBufferSize_16s . . . 356
7.5.1.316 nppsReductionGetBufferSize_16s_Sfs . . . 356
7.5.1.317 nppsReductionGetBufferSize_16sc . . . 356
7.5.1.318 nppsReductionGetBufferSize_16sc_Sfs . . . 356
7.5.1.319 nppsReductionGetBufferSize_16u . . . 357
7.5.1.320 nppsReductionGetBufferSize_32f . . . 357
7.5.1.321 nppsReductionGetBufferSize_32fc . . . 357
7.5.1.322 nppsReductionGetBufferSize_32s . . . 358
7.5.1.323 nppsReductionGetBufferSize_32s_Sfs . . . 358
7.5.1.324 nppsReductionGetBufferSize_32sc . . . 358
7.5.1.325 nppsReductionGetBufferSize_32u . . . 358

7.5.1.326 nppsReductionGetBufferSize_64f . . . 359
7.5.1.327 nppsReductionGetBufferSize_64fc . . . 359
7.5.1.328 nppsReductionGetBufferSize_64s . . . 359
7.5.1.329 nppsReductionGetBufferSize_8u . . . 360
7.5.1.330 nppsRShiftC_16s . . . 360
7.5.1.331 nppsRShiftC_16s_I . . . 360
7.5.1.332 nppsRShiftC_16u . . . 360
7.5.1.333 nppsRShiftC_16u_I . . . 361
7.5.1.334 nppsRShiftC_32s . . . 361
7.5.1.335 nppsRShiftC_32s_I . . . 361
7.5.1.336 nppsRShiftC_32u . . . 362
7.5.1.337 nppsRShiftC_32u_I . . . 362
7.5.1.338 nppsRShiftC_8u . . . 362
7.5.1.339 nppsRShiftC_8u_I . . . 363
7.5.1.340 nppsSet_16s . . . 363
7.5.1.341 nppsSet_16sc . . . 363
7.5.1.342 nppsSet_32f . . . 363
7.5.1.343 nppsSet_32fc . . . 364
7.5.1.344 nppsSet_32s . . . 364
7.5.1.345 nppsSet_32sc . . . 364
7.5.1.346 nppsSet_64f . . . 365
7.5.1.347 nppsSet_64fc . . . 365
7.5.1.348 nppsSet_64s . . . 365
7.5.1.349 nppsSet_64sc . . . 365
7.5.1.350 nppsSet_8u . . . 366
7.5.1.351 nppsSqr_16s_ISfs . . . 366
7.5.1.352 nppsSqr_16s_Sfs . . . 366
7.5.1.353 nppsSqr_16sc_ISfs . . . 367
7.5.1.354 nppsSqr_16sc_Sfs . . . 367
7.5.1.355 nppsSqr_16u_ISfs . . . 367
7.5.1.356 nppsSqr_16u_Sfs . . . 367
7.5.1.357 nppsSqr_32f . . . 368
7.5.1.358 nppsSqr_32f_I . . . 368
7.5.1.359 nppsSqr_32fc . . . 368
7.5.1.360 nppsSqr_32fc_I . . . 369
7.5.1.361 nppsSqr_64f . . . 369

7.5.1.362 nppsSqr_64f_I 369
7.5.1.363 nppsSqr_64fc 369
7.5.1.364 nppsSqr_64fc_I 370
7.5.1.365 nppsSqr_8u_ISfs 370
7.5.1.366 nppsSqr_8u_Sfs 370
7.5.1.367 nppsSqrt_16s_ISfs 370
7.5.1.368 nppsSqrt_16s_Sfs 371
7.5.1.369 nppsSqrt_16sc_ISfs 371
7.5.1.370 nppsSqrt_16sc_Sfs 371
7.5.1.371 nppsSqrt_16u_ISfs 372
7.5.1.372 nppsSqrt_16u_Sfs 372
7.5.1.373 nppsSqrt_32f 372
7.5.1.374 nppsSqrt_32f_I 372
7.5.1.375 nppsSqrt_32fc 373
7.5.1.376 nppsSqrt_32fc_I 373
7.5.1.377 nppsSqrt_32s16s_Sfs 373
7.5.1.378 nppsSqrt_64f 374
7.5.1.379 nppsSqrt_64f_I 374
7.5.1.380 nppsSqrt_64fc 374
7.5.1.381 nppsSqrt_64fc_I 374
7.5.1.382 nppsSqrt_64s16s_Sfs 375
7.5.1.383 nppsSqrt_64s_ISfs 375
7.5.1.384 nppsSqrt_64s_Sfs 375
7.5.1.385 nppsSqrt_8u_ISfs 376
7.5.1.386 nppsSqrt_8u_Sfs 376
7.5.1.387 nppsSub_16s 376
7.5.1.388 nppsSub_16s32f 377
7.5.1.389 nppsSub_16s_I 377
7.5.1.390 nppsSub_16s_ISfs 377
7.5.1.391 nppsSub_16s_Sfs 378
7.5.1.392 nppsSub_16sc_ISfs 378
7.5.1.393 nppsSub_16sc_Sfs 378
7.5.1.394 nppsSub_16u_ISfs 379
7.5.1.395 nppsSub_16u_Sfs 379
7.5.1.396 nppsSub_32f 379
7.5.1.397 nppsSub_32f_I 380

7.5.1.398 nppsSub_32fc 380
7.5.1.399 nppsSub_32fc_I 380
7.5.1.400 nppsSub_32s_ISfs 380
7.5.1.401 nppsSub_32s_Sfs 381
7.5.1.402 nppsSub_32sc_ISfs 381
7.5.1.403 nppsSub_32sc_Sfs 382
7.5.1.404 nppsSub_64f 382
7.5.1.405 nppsSub_64f_I 382
7.5.1.406 nppsSub_64fc 383
7.5.1.407 nppsSub_64fc_I 383
7.5.1.408 nppsSub_8u_ISfs 383
7.5.1.409 nppsSub_8u_Sfs 384
7.5.1.410 nppsSubC_16s_ISfs 384
7.5.1.411 nppsSubC_16s_Sfs 384
7.5.1.412 nppsSubC_16sc_ISfs 385
7.5.1.413 nppsSubC_16sc_Sfs 385
7.5.1.414 nppsSubC_16u_ISfs 385
7.5.1.415 nppsSubC_16u_Sfs 386
7.5.1.416 nppsSubC_32f 386
7.5.1.417 nppsSubC_32f_I 386
7.5.1.418 nppsSubC_32fc 387
7.5.1.419 nppsSubC_32fc_I 387
7.5.1.420 nppsSubC_32s_ISfs 387
7.5.1.421 nppsSubC_32s_Sfs 388
7.5.1.422 nppsSubC_32sc_ISfs 388
7.5.1.423 nppsSubC_32sc_Sfs 388
7.5.1.424 nppsSubC_64f 389
7.5.1.425 nppsSubC_64f_I 389
7.5.1.426 nppsSubC_64fc 389
7.5.1.427 nppsSubC_64fc_I 390
7.5.1.428 nppsSubC_8u_ISfs 390
7.5.1.429 nppsSubC_8u_Sfs 390
7.5.1.430 nppsSubCRev_16s_ISfs 391
7.5.1.431 nppsSubCRev_16s_Sfs 391
7.5.1.432 nppsSubCRev_16sc_ISfs 391
7.5.1.433 nppsSubCRev_16sc_Sfs 392

7.5.1.434 nppsSubCRev_16u_ISfs 392
7.5.1.435 nppsSubCRev_16u_Sfs 392
7.5.1.436 nppsSubCRev_32f 393
7.5.1.437 nppsSubCRev_32f_I 393
7.5.1.438 nppsSubCRev_32fc 393
7.5.1.439 nppsSubCRev_32fc_I 394
7.5.1.440 nppsSubCRev_32s_ISfs 394
7.5.1.441 nppsSubCRev_32s_Sfs 394
7.5.1.442 nppsSubCRev_32sc_ISfs 395
7.5.1.443 nppsSubCRev_32sc_Sfs 395
7.5.1.444 nppsSubCRev_64f 395
7.5.1.445 nppsSubCRev_64f_I 396
7.5.1.446 nppsSubCRev_64fc 396
7.5.1.447 nppsSubCRev_64fc_I 396
7.5.1.448 nppsSubCRev_8u_ISfs 396
7.5.1.449 nppsSubCRev_8u_Sfs 397
7.5.1.450 nppsSum_16s32s_Sfs 397
7.5.1.451 nppsSum_16s_Sfs 398
7.5.1.452 nppsSum_16sc32sc_Sfs 398
7.5.1.453 nppsSum_16sc_Sfs 398
7.5.1.454 nppsSum_32f 399
7.5.1.455 nppsSum_32fc 399
7.5.1.456 nppsSum_32s_Sfs 399
7.5.1.457 nppsSum_64f 400
7.5.1.458 nppsSum_64fc 400
7.5.1.459 nppsXor_16u 400
7.5.1.460 nppsXor_16u_I 401
7.5.1.461 nppsXor_32u 401
7.5.1.462 nppsXor_32u_I 401
7.5.1.463 nppsXor_8u 401
7.5.1.464 nppsXor_8u_I 402
7.5.1.465 nppsXorC_16u 402
7.5.1.466 nppsXorC_16u_I 402
7.5.1.467 nppsXorC_32u 403
7.5.1.468 nppsXorC_32u_I 403
7.5.1.469 nppsXorC_8u 403

7.5.1.470 nppsXorC_8u_I 404
7.5.1.471 nppsZero_16s 404
7.5.1.472 nppsZero_16sc 404
7.5.1.473 nppsZero_32f 404
7.5.1.474 nppsZero_32fc 405
7.5.1.475 nppsZero_32s 405
7.5.1.476 nppsZero_32sc 405
7.5.1.477 nppsZero_64f 405
7.5.1.478 nppsZero_64fc 406
7.5.1.479 nppsZero_64s 406
7.5.1.480 nppsZero_64sc 406
7.5.1.481 nppsZero_8u 406

8 Data Structure Documentation 407

8.1 Npp16sc Struct Reference 407
8.1.1 Detailed Description 407
8.1.2 Field Documentation 407
8.1.2.1 im 407
8.1.2.2 re 407
8.2 Npp32fc Struct Reference 408
8.2.1 Detailed Description 408
8.2.2 Field Documentation 408
8.2.2.1 im 408
8.2.2.2 re 408
8.3 Npp32sc Struct Reference 409
8.3.1 Detailed Description 409
8.3.2 Field Documentation 409
8.3.2.1 im 409
8.3.2.2 re 409
8.4 Npp64fc Struct Reference 410
8.4.1 Detailed Description 410
8.4.2 Field Documentation 410
8.4.2.1 im 410
8.4.2.2 re 410
8.5 Npp64sc Struct Reference 411
8.5.1 Detailed Description 411
8.5.2 Field Documentation 411

8.5.2.1 im 411
8.5.2.2 re 411
8.6 NppiHaarBuffer Struct Reference 412
8.6.1 Field Documentation 412
8.6.1.1 haarBuffer 412
8.6.1.2 haarBufferSize 412
8.7 NppiHaarClassifier_32f Struct Reference 413
8.7.1 Field Documentation 413
8.7.1.1 classifiers 413
8.7.1.2 classifierSize 413
8.7.1.3 classifierStep 413
8.7.1.4 counterDevice 413
8.7.1.5 numClassifiers 413
8.8 NppiPoint Struct Reference 414
8.8.1 Detailed Description 414
8.8.2 Field Documentation 414
8.8.2.1 x 414
8.8.2.2 y 414
8.9 NppiRect Struct Reference 415
8.9.1 Detailed Description 415
8.9.2 Field Documentation 415
8.9.2.1 height 415
8.9.2.2 width 415
8.9.2.3 x 415
8.9.2.4 y 415
8.10 NppiSize Struct Reference 416
8.10.1 Detailed Description 416
8.10.2 Field Documentation 416
8.10.2.1 height 416
8.10.2.2 width 416
8.11 NppLibraryVersion Struct Reference 417
8.11.1 Field Documentation 417
8.11.1.1 build 417
8.11.1.2 major 417
8.11.1.3 minor 417

Chapter 1

NVIDIA Performance Primitives

1.1 What is NPP?

NVIDIA NPP is a library of functions for performing CUDA accelerated processing. The initial set of functionality in the library focuses on imaging and video processing and is widely applicable for developers in these areas. NPP will evolve over time to encompass more of the compute heavy tasks in a variety of problem domains. The NPP library is written to maximize flexibility, while maintaining high performance.

NPP can be used in one of two ways:

- A stand-alone library for adding GPU acceleration to an application with minimal effort. Using this route allows developers to add GPU acceleration to their applications in a matter of hours.
- A cooperative library for interoperating with a developer's GPU code efficiently.

Either route allows developers to harness the massive compute resources of NVIDIA GPUs, while simultaneously reducing development times.

1.2 Documentation

- General API Conventions
- Signal-Processing Specific API Conventions
- Imaging-Processing Specific API Conventions

1.3 Technical Specifications

Supported Platforms:

- Microsoft Windows 7 (64-bit and 32-bit)
- Microsoft Windows Vista (64-bit and 32-bit)
- Microsoft Windows XP (64-bit and 32-bit)
- Linux (Centos & Ubuntu) (64-bit and 32-bit)
- Mac OS X

1.4 Files

NPP is comprises the following files:

1.4.1 Header Files

- npp.h
- nppcore.h
- nppdefs.h
- nppi.h
- npps.h
- nppversion.h

All those header files are located in the CUDA Toolkit's

```
/include/
```

directory.

1.4.2 Library Files

On the Windows platform the NPP stub library is found in the CUDA Toolkit's library directory:

```
/lib/npp.lib
```

The matching DLL is located in the CUDA Toolkit's binary directory:

```
/bin/npp32_32_7.dll      // Dynamic library for 32-bit Windows.
/bin/npp64_32_7.dll      // Dynamic library for 64-bit Windows.
```

On Linux and Mac platforms the dynamic libraries are located in the lib directory

```
/lib/libnpp32.so.3.2.9   // NPP 32-bit dynamic library for Linux
/lib/libnpp64.so.3.2.9   // NPP 64-bit dynamic library for Linux

/lib/libnpp32.3.2.dylib  // NPP 32-bit dynamic library for Mac
/lib/libnpp64.3.2.dylib  // NPP 64-bit dynamic library for Mac
```

1.5 Supported NVIDIA Hardware

NPP runs on all CUDA capable NVIDIA hardware. For details please see http://www.nvidia.com/object/cuda_learn_products.html

Chapter 2

General API Conventions

2.1 Memory Management

The design of all the NPP functions follows the same guidelines as other NVIDIA CUDA libraries like cuFFT and cuBLAS. That is that all pointer arguments in those APIs are device pointers.

This convention enables the individual developer to make smart choices about memory management that minimize the number of memory transfers. It also allows the user the maximum flexibility regarding which of the various memory transfer mechanisms offered by the CUDA runtime is used, e.g. synchronous or asynchronous memory transfers, zero-copy and pinned memory, etc.

The most basic steps involved in using NPP for processing data is as follows:

1. Transfer input data from the host to device using

   ```
   cudaMemCpy(...)
   ```

2. Process data using one or several NPP functions or custom CUDA kernels
3. Transfer the result data from the device to the host using

   ```
   cudaMemCpy(...)
   ```

Throughout NPP there are a number of functions that require the use of host pointers. E.g. various <Primitiv>GetBufferSize(...) functions. Those functions compute the minimum size of (scratch memory) buffer that some primitives require. This buffer size is returned via a host pointer. Since these buffers are allocated via CUDA runtime functions, it would make no sense to place those size values in device memory by default.

2.2 Function Naming

Since NPP is a C API and therefore does not allow for function overloading for different data-types the NPP naming convention addresses the need to differentiate between different flavors of the same algorithm or primitive function but for various data types. This disambiguation of different flavors of a primitive is done via a suffix containing data type and other disambiguating information.

In addition to the flavor suffix, all NPP functions are prefixed with by the letters "npp". Primitives belonging to NPP's image-processing module add the letter "i" to the npp prefix, i.e. are prefixed by "nppi". Similarly signal-processing primitives are prefixed with "npps".

The general naming scheme is:

npp<module info><PrimitiveName>_<data-type info>[_<additional flavor info>](<parameter list>)

The data-type information uses the same names as the Basic NPP Data Types. For example the data-type information "8u" would imply that the primitive operates on Npp8u data.

If a primitive consumes different type data from what it produces, both types will be listed in the order of consumed to produced data type.

Details about the "additional flavor information" is provided for each of the NPP modules, since each problem domain uses different flavor information suffixes.

2.3 Integer Result Scaling

NPP signal processing and imaging primitives often operate on integer data. This integer data is a usually a fixed point fractional representation of some physical magnitue (e.g. luminance). Because of this fixed-point nature of the representation many numerical operations (e.g. addition or multiplication) tend produce results exceeding the original fixed-point range if treated as regular integers.

In cases where the results exceed the original range, these functions clamp the result values back to the valid range. E.g. the maximum positive value for a 16-bit unsigned integer is 32767. A multiplication operation of $4 * 10000 = 40000$ would exceed this range. The result would be clamped to be 32767.

To avoid the level of lost information due to clamping most integer primitives allow for result scaling. Primitives with result scaling have the "Sfs" suffix in their name and provide a parameter "nScaleFactor" that controls the amount of scaling. Before the results of an operation are clamped to the valid output-data range by multiplying them with $2^{\text{-nScaleFactor}}$.

Example: The primitive nppsSqr_8u_Sfs() computes the square of 8-bit unsigned sample values in a signal (1D array of values). The maximum value of a 8-bit value us 255. The square of $255^2 = 65025$ which would be clamped to 255 if no result scaling is performed. In order to map the maximum value of 255 to 255 in the result, one would specify an integer result scaling factor of 8, i.e. multiply each result with $2^{-8} = \frac{1}{2^8} = \frac{1}{256}$. The final result for a signal value of 255 being squared and scaled would be:

$$255^2 \cdot 2^{-8} = 254.00390625$$

which would be rounded to a final result of 254.

A medium gray value of 128 would result in

$$128^2 * 2^{-8} = 64$$

Chapter 3

Signal-Processing Specific API Conventions

3.1 Signal Data

Signal data is passed to and from NPPS primitives via a pointer to the signal's data type.

The general idea behind this fairly low-level way of passing signal data is ease-of-adoption into existing software projects:

- Passing the data pointer rather than a higher- level signal struct allows for easy adoption by not requiring a specific signal representation (that could include total signal size offset, or other additional information). This avoids awkward packing and unpacking of signal data from the host application to an NPP specific signal representation.

3.1.1 Parameter Names for Signal Data

There are three general cases of image-data passing throughout NPP detailed in the following sections.

Those are signals consumed by the algorithm.

3.1.1.1 Source Signal Pointer

The source signal data is generally passed via a pointer named

```
pSrc
```

The source signal pointer is generally defined constant, enforcing that the primitive does not change any image data pointed to by that pointer. E.g.

```
nppsPrimitive_32s(const Npp32s * pSrc, ...)
```

In case the primitive consumes multiple signals as inputs the source pointers are numbered like this:

```
pSrc1, pScr2, ...
```

3.1.1.2 Destination Signal Pointer

The destination signal data is generally passed via a pointer named

```
pDst
```

In case the primitive consumes multiple signals as inputs the source pointers are numbered like this:

```
pDst1, pDst2, ...
```

3.1.1.3 In-Place Signal Pointer

In the case of in-place processing, source and destination are served by the same pointer and thus pointers to in-place signal data are called:

```
pSrcDst
```

3.1.2 Signal Data Alignment Requirements

NPP requires signal sample data to be naturally aligned, i.e. any pointer

```
NppType * p;
```

to a sample in a signal needs to fulfill:

```
assert(p % sizeof(p) == 0);
```

3.1.3 Signal Data Related Error Codes

All NPPI primitives operating on signal data validate the signal-data pointer for proper alignment and test that the point is not null.

Failed validation results in one of the following error codes being returned and the primitive not being executed:

- NPP_NULL_POINTER_ERROR is returned if the image-data pointer is 0 (NULL).
- NPP_ALIGNMENT_ERROR if the signal-data pointer address is not a multiple of the signal's data-type size.

3.2 Signal Length

The vast majority of NPPS functions take a

```
nLength
```

parameter that tells the primitive how many of the signal's samples starting from the given data pointer are to be processed.

3.2.1 Length Related Error Codes

All NPPS primitives taking a length parameter validate this input.

Failed validation results in the following error code being returned and the primitive not being executed:

- NPP_SIZE_ERROR is returned if the length is negative.

Chapter 4

Imaging-Processing Specific API Conventions

4.1 Function Naming

Image processing related functions use a number of suffixes to indicate various different flavors of a primitive beyond just different data types. The flavor suffix uses the following abbreviations:

- "A" if the image is a 4 channel image this indicates the result alpha channel is not affected by the primitive.
- "Cn" the image consists of n channel packed pixels, where n can be 1, 2, 3 or 4.
- "Pn" the image consists of n separate image planes, where n can be 1, 2, 3 or 4.
- "C" (following the channel information) indicates that the primitive only operates on one of the color channels, the "channel-of-interest". All other output channels are not affected by the primitive.
- "I" indicates that the primitive works "in-place". In this case the image-data pointer is usually named "pSrcDst" to indicate that the image data serves as source and destination at the same time.
- "M" indicates "masked operation". These types of primitives have an additional "mask image" as as input. Each pixel in the destination image corresponds to a pixel in the mask image. Only pixels with a corresponding non-zero mask pixel are being processed.
- "R" indicates the primitive operates only on a rectangular "region-of-interest" or "ROI". All ROI primitives take an additional input parameter of type NppiSize, which specifies the width and height of the rectangular region that the primitive should process. For details on how primitives operate on ROIs see: Region-of-Interest (ROI).
- "Sfs" indicates the result values are processed by fixed scaling and saturation before they're written out.

The suffixes above always appear in alphabetical order. E.g. a 4 channel primitive not affecting the alpha channel with masked operation, in place and with scaling/saturation and ROI would have the postfix: "AC4IMRSfs".

4.2 Image Data

Image data is passed to and from NPPI primitives via a pair of parameters:

1. A pointer to the image's underlying data type.
2. A line step in bytes (also sometimes called line stride).

The general idea behind this fairly low-level way of passing image data is ease-of-adoption into existing software projects:

- Passing a raw pointer to the underlying pixel data type, rather than structured (by color) channel pixel data allows usage of the function in a wide variety of situations avoiding risky type cast or expensive image data copies.
- Passing the data pointer and line step individually rather than a higher- level image struct again allows for easy adoption by not requiring a specific image representation and thus avoiding awkward packing and unpacking of image data from the host application to an NPP specific image representation.

4.2.1 Line Step

The line step (also called "line stride" or "row step") allows lines of oddly sized images to start on well-aligned addresses by adding a number of unused bytes at the ends of the lines. This type of line padding has been common practice in digital image processing for a long time and is not particular to GPU image processing.

The line step is the number of bytes in a line **including the padding.** An other way to interpret this number is to say that it is the number of bytes between the first pixel of successive rows in the image, or generally the number of bytes between two neighboring pixels in any column of pixels.

The general reason for the existence of the line step it is that uniformly aligned rows of pixel enable optimizations of memory-access patterns.

Even though all functions in NPP will work with arbitrarily aligned images, best performance can only be achieved with well aligned image data. Any image data allocated with the NPP image allocators or the 2D memory allocators in the CUDA runtime, is well aligned.

Particularly on older CUDA capable GPUs it is likely that the performance decrease for misaligned data is substantial (orders of magnitude).

All image data passed to NPPI primitives requires a line step to be provided. It is important to keep in mind that this line step is always specified in terms of bytes, not pixels.

4.2.2 Parameter Names for Image Data

There are three general cases of image-data passing throughout NPP detailed in the following sections.

4.2.2.1 Passing Source-Image Data

Those are images consumed by the algorithm.

4.2.2.1.1 Source-Image Pointer

The source image data is generally passed via a pointer named

```
pSrc
```

The source image pointer is generally defined constant, enforcing that the primitive does not change any image data pointed to by that pointer. E.g.

```
nppiPrimitive_32s_C1R(const Npp32s * pSrc, ...)
```

In case the primitive consumes multiple images as inputs the source pointers are numbered like this:

```
pSrc1, pScr2, ...
```

4.2.2.1.2 Source-Image Line Step

The source-image line step is the number of bytes between successive rows in the image. The source-image line step parameter is

```
nSrcStep
```

or in the case of multiple source images

```
nSrcStep1, nSrcStep2, ...
```

4.2.2.2 Passing Destination-Image Data

Those are images produced by the algorithm.

4.2.2.2.1 Destination-Image Pointer

The destination image data is generally passed via a pointer named

```
pDst
```

In case the primitive consumes multiple images as inputs the source pointers are numbered like this:

```
pDst1, pDst2, ...
```

4.2.2.2.2 Destination-Image Line Step

The destination-image line step parameter is

```
nDstStep
```

or in the case of multiple destination images

```
nDstStep1, nDstStep2, ...
```

4.2.2.3 Passing In-Place Image Data

4.2.2.3.1 In-Place Image Pointer

In the case of in-place processing, source and destination are served by the same pointer and thus pointers to in-place image data are called:

```
pSrcDst
```

4.2.2.3.2 In-Place Line Step

The in-place nSrcDstStep

4.2.3 Image Data Alignment Requirements

NPP requires pixel data to adhere to certain alignment constraints: For 2 and 4 channel images the following alignment requirement holds: data_pointer % (#channels $*$ sizeof(channel type)) == 0. E.g. a 4 channel image with underlying type Npp8u (8-bit unsigned) would require all pixels to fall on addresses that are multiples of 4 (4 channels $*$ 1 byte size).

As a logical consequence of all pixels being aligned to their natural size the image line steps of 2 and 4 channel images also need to be multiples of the pixel size.

1 and 3 channel images only require that pixel pointers are aligned to the underlying data type, i.e. pData % sizof(data type) == 0. And consequentially line steps are also held to this requirement.

4.2.4 Image Data Related Error Codes

All NPPI primitives operating on image data validate the image-data pointer for proper alignment and test that the point is not null. They also validate the line stride for proper alignment and guard against the step being less or equal to 0. Failed validation results in one of the following error codes being returnd and the primitive not being executed:

- NPP_STEP_ERROR is returned if the data step is 0 or negative.
- NPP_NOT_EVEN_STEP_ERROR is returned if the line step is not a multiple of the pixel size for 2 and 4 channel images.
- NPP_NULL_POINTER_ERROR is returned if the image-data pointer is 0 (NULL).
- NPP_ALIGNMENT_ERROR if the image-data pointer address is not a multiple of the pixel size for 2 and 4 channel images.

4.3 Region-of-Interest (ROI)

In practice processing a rectangular sub-region of an image is often more common than processing complete images. The vast majority of NPP's image-processing primitives allow for processing of such sub regions also referred to as regions-of-interest or ROIs.

All primitives supporting ROI processing are marked by a "R" in their name suffix. Where possible, the ROI a primitive operates on is passed as a single NppiSize struct, which provides the with and height of the ROI. This raises the obvious question how the primitive knows where in the image this rectangle of (width, height) is located. The "start pixel" of the ROI is implicitly given by the image-data pointer. I.e. instead of explicitly passing a pixel coordinate for the upper-right corner of the ROI the primive's user needs to perform the necessary offset computation on the image data pointers, such that the pointers passed to the primitive thus point to the start of the ROI.

In practice this means that for an image (pSrc, nSrcStep) and the start-pixel of the ROI being given by (xROI, yROI), one would pass

pSrcOffset = pSrc + yROI $*$ nSrcStep + xROI $*$ PixelSize;

as the image-data source to the primitive. PixelSize is typically computed as

PixelSize = NumberOfColorChannels $*$ sizeof(PixelDataType).

E.g. for a pimitive like nppiSet_16s_C4R() we would have

- NumberOfColorChannels == 4;
- sizeof(Npp16s) == 2;
- and thus PixelSize = 4 $*$ 2 = 8;

4.3.1 ROI Related Error Codes

All NPPI primitives operating on ROIs of image data validate the ROI size and image's step size. Failed validation results in one of the following error codes being returned and the primitive not being executed:

- NPP_SIZE_ERROR is returned if either the ROI width or ROI height are negative.
- NPP_STEP_ERROR is returned if the ROI width exceeds the image's line step. In mathematical terms (widthROI $*$ PixelSize) $>$ nLinStep indicates an error.

Chapter 5

Module Index

5.1 Modules

Here is a list of all modules:

NPP Core 21
NPP Type Definitions and Constants 24
    Basic NPP Data Types 32
NPP Image Processing 34
NPP Signal Processing 221

Chapter 6

Data Structure Index

6.1 Data Structures

Here are the data structures with brief descriptions:

Npp16sc (Complex Number This struct represents a short complex number) 407
Npp32fc (Complex Number This struct represents a single floating-point complex number) . . . 408
Npp32sc (Complex Number This struct represents a signed int complex number) 409
Npp64fc (Complex Number This struct represents a double floating-point complex number) . . 410
Npp64sc (Complex Number This struct represents a long long complex number) 411
NppiHaarBuffer 412
NppiHaarClassifier_32f 413
NppiPoint (2D Point) 414
NppiRect (2D Rectangle This struct contains position and size information of a rectangle in two space) 415
NppiSize (2D Size This struct typically represents the size of a a rectangular region in two space) 416
NppLibraryVersion 417

Chapter 7

Module Documentation

7.1 NPP Core

Basic functions for library management, in particular library version and device property query functions.

Functions

- const NppLibraryVersion ∗ nppGetLibVersion ()

  Get the NPP library version.

- NppGpuComputeCapability nppGetGpuComputeCapability ()

  What CUDA compute model is supported by the default CUDA device?

- int nppGetGpuNumSMs ()

  Get the number of Streaming Multiprocessors (SM) on the default CUDA device.

- int nppGetMaxThreadsPerBlock ()

  Get the maximum number of threads per block on the default CUDA device.

- const char ∗ nppGetGpuName ()

  Get the name of the default CUDA device.

- cudaStream_t nppGetStream ()

  Get the NPP CUDA stream.

- void nppSetStream (cudaStream_t hStream)

  Set the NPP CUDA stream.

7.1.1 Detailed Description

Basic functions for library management, in particular library version and device property query functions.

7.1.2 Function Documentation

7.1.2.1 NppGpuComputeCapability nppGetGpuComputeCapability ()

What CUDA compute model is supported by the default CUDA device?

Before trying to call any NPP functions, the user should make a call this function to ensure that the current machine has a CUDA capable device.

Returns:

An enum value representing if a CUDA capable device was found and what level of compute capabilities it supports.

7.1.2.2 const char∗ nppGetGpuName ()

Get the name of the default CUDA device.

Returns:

Name string of the graphics-card/compute device in a system.

7.1.2.3 int nppGetGpuNumSMs ()

Get the number of Streaming Multiprocessors (SM) on the default CUDA device.

Returns:

Number of SMs of the default CUDA device.

7.1.2.4 const NppLibraryVersion∗ nppGetLibVersion ()

Get the NPP library version.

Returns:

A struct containing separate values for major and minor revision and build number.

7.1.2.5 int nppGetMaxThreadsPerBlock ()

Get the maximum number of threads per block on the default CUDA device.

Returns:

Maximum number of threads per block on the default CUDA device.

7.1.2.6 cudaStream_t nppGetStream ()

Get the NPP CUDA stream.

NPP enables concurrent device tasks via a global stream state varible. The NPP stream by default is set to stream 0, i.e. non-concurrent mode. A user can set the NPP stream to any valid CUDA stream. All CUDA commands issued by NPP (e.g. kernels launched by the NPP library) are then issed to that NPP stream.

7.1.2.7 void nppSetStream (cudaStream_t *hStream*)

Set the NPP CUDA stream.

See also:

nppGetStream()

7.2 NPP Type Definitions and Constants

Data Structures

- struct NppLibraryVersion
- struct NppiPoint

  2D Point

- struct NppiSize

  2D Size This struct typically represents the size of a a rectangular region in two space.

- struct NppiRect

  2D Rectangle This struct contains position and size information of a rectangle in two space.

- struct NppiHaarClassifier_32f
- struct NppiHaarBuffer

Modules

- Basic NPP Data Types

Defines

- #define NPP_MIN_8U (0)

  Minimum 8-bit unsigned integer.

- #define NPP_MAX_8U (255)

  Maximum 8-bit unsigned integer.

- #define NPP_MIN_16U (0)

  Minimum 16-bit unsigned integer.

- #define NPP_MAX_16U (65535)

  Maximum 16-bit unsigned integer.

- #define NPP_MIN_32U (0)

  Minimum 32-bit unsigned integer.

- #define NPP_MAX_32U (4294967295)

  Maximum 32-bit unsigned integer.

- #define NPP_MIN_8S (-128)

  Minimum 8-bit signed integer.

- #define NPP_MAX_8S (127)

  Maximum 8-bit signed integer.

- #define NPP_MIN_16S (-32768)

  Minimum 16-bit signed integer.

- #define NPP_MAX_16S (32767)

  Maximum 16-bit signed integer.

- #define NPP_MIN_32S (-2147483648)

  Minimum 32-bit signed integer.

- #define NPP_MAX_32S (2147483647)

  Maximum 32-bit signed integer.

- #define NPP_MAX_64S (9223372036854775807LL)

  Minimum 64-bit signed integer.

- #define NPP_MIN_64S (-9223372036854775808LL)

  Maximum 64-bit signed integer.

- #define NPP_MINABS_32F (1.175494351e-38f)

  Smallest positive 32-bit floating point value.

- #define NPP_MAXABS_32F (3.402823466e+38f)

  Largest positive 32-bit floating point value.

- #define NPP_MINABS_64F (2.2250738585072014e-308)

  Smallest positive 64-bit floating point value.

- #define NPP_MAXABS_64F (1.7976931348623158e+308)

  Largest positive 64-bit floating point value.

Enumerations

- enum NppiInterpolationMode {
  NPPI_INTER_UNDEFINED = 0,
  NPPI_INTER_NN = 1,
  NPPI_INTER_LINEAR = 2,
  NPPI_INTER_CUBIC = 4,
  NPPI_INTER_SUPER = 8,
  NPPI_INTER_LANCZOS = 16,
  NPPI_SMOOTH_EDGE = (1 << 31) }

  Filtering methods.

- enum NppStatus {
  NPP_NOT_SUPPORTED_MODE_ERROR = -9999,
  NPP_ROUND_MODE_NOT_SUPPORTED_ERROR = -213,
  NPP_RESIZE_NO_OPERATION_ERROR = -50,
  NPP_NOT_SUFFICIENT_COMPUTE_CAPABILITY = -27,
  NPP_BAD_ARG_ERROR = -26,

NPP_LUT_NUMBER_OF_LEVELS_ERROR = -25,
NPP_TEXTURE_BIND_ERROR = -24,
NPP_COEFF_ERROR = -23,
NPP_RECT_ERROR = -22,
NPP_QUAD_ERROR = -21,
NPP_WRONG_INTERSECTION_ROI_ERROR = -20,
NPP_NOT_EVEN_STEP_ERROR = -19,
NPP_INTERPOLATION_ERROR = -18,
NPP_RESIZE_FACTOR_ERROR = -17,
NPP_HAAR_CLASSIFIER_PIXEL_MATCH_ERROR = -16,
NPP_MEMFREE_ERR = -15,
NPP_MEMSET_ERR = -14,
NPP_MEMCPY_ERROR = -13,
NPP_MEM_ALLOC_ERR = -12,
NPP_HISTO_NUMBER_OF_LEVELS_ERROR = -11,
NPP_MIRROR_FLIP_ERR = -10,
NPP_INVALID_INPUT = -9,
NPP_ALIGNMENT_ERROR = -8,
NPP_STEP_ERROR = -7,
NPP_SIZE_ERROR = -6,
NPP_POINTER_ERROR = -5,
NPP_NULL_POINTER_ERROR = -4,
NPP_CUDA_KERNEL_EXECUTION_ERROR = -3,
NPP_NOT_IMPLEMENTED_ERROR = -2,
NPP_ERROR = -1,
NPP_NO_ERROR = 0,
NPP_SUCCESS = NPP_NO_ERROR,
NPP_WARNING = 1,
NPP_WRONG_INTERSECTION_QUAD_WARNING = 2,
NPP_MISALIGNED_DST_ROI_WARNING = 3,
NPP_AFFINE_QUAD_INCORRECT_WARNING = 4,
NPP_DOUBLE_SIZE_WARNING = 5,
NPP_ODD_ROI_WARNING = 6 }

Error Status Codes.

- enum NppGpuComputeCapability {
NPP_CUDA_UNKNOWN_VERSION = -1,
NPP_CUDA_NOT_CAPABLE,
NPP_CUDA_1_0,
NPP_CUDA_1_1,
NPP_CUDA_1_2,
NPP_CUDA_1_3,
NPP_CUDA_2_0 }

- enum NppiAxis {
  NPP_HORIZONTAL_AXIS,
  NPP_VERTICAL_AXIS,
  NPP_BOTH_AXIS }
- enum NppCmpOp {
  NPP_CMP_LESS,
  NPP_CMP_LESS_EQ,
  NPP_CMP_EQ,
  NPP_CMP_GREATER_EQ,
  NPP_CMP_GREATER }
- enum NppRoundMode {
  NPP_RND_ZERO,
  NPP_RND_NEAR,
  NPP_RND_FINANCIAL }
- enum NppHintAlgorithm {
  nppAlgHintNone,
  nppAlgHintFast,
  nppAlgHintAccurate }

7.2.1 Define Documentation

7.2.1.1 #define NPP_MAX_16S (32767)

Maximum 16-bit signed integer.

7.2.1.2 #define NPP_MAX_16U (65535)

Maximum 16-bit unsigned integer.

7.2.1.3 #define NPP_MAX_32S (2147483647)

Maximum 32-bit signed integer.

7.2.1.4 #define NPP_MAX_32U (4294967295)

Maximum 32-bit unsigned integer.

7.2.1.5 #define NPP_MAX_64S (9223372036854775807LL)

Minimum 64-bit signed integer.

7.2.1.6 #define NPP_MAX_8S (127)

Maximum 8-bit signed integer.

7.2.1.7 #define NPP_MAX_8U (255)

Maximum 8-bit unsigned integer.

7.2.1.8 #define NPP_MAXABS_32F (3.402823466e+38f)

Largest positive 32-bit floating point value.

7.2.1.9 #define NPP_MAXABS_64F (1.7976931348623158e+308)

Largest positive 64-bit floating point value.

7.2.1.10 #define NPP_MIN_16S (-32768)

Minimum 16-bit signed integer.

7.2.1.11 #define NPP_MIN_16U (0)

Minimum 16-bit unsigned integer.

7.2.1.12 #define NPP_MIN_32S (-2147483648)

Minimum 32-bit signed integer.

7.2.1.13 #define NPP_MIN_32U (0)

Minimum 32-bit unsigned integer.

7.2.1.14 #define NPP_MIN_64S (-9223372036854775808LL)

Maximum 64-bit signed integer.

7.2.1.15 #define NPP_MIN_8S (-128)

Minimum 8-bit signed integer.

7.2.1.16 #define NPP_MIN_8U (0)

Minimum 8-bit unsigned integer.

7.2.1.17 #define NPP_MINABS_32F (1.175494351e-38f)

Smallest positive 32-bit floating point value.

7.2.1.18 #define NPP_MINABS_64F (2.2250738585072014e-308)

Smallest positive 64-bit floating point value.

7.2.2 Enumeration Type Documentation

7.2.2.1 enum NppCmpOp

Enumerator:

NPP_CMP_LESS

NPP_CMP_LESS_EQ

NPP_CMP_EQ

NPP_CMP_GREATER_EQ

NPP_CMP_GREATER

7.2.2.2 enum NppGpuComputeCapability

Enumerator:

NPP_CUDA_UNKNOWN_VERSION	Indicates that the compute-capability query failed.
NPP_CUDA_NOT_CAPABLE	Indicates that no CUDA capable device was found on machine.
NPP_CUDA_1_0	Indicates that CUDA 1.0 capable device is default device on machine.
NPP_CUDA_1_1	Indicates that CUDA 1.1 capable device.
NPP_CUDA_1_2	Indicates that CUDA 1.2 capable device.
NPP_CUDA_1_3	Indicates that CUDA 1.3 capable device.
NPP_CUDA_2_0	Indicates that CUDA 2.0 or better is default device on machine.

7.2.2.3 enum NppHintAlgorithm

Enumerator:

nppAlgHintNone

nppAlgHintFast

nppAlgHintAccurate

7.2.2.4 enum NppiAxis

Enumerator:

NPP_HORIZONTAL_AXIS

NPP_VERTICAL_AXIS

NPP_BOTH_AXIS

7.2.2.5 enum NppiInterpolationMode

Filtering methods.

Enumerator:

NPPI_INTER_UNDEFINED	
NPPI_INTER_NN	Nearest neighbor filtering.
NPPI_INTER_LINEAR	Linear interpolation.
NPPI_INTER_CUBIC	Cubic interpolation.
NPPI_INTER_SUPER	Super sampling.
NPPI_INTER_LANCZOS	Lanczos filtering.
NPPI_SMOOTH_EDGE	Smooth edge filtering.

7.2.2.6 enum NppRoundMode

Enumerator:

NPP_RND_ZERO

NPP_RND_NEAR

NPP_RND_FINANCIAL

7.2.2.7 enum NppStatus

Error Status Codes.

Almost all NPP function return error-status information using these return codes. Negative return codes indicate errors, positive return codes indicate warnings, a return code of 0 indicates success.

Enumerator:

NPP_NOT_SUPPORTED_MODE_ERROR

NPP_ROUND_MODE_NOT_SUPPORTED_ERROR

NPP_RESIZE_NO_OPERATION_ERROR

NPP_NOT_SUFFICIENT_COMPUTE_CAPABILITY

NPP_BAD_ARG_ERROR

NPP_LUT_NUMBER_OF_LEVELS_ERROR

NPP_TEXTURE_BIND_ERROR

NPP_COEFF_ERROR

NPP_RECT_ERROR

NPP_QUAD_ERROR

NPP_WRONG_INTERSECTION_ROI_ERROR

NPP_NOT_EVEN_STEP_ERROR

NPP_INTERPOLATION_ERROR

NPP_RESIZE_FACTOR_ERROR

NPP_HAAR_CLASSIFIER_PIXEL_MATCH_ERROR

NPP_MEMFREE_ERR

NPP_MEMSET_ERR

NPP_MEMCPY_ERROR

NPP_MEM_ALLOC_ERR

NPP_HISTO_NUMBER_OF_LEVELS_ERROR

NPP_MIRROR_FLIP_ERR

NPP_INVALID_INPUT

NPP_ALIGNMENT_ERROR

NPP_STEP_ERROR Step is less or equal zero.

NPP_SIZE_ERROR

NPP_POINTER_ERROR

NPP_NULL_POINTER_ERROR

NPP_CUDA_KERNEL_EXECUTION_ERROR

NPP_NOT_IMPLEMENTED_ERROR

NPP_ERROR

NPP_NO_ERROR Error free operation.

NPP_SUCCESS Successful operation (same as NPP_NO_ERROR).

NPP_WARNING

NPP_WRONG_INTERSECTION_QUAD_WARNING

NPP_MISALIGNED_DST_ROI_WARNING Speed reduction due to uncoalesced memory accesses warning.

NPP_AFFINE_QUAD_INCORRECT_WARNING Indicates that the quadrangle passed to one of affine warping functions doesn't have necessary properties. First 3 vertices are used, the fourth vertex discarded.

NPP_DOUBLE_SIZE_WARNING Indicates that in case of 422/411/420 sampling the ROI width/height was modified for proper processing.

NPP_ODD_ROI_WARNING Indicates that for 422/411/420 sampling the ROI width/height was forced to even value.

7.3 Basic NPP Data Types

Data Structures

- struct Npp16sc

  Complex Number This struct represents a short complex number.

- struct Npp32sc

  Complex Number This struct represents a signed int complex number.

- struct Npp32fc

  Complex Number This struct represents a single floating-point complex number.

- struct Npp64sc

  Complex Number This struct represents a long long complex number.

- struct Npp64fc

  Complex Number This struct represents a double floating-point complex number.

Typedefs

- typedef unsigned char Npp8u

  8-bit unsigned chars

- typedef signed char Npp8s

  8-bit signed chars

- typedef unsigned short Npp16u

  16-bit unsigned integers

- typedef short Npp16s

  16-bit signed integers

- typedef unsigned int Npp32u

  32-bit unsigned integers

- typedef int Npp32s

  32-bit signed integers

- typedef unsigned long long Npp64u

  64-bit unsigned integers

- typedef long long Npp64s

  64-bit signed integers

- typedef float Npp32f

  32-bit (IEEE) floating-point numbers

- typedef double Npp64f
  64-bit floating-point numbers

7.3.1 Typedef Documentation

7.3.1.1 typedef short Npp16s

16-bit signed integers

7.3.1.2 typedef unsigned short Npp16u

16-bit unsigned integers

7.3.1.3 typedef float Npp32f

32-bit (IEEE) floating-point numbers

7.3.1.4 typedef int Npp32s

32-bit signed integers

7.3.1.5 typedef unsigned int Npp32u

32-bit unsigned integers

7.3.1.6 typedef double Npp64f

64-bit floating-point numbers

7.3.1.7 typedef long long Npp64s

64-bit signed integers

7.3.1.8 typedef unsigned long long Npp64u

64-bit unsigned integers

7.3.1.9 typedef signed char Npp8s

8-bit signed chars

7.3.1.10 typedef unsigned char Npp8u

8-bit unsigned chars

7.4 NPP Image Processing

Functions

- NppStatus nppiSqrIntegral_8u32s32f_C1R (Npp8u ∗pSrc, int nSrcStep, Npp32s ∗pDst, int nDstStep, Npp32f ∗pSqr, int nSqrStep, NppiSize srcROI, Npp32s val, Npp32f valSqr, Npp32s integralImageNewHeight)

  SqrIntegral Transforms an image to integral and integral of pixel squares representation.

- NppStatus nppiRectStdDev_32s32f_C1R (const Npp32s ∗pSrc, int nSrcStep, const Npp32f ∗pSqr, int nSqrStep, Npp32f ∗pDst, int nDstStep, NppiSize oSizeROI, NppiRect rect)

  RectStdDev Computes the standard deviation of integral images.

Image-Memory Allocation

ImageAllocator methods for 2D arrays of data.

The allocators have width and height parameters to specify the size of the image data being allocated. They return a pointer to the newly created memory and return the numbers of bytes between successive lines.

All allocators return memory with line strides that are beneficial for performance. It is not mandatory to use these allocators. Any valid CUDA device-memory pointers can be used by the NPP primitives and there are no restrictions on line strides.

- Npp8u ∗ nppiMalloc_8u_C1 (int nWidthPixels, int nHeightPixels, int ∗pStepBytes)

  8-bit unsigned image memory allocator.

- Npp8u ∗ nppiMalloc_8u_C2 (int nWidthPixels, int nHeightPixels, int ∗pStepBytes)

  2 channel 8-bit unsigned image memory allocator.

- Npp8u ∗ nppiMalloc_8u_C3 (int nWidthPixels, int nHeightPixels, int ∗pStepBytes)

  3 channel 8-bit unsigned image memory allocator.

- Npp8u ∗ nppiMalloc_8u_C4 (int nWidthPixels, int nHeightPixels, int ∗pStepBytes)

  4 channel 8-bit unsigned image memory allocator.

- Npp16u ∗ nppiMalloc_16u_C1 (int nWidthPixels, int nHeightPixels, int ∗pStepBytes)

  16-bit unsigned image memory allocator.

- Npp16u ∗ nppiMalloc_16u_C2 (int nWidthPixels, int nHeightPixels, int ∗pStepBytes)

  2 channel 16-bit unsigned image memory allocator.

- Npp16u ∗ nppiMalloc_16u_C3 (int nWidthPixels, int nHeightPixels, int ∗pStepBytes)

  3 channel 16-bit unsigned image memory allocator.

- Npp16u ∗ nppiMalloc_16u_C4 (int nWidthPixels, int nHeightPixels, int ∗pStepBytes)

  4 channel 16-bit unsigned image memory allocator.

- Npp16s ∗ nppiMalloc_16s_C1 (int nWidthPixels, int nHeightPixels, int ∗pStepBytes)

  16-bit signed image memory allocator.

- Npp16s ∗ nppiMalloc_16s_C2 (int nWidthPixels, int nHeightPixels, int ∗pStepBytes)

  2 channel 16-bit signed image memory allocator.

- Npp16s ∗ nppiMalloc_16s_C4 (int nWidthPixels, int nHeightPixels, int ∗pStepBytes)

  4 channel 16-bit signed image memory allocator.

- Npp32s ∗ nppiMalloc_32s_C1 (int nWidthPixels, int nHeightPixels, int ∗pStepBytes)

  32-bit signed image memory allocator.

- Npp32s ∗ nppiMalloc_32s_C3 (int nWidthPixels, int nHeightPixels, int ∗pStepBytes)

  3 channel 32-bit signed image memory allocator.

- Npp32s ∗ nppiMalloc_32s_C4 (int nWidthPixels, int nHeightPixels, int ∗pStepBytes)

  4 channel 32-bit signed image memory allocator.

- Npp32f ∗ nppiMalloc_32f_C1 (int nWidthPixels, int nHeightPixels, int ∗pStepBytes)

  32-bit floating point image memory allocator.

- Npp32f ∗ nppiMalloc_32f_C2 (int nWidthPixels, int nHeightPixels, int ∗pStepBytes)

  2 channel 32-bit floating point image memory allocator.

- Npp32f ∗ nppiMalloc_32f_C3 (int nWidthPixels, int nHeightPixels, int ∗pStepBytes)

  3 channel 32-bit floating point image memory allocator.

- Npp32f ∗ nppiMalloc_32f_C4 (int nWidthPixels, int nHeightPixels, int ∗pStepBytes)

  4 channel 32-bit floating point image memory allocator.

- void nppiFree (void ∗pData)

  Free method for any 2D allocated memory.

Image-Memory Set

Set methods for images of various types.

Images are passed to these primitives via a pointer to the image data (first pixel in the ROI) and a step-width, i.e. the number of bytes between successive lines. The size of the area to be set (region-of-interest, ROI) is specified via a NppiSize struct. In addition to the image data and ROI, all methods have a parameter to specify the value being set. In case of single channel images this is a single value, in case of multi-channel, an array of values is passed.

- NppStatus nppiSet_8u_C1R (Npp8u nValue, Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI)

  8-bit unsigned image set.

- NppStatus nppiSet_8u_C1MR (Npp8u nValue, Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI, const Npp8u ∗pMask, int nMaskStep)

  Masked 8-bit unsigned image set.

- NppStatus nppiSet_8u_C4R (const Npp8u aValues[4], Npp8u ∗pDst, int nDstStep, NppiSize oSize-ROI)

4 channel 8-bit unsigned image set.

- NppStatus nppiSet_8u_C4MR (const Npp8u aValues[4], Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI, const Npp8u ∗pMask, int nMaskStep)

  Masked 4 channel 8-bit unsigned image set.

- NppStatus nppiSet_8u_AC4R (const Npp8u aValues[3], Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI)

  4 channel 8-bit unsigned image set method, not affecting Alpha channel.

- NppStatus nppiSet_8u_AC4MR (const Npp8u aValues[3], Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI, const Npp8u ∗pMask, int nMaskStep)

  Masked 4 channel 8-bit unsigned image set method, not affecting Alpha channel.

- NppStatus nppiSet_8u_C4CR (Npp8u nValue, Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI)

  4 channel 8-bit unsigned image set affecting only single channel.

- NppStatus nppiSet_16u_C1R (Npp16u nValue, Npp16u ∗pDst, int nDstStep, NppiSize oSizeROI)

  16-bit unsigned image set.

- NppStatus nppiSet_16u_C1MR (Npp16u nValue, Npp16u ∗pDst, int nDstStep, NppiSize oSizeROI, const Npp8u ∗pMask, int nMaskStep)

  Masked 16-bit unsigned image set.

- NppStatus nppiSet_16u_C2R (const Npp16u aValues[2], Npp16u ∗pDst, int nDstStep, NppiSize oSizeROI)

  2 channel 16-bit unsigned image set.

- NppStatus nppiSet_16u_C4R (const Npp16u aValues[4], Npp16u ∗pDst, int nDstStep, NppiSize oSizeROI)

  4 channel 16-bit unsigned image set.

- NppStatus nppiSet_16u_C4MR (const Npp16u aValues[4], Npp16u ∗pDst, int nDstStep, NppiSize oSizeROI, const Npp8u ∗pMask, int nMaskStep)

  Masked 4 channel 16-bit unsigned image set.

- NppStatus nppiSet_16u_AC4R (const Npp16u aValues[3], Npp16u ∗pDst, int nDstStep, NppiSize oSizeROI)

  4 channel 16-bit unsigned image set method, not affecting Alpha channel.

- NppStatus nppiSet_16u_AC4MR (const Npp16u aValues[3], Npp16u ∗pDst, int nDstStep, NppiSize oSizeROI, const Npp8u ∗pMask, int nMaskStep)

  Masked 4 channel 16-bit unsigned image set method, not affecting Alpha channel.

- NppStatus nppiSet_16u_C4CR (Npp16u nValue, Npp16u ∗pDst, int nDstStep, NppiSize oSizeROI)

  4 channel 16-bit unsigned image set affecting only single channel.

- NppStatus nppiSet_16s_C1R (Npp16s nValue, Npp16s ∗pDst, int nDstStep, NppiSize oSizeROI)

  16-bit image set.

- NppStatus nppiSet_16s_C1MR (Npp16s nValue, Npp16s ∗pDst, int nDstStep, NppiSize oSizeROI, const Npp8u ∗pMask, int nMaskStep)

  Masked 16-bit image set.

- NppStatus nppiSet_16s_C2R (const Npp16s aValues[2], Npp16s ∗pDst, int nDstStep, NppiSize oSizeROI)

  2 channel 16-bit image set.

- NppStatus nppiSet_16s_C4R (const Npp16s aValues[4], Npp16s ∗pDst, int nDstStep, NppiSize oSizeROI)

  4 channel 16-bit image set.

- NppStatus nppiSet_16s_C4MR (const Npp16s aValues[4], Npp16s ∗pDst, int nDstStep, NppiSize oSizeROI, const Npp8u ∗pMask, int nMaskStep)

  Masked 4 channel 16-bit image set.

- NppStatus nppiSet_16s_AC4R (const Npp16s aValues[3], Npp16s ∗pDst, int nDstStep, NppiSize oSizeROI)

  4 channel 16-bit image set method, not affecting Alpha channel.

- NppStatus nppiSet_16s_AC4MR (const Npp16s aValues[3], Npp16s ∗pDst, int nDstStep, NppiSize oSizeROI, const Npp8u ∗pMask, int nMaskStep)

  Masked 4 channel 16-bit image set method, not affecting Alpha channel.

- NppStatus nppiSet_16s_C4CR (Npp16s nValue, Npp16s ∗pDst, int nDstStep, NppiSize oSizeROI)

  4 channel 16-bit unsigned image set affecting only single channel.

- NppStatus nppiSet_32s_C1R (Npp32s nValue, Npp32s ∗pDst, int nDstStep, NppiSize oSizeROI)

  32-bit image set.

- NppStatus nppiSet_32s_C1MR (Npp32s nValue, Npp32s ∗pDst, int nDstStep, NppiSize oSizeROI, const Npp8u ∗pMask, int nMaskStep)

  Masked 32-bit image set.

- NppStatus nppiSet_32s_C4R (const Npp32s aValues[4], Npp32s ∗pDst, int nDstStep, NppiSize oSizeROI)

  4 channel 32-bit image set.

- NppStatus nppiSet_32s_C4MR (const Npp32s aValues[4], Npp32s ∗pDst, int nDstStep, NppiSize oSizeROI, const Npp8u ∗pMask, int nMaskStep)

  Masked 4 channel 32-bit image set.

- NppStatus nppiSet_32s_AC4R (const Npp32s aValues[3], Npp32s ∗pDst, int nDstStep, NppiSize oSizeROI)

  4 channel 16-bit image set method, not affecting Alpha channel.

- NppStatus nppiSet_32s_AC4MR (const Npp32s aValues[3], Npp32s ∗pDst, int nDstStep, NppiSize oSizeROI, const Npp8u ∗pMask, int nMaskStep)

  Masked 4 channel 16-bit image set method, not affecting Alpha channel.

- NppStatus nppiSet_32s_C4CR (Npp32s nValue, Npp32s ∗pDst, int nDstStep, NppiSize oSizeROI)

4 channel 32-bit unsigned image set affecting only single channel.

- NppStatus nppiSet_32f_C1R (Npp32f nValue, Npp32f ∗pDst, int nDstStep, NppiSize oSizeROI)

  32-bit floating point image set.

- NppStatus nppiSet_32f_C1MR (Npp32f nValue, Npp32f ∗pDst, int nDstStep, NppiSize oSizeROI, const Npp8u ∗pMask, int nMaskStep)

  Masked 32-bit floating point image set.

- NppStatus nppiSet_32f_C4R (const Npp32f aValues[4], Npp32f ∗pDst, int nDstStep, NppiSize oSizeROI)

  4 channel 32-bit floating point image set.

- NppStatus nppiSet_32f_C4MR (const Npp32f aValues[4], Npp32f ∗pDst, int nDstStep, NppiSize oSizeROI, const Npp8u ∗pMask, int nMaskStep)

  Masked 4 channel 32-bit floating point image set.

- NppStatus nppiSet_32f_AC4R (const Npp32f aValues[3], Npp32f ∗pDst, int nDstStep, NppiSize oSizeROI)

  4 channel 32-bit floating point image set method, not affecting Alpha channel.

- NppStatus nppiSet_32f_AC4MR (const Npp32f aValues[3], Npp32f ∗pDst, int nDstStep, NppiSize oSizeROI, const Npp8u ∗pMask, int nMaskStep)

  Masked 4 channel 32-bit floating point image set method, not affecting Alpha channel.

- NppStatus nppiSet_32f_C4CR (Npp32f nValue, Npp32f ∗pDst, int nDstStep, NppiSize oSizeROI)

  4 channel 32-bit floating point image set affecting only single channel.

Image-Memory Copy

Copy methods for images of various types.

Images are passed to these primitives via a pointer to the image data (first pixel in the ROI) and a step-width, i.e. the number of bytes between successive lines. The size of the area to be copied (region-of-interest, ROI) is specified via a NppiSize struct.

- NppStatus nppiCopy_8u_C1R (const Npp8u ∗pSrc, int nSrcStep, Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI)

  8-bit unsigned image copy.

- NppStatus nppiCopy_8u_C4R (const Npp8u ∗pSrc, int nSrcStep, Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI)

  4 channel 8-bit unsigned image copy.

- NppStatus nppiCopy_8u_AC4R (const Npp8u ∗pSrc, int nSrcStep, Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI)

  4 channel 8-bit unsigned image copy, not affecting Alpha channel.

- NppStatus nppiCopy_16u_C1R (const Npp16u ∗pSrc, int nSrcStep, Npp16u ∗pDst, int nDstStep, NppiSize oSizeROI)

16-bit unsigned image copy.

- NppStatus nppiCopy_16u_C4R (const Npp16u ∗pSrc, int nSrcStep, Npp16u ∗pDst, int nDstStep, NppiSize oSizeROI)

  4 channel 16-bit unsigned image copy.

- NppStatus nppiCopy_16u_AC4R (const Npp16u ∗pSrc, int nSrcStep, Npp16u ∗pDst, int nDstStep, NppiSize oSizeROI)

  4 channel 16-bit unsigned image copy, not affecting Alpha channel.

- NppStatus nppiCopy_16s_C1R (const Npp16s ∗pSrc, int nSrcStep, Npp16s ∗pDst, int nDstStep, NppiSize oSizeROI)

  16-bit image copy.

- NppStatus nppiCopy_16s_C4R (const Npp16s ∗pSrc, int nSrcStep, Npp16s ∗pDst, int nDstStep, NppiSize oSizeROI)

  4 channel 16-bit image copy.

- NppStatus nppiCopy_16s_AC4R (const Npp16s ∗pSrc, int nSrcStep, Npp16s ∗pDst, int nDstStep, NppiSize oSizeROI)

  4 channel 16-bit image copy, not affecting Alpha.

- NppStatus nppiCopy_32s_C1R (const Npp32s ∗pSrc, int nSrcStep, Npp32s ∗pDst, int nDstStep, NppiSize oSizeROI)

  32-bit image copy.

- NppStatus nppiCopy_32s_C4R (const Npp32s ∗pSrc, int nSrcStep, Npp32s ∗pDst, int nDstStep, NppiSize oSizeROI)

  4 channel 32-bit image copy.

- NppStatus nppiCopy_32s_AC4R (const Npp32s ∗pSrc, int nSrcStep, Npp32s ∗pDst, int nDstStep, NppiSize oSizeROI)

  4 channel 32-bit image copy, not affecting Alpha.

- NppStatus nppiCopy_32f_C1R (const Npp32f ∗pSrc, int nSrcStep, Npp32f ∗pDst, int nDstStep, NppiSize oSizeROI)

  32-bit floating point image copy.

- NppStatus nppiCopy_32f_C4R (const Npp32f ∗pSrc, int nSrcStep, Npp32f ∗pDst, int nDstStep, NppiSize oSizeROI)

  4 channel 32-bit floating point image copy.

- NppStatus nppiCopy_32f_AC4R (const Npp32f ∗pSrc, int nSrcStep, Npp32f ∗pDst, int nDstStep, NppiSize oSizeROI)

  4 channel 32-bit floating point image copy, not affecting Alpha.

Bit-Depth Conversion

Convert bit-depth up and down.

The integer conversion methods do not involve any scaling. Conversions that reduce bit-depth saturate values exceeding the reduced range to the range's maximum/minimum value. When converting from floating-point values to integer values, a rounding mode can be specified. After rounding to integer values the values get saturated to the destination data type's range.

- NppStatus nppiConvert_8u16u_C1R (const Npp8u ∗pSrc, int nSrcStep, Npp16u ∗pDst, int nDstStep, NppiSize oSizeROI)

  8-bit unsigned to 16-bit unsigned conversion.

- NppStatus nppiConvert_16u8u_C1R (const Npp16u ∗pSrc, int nSrcStep, Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI)

  16-bit unsigned to 8-bit unsigned conversion.

- NppStatus nppiConvert_8u16u_C4R (const Npp8u ∗pSrc, int nSrcStep, Npp16u ∗pDst, int nDstStep, NppiSize oSizeROI)

  4 channel 8-bit unsigned to 16-bit unsigned conversion.

- NppStatus nppiConvert_16u8u_C4R (const Npp16u ∗pSrc, int nSrcStep, Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI)

  4 channel 16-bit unsigned to 8-bit unsigned conversion.

- NppStatus nppiConvert_8u16u_AC4R (const Npp8u ∗pSrc, int nSrcStep, Npp16u ∗pDst, int nDstStep, NppiSize oSizeROI)

  4 channel 8-bit unsigned to 16-bit unsigned conversion, not affecting Alpha.

- NppStatus nppiConvert_16u8u_AC4R (const Npp16u ∗pSrc, int nSrcStep, Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI)

  4 channel 16-bit unsigned to 8-bit unsigned conversion, not affecting Alpha.

- NppStatus nppiConvert_8u16s_C1R (const Npp8u ∗pSrc, int nSrcStep, Npp16s ∗pDst, int nDstStep, NppiSize oSizeROI)

  8-bit unsigned to 16-bit signed conversion.

- NppStatus nppiConvert_16s8u_C1R (const Npp16s ∗pSrc, int nSrcStep, Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI)

  4 channel 16-bit signed to 8-bit unsigned conversion.

- NppStatus nppiConvert_8u16s_C4R (const Npp8u ∗pSrc, int nSrcStep, Npp16s ∗pDst, int nDstStep, NppiSize oSizeROI)

  4 channel 8-bit unsigned to 16-bit signed conversion.

- NppStatus nppiConvert_16s8u_C4R (const Npp16s ∗pSrc, int nSrcStep, Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI)

  4 channel 16-bit signed to 8-bit unsignedconversion, not affecting Alpha.

- NppStatus nppiConvert_8u16s_AC4R (const Npp8u ∗pSrc, int nSrcStep, Npp16s ∗pDst, int nDstStep, NppiSize oSizeROI)

  4 channel 8-bit unsigned to 16-bit signed conversion, not affecting Alpha.

- NppStatus nppiConvert_16s8u_AC4R (const Npp16s ∗pSrc, int nSrcStep, Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI)

4 channel 16-bit signed to 8-bit unsigned conversion, not affecting Alpha.

- NppStatus nppiConvert_16s32f_C1R (const Npp16s ∗pSrc, int nSrcStep, Npp32f ∗pDst, int nDstStep, NppiSize oSizeROI)

  16-bit singedto 32-bit floating point conversion.

- NppStatus nppiConvert_32f16s_C1R (const Npp32f ∗pSrc, int nSrcStep, Npp16s ∗pDst, int nDstStep, NppiSize oSizeROI, NppRoundMode eRoundMode)

  32-bit floating point to 16-bit conversion.

- NppStatus nppiConvert_8u32f_C1R (const Npp8u ∗pSrc, int nSrcStep, Npp32f ∗pDst, int nDstStep, NppiSize oSizeROI)

  8-bit unsigned to 32-bit floating point conversion.

- NppStatus nppiConvert_16u32f_C1R (const Npp16u ∗pSrc, int nSrcStep, Npp32f ∗pDst, int nDstStep, NppiSize oSizeROI)

  16-bit unsigned to 32-bit floating point conversion.

- NppStatus nppiConvert_32f16u_C1R (const Npp32f ∗pSrc, int nSrcStep, Npp16u ∗pDst, int nDstStep, NppiSize oSizeROI, NppRoundMode eRoundMode)

  32-bit floating point to 16-bit unsigned conversion.

- NppStatus nppiConvert_32f8u_C1R (const Npp32f ∗pSrc, int nSrcStep, Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI, NppRoundMode eRoundMode)

  32-bit floating point to 8-bit unsigned conversion.

- NppStatus nppiConvert_16u32s_C1R (const Npp16u ∗pSrc, int nSrcStep, Npp32s ∗pDst, int nDstStep, NppiSize oSizeROI)

  16-bit unsigned to 32-bit signed conversion.

- NppStatus nppiConvert_16s32s_C1R (const Npp16s ∗pSrc, int nSrcStep, Npp32s ∗pDst, int nDstStep, NppiSize oSizeROI)

  16-bit to 32-bit conversion.

Copy Const Border

Methods for copying images and padding borders with a constant, user-specifiable color.

- NppStatus nppiCopyConstBorder_8u_C1R (const Npp8u ∗pSrc, int nSrcStep, NppiSize oSrcSizeROI, Npp8u ∗pDst, int nDstStep, NppiSize oDstSizeROI, int nTopBorderHeight, int nLeftBorderWidth, Npp8u nValue)

  8-bit unsigned image copy width constant border color.

- NppStatus nppiCopyConstBorder_8u_C4R (const Npp8u ∗pSrc, int nSrcStep, NppiSize oSrcSizeROI, Npp8u ∗pDst, int nDstStep, NppiSize oDstSizeROI, int nTopBorderHeight, int nLeftBorderWidth, const Npp8u aValue[4])

  4channel 8-bit unsigned image copy with constant border color.

- NppStatus nppiCopyConstBorder_8u_AC4R (const Npp8u ∗pSrc, int nSrcStep, NppiSize oSrcSizeROI, Npp8u ∗pDst, int nDstStep, NppiSize oDstSizeROI, int nTopBorderHeight, int nLeftBorderWidth, const Npp8u aValue[3])

  4 channel 8-bit unsigned image copy with constant border color.

- NppStatus nppiCopyConstBorder_32s_C1R (const Npp32s ∗pSrc, int nSrcStep, NppiSize oSrcSizeROI, Npp32s ∗pDst, int nDstStep, NppiSize oDstSizeROI, int nTopBorderHeight, int nLeftBorderWidth, Npp32s nValue)

  32-bit image copy with constant border color.

Image Transpose

Methods for transposing images of various types.

Like matrix transpose, image transpose is a mirror along the image's diagonal (upper-left to lower-right corner).

- NppStatus nppiTranspose_8u_C1R (const Npp8u ∗pSrc, int nSrcStep, Npp8u ∗pDst, int nDstStep, NppiSize oROI)

  8-bit image transpose.

Image Color Channel Swap

Methods for exchanging the color channels of an image.

The methods support arbitrary permutations of the original channels, including replication.

- NppStatus nppiSwapChannels_8u_C4IR (Npp8u ∗pSrcDst, int nSrcDstStep, NppiSize oSizeROI, const int aDstOrder[4])

  4 channel 8-bit unsigned swap channels, in-place.

Arithmetic with Constant Values

Methods performing image arithmetic with the second operand being a constant rather than an image.

- NppStatus nppiAddC_32f_C1R (const Npp32f ∗pSrc, int nSrcStep, Npp32f nValue, Npp32f ∗pDst, int nDstStep, NppiSize oSizeROI)

  32-bit floating point image add constant.

- NppStatus nppiSubC_32f_C1R (const Npp32f ∗pSrc, int nSrcStep, Npp32f nValue, Npp32f ∗pDst, int nDstStep, NppiSize oSizeROI)

  32-bit floating point image subtract constant.

- NppStatus nppiMulC_32f_C1R (const Npp32f ∗pSrc, int nSrcStep, Npp32f nValue, Npp32f ∗pDst, int nDstStep, NppiSize oSizeROI)

  32-bit floating point image multiply constant.

- NppStatus nppiDivC_32f_C1R (const Npp32f ∗pSrc, int nSrcStep, Npp32f nValue, Npp32f ∗pDst, int nDstStep, NppiSize oSizeROI)

  32-bit floating point image divide by constant.

- NppStatus nppiAbsDiffC_32f_C1R (const Npp32f ∗pSrc, int nSrcStep, Npp32f ∗pDst, int nDstStep, NppiSize oSizeROI, Npp32f nValue)

  32-bit floating point image absolute difference from constant.

- NppStatus nppiAddC_32fc_C1R (const Npp32fc ∗pSrc, int nSrcStep, Npp32fc nValue, Npp32fc ∗pDst, int nDstStep, NppiSize oSizeROI)

  32-bit complex floating point image add constant.

- NppStatus nppiSubC_32fc_C1R (const Npp32fc ∗pSrc, int nSrcStep, Npp32fc nValue, Npp32fc ∗pDst, int nDstStep, NppiSize oSizeROI)

  32-bit complex floating point image subtract constant.

- NppStatus nppiMulC_32fc_C1R (const Npp32fc ∗pSrc, int nSrcStep, Npp32fc nValue, Npp32fc ∗pDst, int nDstStep, NppiSize oSizeROI)

  32-bit complex floating point image multiply constant.

- NppStatus nppiDivC_32fc_C1R (const Npp32fc ∗pSrc, int nSrcStep, Npp32fc nValue, Npp32fc ∗pDst, int nDstStep, NppiSize oSizeROI)

  32-bit complex floating point image divide by constant.

Image Addition

Methods for adding two images.

- NppStatus nppiAdd_8u_C1RSfs (const Npp8u ∗pSrc1, int nSrc1Step, const Npp8u ∗pSrc2, int nSrc2Step, Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI, int nScaleFactor)

  8-bit unsigned image add.

- NppStatus nppiAdd_8u_C4RSfs (const Npp8u ∗pSrc1, int nSrc1Step, const Npp8u ∗pSrc2, int nSrc2Step, Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI, int nScaleFactor)

  4 channel 8-bit unsigned image add.

- NppStatus nppiAdd_8u_AC4RSfs (const Npp8u ∗pSrc1, int nSrc1Step, const Npp8u ∗pSrc2, int nSrc2Step, Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI, int nScaleFactor)

  4 channel 8-bit unsigned image add, not affecting Alpha.

- NppStatus nppiAdd_32f_C1R (const Npp32f ∗pSrc1, int nSrc1Step, const Npp32f ∗pSrc2, int nSrc2Step, Npp32f ∗pDst, int nDstStep, NppiSize oSizeROI)

  32-bit floating point image add.

- NppStatus nppiAdd_32s_C1R (const Npp32s ∗pSrc1, int nSrc1Step, const Npp32s ∗pSrc2, int nSrc2Step, Npp32s ∗pDst, int nDstStep, NppiSize oSizeROI)

  32-bit image add.

Image Subtraction

Methods for subtracting one image from another.

- NppStatus nppiSub_8u_C1RSfs (const Npp8u ∗pSrc1, int nSrc1Step, const Npp8u ∗pSrc2, int nSrc2Step, Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI, int nScaleFactor)

  8-bit unsigned image subtraction.

- NppStatus nppiSub_8u_C4RSfs (const Npp8u ∗pSrc1, int nSrc1Step, const Npp8u ∗pSrc2, int nSrc2Step, Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI, int nScaleFactor)

  4 channel 8-bit unsigned image subtraction.

- NppStatus nppiSub_8u_AC4RSfs (const Npp8u ∗pSrc1, int nSrc1Step, const Npp8u ∗pSrc2, int nSrc2Step, Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI, int nScaleFactor)

  4 channel 8-bit unsigned image subtraction, not affecting Alpha.

- NppStatus nppiSub_32f_C1R (const Npp32f ∗pSrc1, int nSrc1Step, const Npp32f ∗pSrc2, int nSrc2Step, Npp32f ∗pDst, int nDstStep, NppiSize oSizeROI)

  32-bit floating point image subtraction.

- NppStatus nppiSub_32s_C1R (const Npp32s ∗pSrc1, int nSrc1Step, const Npp32s ∗pSrc2, int nSrc2Step, Npp32s ∗pDst, int nDstStep, NppiSize oSizeROI)

  32-bit image subtraction.

Image Multiplication

Methods for multiplying two images.

- NppStatus nppiMul_8u_C1RSfs (const Npp8u ∗pSrc1, int nSrc1Step, const Npp8u ∗pSrc2, int nSrc2Step, Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI, int nScaleFactor)

  8-bit unsigned image multiplication.

- NppStatus nppiMul_8u_C4RSfs (const Npp8u ∗pSrc1, int nSrc1Step, const Npp8u ∗pSrc2, int nSrc2Step, Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI, int nScaleFactor)

  4 channel 8-bit unsigned image multiplication.

- NppStatus nppiMul_8u_AC4RSfs (const Npp8u ∗pSrc1, int nSrc1Step, const Npp8u ∗pSrc2, int nSrc2Step, Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI, int nScaleFactor)

  4 channel 8-bit unsigned image multiplication, not affecting Alpha.

- NppStatus nppiMul_32f_C1R (const Npp32f ∗pSrc1, int nSrc1Step, const Npp32f ∗pSrc2, int nSrc2Step, Npp32f ∗pDst, int nDstStep, NppiSize oSizeROI)

  4 channel 32-bit floating point image multiplication.

- NppStatus nppiMul_32s_C1R (const Npp32s ∗pSrc1, int nSrc1Step, const Npp32s ∗pSrc2, int nSrc2Step, Npp32s ∗pDst, int nDstStep, NppiSize oSizeROI)

  4 channel 32-bit image multiplication.

Image Division

Methods for dividing one image by another.

- NppStatus nppiDiv_8u_C1RSfs (const Npp8u ∗pSrc1, int nSrc1Step, const Npp8u ∗pSrc2, int nSrc2Step, Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI, int nScaleFactor)

  8-bit unsignedimage division.

- NppStatus nppiDiv_8u_C4RSfs (const Npp8u ∗pSrc1, int nSrc1Step, const Npp8u ∗pSrc2, int nSrc2Step, Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI, int nScaleFactor)

  4 channel 8-bit unsigned image division.

- NppStatus nppiDiv_8u_AC4RSfs (const Npp8u ∗pSrc1, int nSrc1Step, const Npp8u ∗pSrc2, int nSrc2Step, Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI, int nScaleFactor)

  4 channel 8-bit unsigned image division, not affecting Alpha.

- NppStatus nppiDiv_32f_C1R (const Npp32f ∗pSrc1, int nSrc1Step, const Npp32f ∗pSrc2, int nSrc2Step, Npp32f ∗pDst, int nDstStep, NppiSize oSizeROI)

  32-bit floating point image division.

- NppStatus nppiDiv_32s_C1R (const Npp32s ∗pSrc1, int nSrc1Step, const Npp32s ∗pSrc2, int nSrc2Step, Npp32s ∗pDst, int nDstStep, NppiSize oSizeROI)

  32-bit image division.

Image Absolute Difference Methods

Per-pixel absolute difference methods.

- NppStatus nppiAbsDiff_8u_C1R (const Npp8u ∗pSrc1, int nSrc1Step, const Npp8u ∗pSrc2, int nSrc2Step, Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI)

  8-bit unsigned absolute difference.

- NppStatus nppiAbsDiff_8u_C4R (const Npp8u ∗pSrc1, int nSrc1Step, const Npp8u ∗pSrc2, int nSrc2Step, Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI)

  4 channel 8-bit unsigned absolute difference.

- NppStatus nppiAbsDiff_8u_AC4R (const Npp8u ∗pSrc1, int nSrc1Step, const Npp8u ∗pSrc2, int nSrc2Step, Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI)

  4 channel 8-bit unsigned absolute difference, not affecting Alpha.

- NppStatus nppiAbsDiff_32f_C1R (const Npp32f ∗pSrc1, int nSrc1Step, const Npp32f ∗pSrc2, int nSrc2Step, Npp32f ∗pDst, int nDstStep, NppiSize oSizeROI)

  32-bit floating point absolute difference.

- NppStatus nppiAbsDiff_32s_C1R (const Npp32s ∗pSrc1, int nSrc1Step, const Npp32s ∗pSrc2, int nSrc2Step, Npp32s ∗pDst, int nDstStep, NppiSize oSizeROI)

  32-bit absolute difference.

Other Image Arithmetic

- NppStatus nppiLn_32f_C1R (const Npp32f ∗pSrc, int nSrcStep, Npp32f ∗pDst, int nDstStep, NppiSize oSizeROI)

  32-bit floating point logarithm.

- NppStatus nppiExp_32f_C1R (const Npp32f ∗pSrc, int nSrcStep, Npp32f ∗pDst, int nDstStep, NppiSize oSizeROI)

  32-bit floating point exponentiation.

Image Threshold Methods

Threshold pixels.

- NppStatus nppiThreshold_32f_C1R (const Npp32f ∗pSrc, int nSrcStep, Npp32f ∗pDst, int nDstStep, NppiSize oSizeROI, Npp32f nThreshold, NppCmpOp eComparisonOperation)

  32-bit floating point threshold.

- NppStatus nppiThreshold_8u_AC4R (const Npp8u ∗pSrc, int nSrcStep, Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI, const Npp8u aThresholds[3], NppCmpOp eComparisonOperation)

  4 channel 8-bit unsigned image threshold, not affecting Alpha.

Image Compare Methods

Compare the pixels of two images and create a binary result image.

In case of multi-channel image types, the condition must be fulfilled for all channels, otherwise the comparison is considered false. The "binary" result image is of type 8u_C1. False is represented by 0, true by NPP_MAX_8U.

- NppStatus nppiCompare_8u_C4R (const Npp8u ∗pSrc1, int nSrc1Step, const Npp8u ∗pSrc2, int nSrc2Step, Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI, NppCmpOp eComparisonOperation)

  4 channel 8-bit unsigned image compare.

- NppStatus nppiCompare_8u_AC4R (const Npp8u ∗pSrc1, int nSrc1Step, const Npp8u ∗pSrc2, int nSrc2Step, Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI, NppCmpOp eComparisonOperation)

  4 channel 8-bit unsigned image compare, not affecting Alpha.

- NppStatus nppiCompare_32f_C1R (const Npp32f ∗pSrc1, int nSrc1Step, const Npp32f ∗pSrc2, int nSrc2Step, Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI, NppCmpOp eComparisonOperation)

  32-bit floating point image compare.

Mean_StdDev

Computes the mean and standard deviation of image pixel values

- NppStatus nppiMean_StdDev_8u_C1R (const Npp8u ∗pSrc, int nSrcStep, NppiSize oSizeROI, Npp64f ∗pMean, Npp64f ∗pStdDev)

8-bit unsigned mean standard deviation.

NormDiff

Norm of pixel differences between two images.

- NppStatus nppiNormDiff_L1_8u_C1R (const Npp8u ∗pSrc1, int nSrcStep1, const Npp8u ∗pSrc2, int nSrcStep2, NppiSize oSizeROI, Npp64f ∗pRetVal)

  8-bit unsigned L1 norm of pixel differences.
- NppStatus nppiNormDiff_L2_8u_C1R (const Npp8u ∗pSrc1, int nSrcStep1, const Npp8u ∗pSrc2, int nSrcStep2, NppiSize oSizeROI, Npp64f ∗pRetVal)

  8-bit unsigned L2 norm of pixel differences.
- NppStatus nppiNormDiff_Inf_8u_C1R (const Npp8u ∗pSrc1, int nSrcStep1, const Npp8u ∗pSrc2, int nSrcStep2, NppiSize oSizeROI, Npp64f ∗pRetVal)

  8-bit unsigned Infinity Norm of pixel differences.

1D Linear Filter

1D mask Linear Convolution Filter, with rescaling, for 8 bit images.

- NppStatus nppiFilterColumn_8u_C1R (const Npp8u ∗pSrc, Npp32s nSrcStep, Npp8u ∗pDst, Npp32s nDstStep, NppiSize oROI, const Npp32s ∗pKernel, Npp32s nMaskSize, Npp32s nAnchor, Npp32s nDivisor)

  8-bit unsigned 1D (column) image convolution.
- NppStatus nppiFilterColumn_8u_C4R (const Npp8u ∗pSrc, Npp32s nSrcStep, Npp8u ∗pDst, Npp32s nDstStep, NppiSize oROI, const Npp32s ∗pKernel, Npp32s nMaskSize, Npp32s nAnchor, Npp32s nDivisor)

  4 channel 8-bit unsigned 1D (column) image convolution.
- NppStatus nppiFilterRow_8u_C1R (const Npp8u ∗pSrc, Npp32s nSrcStep, Npp8u ∗pDst, Npp32s nDstStep, NppiSize oROI, const Npp32s ∗pKernel, Npp32s nMaskSize, Npp32s nAnchor, Npp32s nDivisor)

  8-bit unsigned 1D (row) image convolution.
- NppStatus nppiFilterRow_8u_C4R (const Npp8u ∗pSrc, Npp32s nSrcStep, Npp8u ∗pDst, Npp32s nDstStep, NppiSize oROI, const Npp32s ∗pKernel, Npp32s nMaskSize, Npp32s nAnchor, Npp32s nDivisor)

  4 channel 8-bit unsigned 1D (row) image convolution.

1D Window Sum

1D mask Window Sum for 8 bit images.

- NppStatus nppiSumWindowColumn_8u32f_C1R (const Npp8u ∗pSrc, Npp32s nSrcStep, Npp32f ∗pDst, Npp32s nDstStep, NppiSize oROI, Npp32s nMaskSize, Npp32s nAnchor)

  8-bit unsigned 1D (column) sum to 32f.

- NppStatus nppiSumWindowRow_8u32f_C1R (const Npp8u ∗pSrc, Npp32s nSrcStep, Npp32f ∗pDst, Npp32s nDstStep, NppiSize oROI, Npp32s nMaskSize, Npp32s nAnchor)

  8-bit unsigned 1D (row) sum to 32f.

2D Morphology Filter

Image dilate and erod operations.

- NppStatus nppiDilate_8u_C1R (const Npp8u ∗pSrc, Npp32s nSrcStep, Npp8u ∗pDst, Npp32s nDstStep, NppiSize oSizeROI, const Npp8u ∗pMask, NppiSize oMaskSize, NppiPoint oAnchor)

  8-bit unsigned image dilation.

- NppStatus nppiDilate_8u_C4R (const Npp8u ∗pSrc, int nSrcStep, Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI, const Npp8u ∗pMask, NppiSize oMaskSize, NppiPoint oAnchor)

  4 channel 8-bit unsigned image dilation.

- NppStatus nppiErode_8u_C1R (const Npp8u ∗pSrc, Npp32s nSrcStep, Npp8u ∗pDst, Npp32s nDstStep, NppiSize oSizeROI, const Npp8u ∗pMask, NppiSize oMaskSize, NppiPoint oAnchor)

  8-bit unsigned image erosion.

- NppStatus nppiErode_8u_C4R (const Npp8u ∗pSrc, Npp32s nSrcStep, Npp8u ∗pDst, Npp32s nDstStep, NppiSize oSizeROI, const Npp8u ∗pMask, NppiSize oMaskSize, NppiPoint oAnchor)

  4 channel 8-bit unsigned image erosion.

Convolution (2D Masks)

General purpose 2D convolution filters.

- NppStatus nppiFilter_8u_C1R (const Npp8u ∗pSrc, Npp32s nSrcStep, Npp8u ∗pDst, Npp32s nDstStep, NppiSize oSizeROI, const Npp32s ∗pKernel, NppiSize oKernelSize, NppiPoint oAnchor, Npp32s nDivisor)

  8-bit unsigned convolution filter.

- NppStatus nppiFilter_8u_C4R (const Npp8u ∗pSrc, Npp32s nSrcStep, Npp8u ∗pDst, Npp32s nDstStep, NppiSize oSizeROI, const Npp32s ∗pKernel, NppiSize oKernelSize, NppiPoint oAnchor, Npp32s nDivisor)

  4 channel 8-bit unsigned convolution filter.

2D Linear Fixed Filters

2D linear fixed filters for 8 bit images.

- NppStatus nppiFilterBox_8u_C1R (const Npp8u ∗pSrc, Npp32s nSrcStep, Npp8u ∗pDst, Npp32s nDstStep, NppiSize oSizeROI, NppiSize oMaskSize, NppiPoint oAnchor)

  8-bit unsigned box filter.

- NppStatus nppiFilterBox_8u_C4R (const Npp8u ∗pSrc, Npp32s nSrcStep, Npp8u ∗pDst, Npp32s nDstStep, NppiSize oSizeROI, NppiSize oMaskSize, NppiPoint oAnchor)

  4 channel 8-bit unsigned box filter.

Image Rank Filters

Min, Median, and Max image filters.

- NppStatus nppiFilterMax_8u_C1R (const Npp8u ∗pSrc, Npp32s nSrcStep, Npp8u ∗pDst, Npp32s nDstStep, NppiSize oSizeROI, NppiSize oMaskSize, NppiPoint oAnchor)

  8-bit unsigned maximum filter.

- NppStatus nppiFilterMax_8u_C4R (const Npp8u ∗pSrc, Npp32s nSrcStep, Npp8u ∗pDst, Npp32s nDstStep, NppiSize oSizeROI, NppiSize oMaskSize, NppiPoint oAnchor)

  4 channel 8-bit unsigned maximum filter.

- NppStatus nppiFilterMin_8u_C1R (const Npp8u ∗pSrc, Npp32s nSrcStep, Npp8u ∗pDst, Npp32s nDstStep, NppiSize oSizeROI, NppiSize oMaskSize, NppiPoint oAnchor)

  8-bit unsigned minimum filter.

- NppStatus nppiFilterMin_8u_C4R (const Npp8u ∗pSrc, Npp32s nSrcStep, Npp8u ∗pDst, Npp32s nDstStep, NppiSize oSizeROI, NppiSize oMaskSize, NppiPoint oAnchor)

  4 channel 8-bit unsigned minimum filter.

Image Linear Transforms

Linear image transforms, like Fourier and DCT transformations.

- NppStatus nppiMagnitude_32fc32f_C1R (const Npp32fc ∗pSrc, int nSrcStep, Npp32f ∗pDst, int nDstStep, NppiSize oSizeROI)

  32-bit floating point complex to 32-bit floating point magnitude.

- NppStatus nppiMagnitudeSqr_32fc32f_C1R (const Npp32fc ∗pSrc, int nSrcStep, Npp32f ∗pDst, int nDstStep, NppiSize oSizeROI)

  32-bit floating point complex to 32-bit floating point squared magnitude.

Histogram

- NppStatus nppiEvenLevelsHost_32s (Npp32s ∗hpLevels, int nLevels, Npp32s nLowerLevel, Npp32s nUpperLevel)

  Compute levels with even distribution.

- NppStatus nppiHistogramEvenGetBufferSize_8u_C1R (NppiSize oSizeROI, int nLevels, int ∗hpBufferSize)

  Scratch-buffer size for nppiHistogramEven_8u_C1R.

- NppStatus nppiHistogramEven_8u_C1R (const Npp8u ∗pSrc, int nSrcStep, NppiSize oSizeROI, Npp32s ∗pHist, int nLevels, Npp32s nLowerLevel, Npp32s nUpperLevel, Npp8u ∗pBuffer)

  8-bit unsigned histogram with evenly distributed bins.

- NppStatus nppiHistogramEvenGetBufferSize_8u_C4R (NppiSize oSizeROI, int nLevels[4], int ∗hpBufferSize)

  Scratch-buffer size for nppiHistogramEven_8u_C4R.

- NppStatus nppiHistogramEven_8u_C4R (const Npp8u ∗pSrc, int nSrcStep, NppiSize oSizeROI, Npp32s ∗pHist[4], int nLevels[4], Npp32s nLowerLevel[4], Npp32s nUpperLevel[4], Npp8u ∗pBuffer)

  4 channel 8-bit unsigned histogram with evenly distributed bins.

- NppStatus nppiHistogramEvenGetBufferSize_8u_AC4R (NppiSize oSizeROI, int nLevels[3], int ∗hpBufferSize)

  Scratch-buffer size for nppiHistogramEven_8u_AC4R.

- NppStatus nppiHistogramEven_8u_AC4R (const Npp8u ∗pSrc, int nSrcStep, NppiSize oSizeROI, Npp32s ∗pHist[3], int nLevels[3], Npp32s nLowerLevel[3], Npp32s nUpperLevel[3], Npp8u ∗pBuffer)

  4 channel (alpha as the last channel) 8-bit unsigned histogram with evenly distributed bins.

- NppStatus nppiHistogramEvenGetBufferSize_16u_C1R (NppiSize oSizeROI, int nLevels, int ∗hpBufferSize)

  Scratch-buffer size for nppiHistogramEven_16u_C1R.

- NppStatus nppiHistogramEven_16u_C1R (const Npp16u ∗pSrc, int nSrcStep, NppiSize oSizeROI, Npp32s ∗pHist, int nLevels, Npp32s nLowerLevel, Npp32s nUpperLevel, Npp8u ∗pBuffer)

  16-bit unsigned histogram with evenly distributed bins.

- NppStatus nppiHistogramEvenGetBufferSize_16u_C4R (NppiSize oSizeROI, int nLevels[4], int ∗hpBufferSize)

  Scratch-buffer size for nppiHistogramEven_16u_C4R.

- NppStatus nppiHistogramEven_16u_C4R (const Npp16u ∗pSrc, int nSrcStep, NppiSize oSizeROI, Npp32s ∗pHist[4], int nLevels[4], Npp32s nLowerLevel[4], Npp32s nUpperLevel[4], Npp8u ∗pBuffer)

  4 channel 16-bit unsigned histogram with evenly distributed bins.

- NppStatus nppiHistogramEvenGetBufferSize_16u_AC4R (NppiSize oSizeROI, int nLevels[3], int ∗hpBufferSize)

  Scratch-buffer size for nppiHistogramEven_16u_AC4R.

- NppStatus nppiHistogramEven_16u_AC4R (const Npp16u ∗pSrc, int nSrcStep, NppiSize oSizeROI, Npp32s ∗pHist[3], int nLevels[3], Npp32s nLowerLevel[3], Npp32s nUpperLevel[3], Npp8u ∗pBuffer)

  4 channel (alpha as the last channel) 16-bit unsigned histogram with evenly distributed bins.

- NppStatus nppiHistogramEvenGetBufferSize_16s_C1R (NppiSize oSizeROI, int nLevels, int ∗hpBufferSize)

  Scratch-buffer size for nppiHistogramEven_16s_C1R.

- NppStatus nppiHistogramEven_16s_C1R (const Npp16s ∗pSrc, int nSrcStep, NppiSize oSizeROI, Npp32s ∗pHist, int nLevels, Npp32s nLowerLevel, Npp32s nUpperLevel, Npp8u ∗pBuffer)

  16-bit signed histogram with evenly distributed bins.

- NppStatus nppiHistogramEvenGetBufferSize_16s_C4R (NppiSize oSizeROI, int nLevels[4], int ∗hpBufferSize)

  Scratch-buffer size for nppiHistogramEven_16s_C4R.

- NppStatus nppiHistogramEven_16s_C4R (const Npp16s ∗pSrc, int nSrcStep, NppiSize oSizeROI, Npp32s ∗pHist[4], int nLevels[4], Npp32s nLowerLevel[4], Npp32s nUpperLevel[4], Npp8u ∗pBuffer)

  4 channel 16-bit signed histogram with evenly distributed bins.

- NppStatus nppiHistogramEvenGetBufferSize_16s_AC4R (NppiSize oSizeROI, int nLevels[3], int ∗hpBufferSize)

  Scratch-buffer size for nppiHistogramEven_16s_AC4R.

- NppStatus nppiHistogramEven_16s_AC4R (const Npp16s ∗pSrc, int nSrcStep, NppiSize oSizeROI, Npp32s ∗pHist[3], int nLevels[3], Npp32s nLowerLevel[3], Npp32s nUpperLevel[3], Npp8u ∗pBuffer)

  4 channel (alpha as the last channel) 16-bit signed histogram with evenly distributed bins.

- NppStatus nppiHistogramRangeGetBufferSize_8u_C1R (NppiSize oSizeROI, int nLevels, int ∗hpBufferSize)

  Scratch-buffer size for nppiHistogramRange_8u_C1R.

- NppStatus nppiHistogramRange_8u_C1R (const Npp8u ∗pSrc, int nSrcStep, NppiSize oSizeROI, Npp32s ∗pHist, const Npp32s ∗pLevels, int nLevels, Npp8u ∗pBuffer)

  8-bit unsigned histogram with bins determined by pLevels array.

- NppStatus nppiHistogramRangeGetBufferSize_8u_C4R (NppiSize oSizeROI, int nLevels[4], int ∗hpBufferSize)

  Scratch-buffer size for nppiHistogramRange_8u_C4R.

- NppStatus nppiHistogramRange_8u_C4R (const Npp8u ∗pSrc, int nSrcStep, NppiSize oSizeROI, Npp32s ∗pHist[4], const Npp32s ∗pLevels[4], int nLevels[4], Npp8u ∗pBuffer)

  4 channel 8-bit unsigned histogram with bins determined by pLevels.

- NppStatus nppiHistogramRangeGetBufferSize_8u_AC4R (NppiSize oSizeROI, int nLevels[3], int ∗hpBufferSize)

  Scratch-buffer size for nppiHistogramRange_8u_AC4R.

- NppStatus nppiHistogramRange_8u_AC4R (const Npp8u ∗pSrc, int nSrcStep, NppiSize oSizeROI, Npp32s ∗pHist[3], const Npp32s ∗pLevels[3], int nLevels[3], Npp8u ∗pBuffer)

  4 channel (alpha as a last channel) 8-bit unsigned histogram with bins determined by pLevels.

- NppStatus nppiHistogramRangeGetBufferSize_16u_C1R (NppiSize oSizeROI, int nLevels, int ∗hpBufferSize)

  Scratch-buffer size for nppiHistogramRange_16u_C1R.

- NppStatus nppiHistogramRange_16u_C1R (const Npp16u ∗pSrc, int nSrcStep, NppiSize oSizeROI, Npp32s ∗pHist, const Npp32s ∗pLevels, int nLevels, Npp8u ∗pBuffer)

  16-bit unsigned histogram with bins determined by pLevels array.

- NppStatus nppiHistogramRangeGetBufferSize_16u_C4R (NppiSize oSizeROI, int nLevels[4], int ∗hpBufferSize)

  Scratch-buffer size for nppiHistogramRange_16u_C4R.

- NppStatus nppiHistogramRange_16u_C4R (const Npp16u ∗pSrc, int nSrcStep, NppiSize oSizeROI, Npp32s ∗pHist[4], const Npp32s ∗pLevels[4], int nLevels[4], Npp8u ∗pBuffer)

  4 channel 16-bit unsigned histogram with bins determined by pLevels.

- NppStatus nppiHistogramRangeGetBufferSize_16u_AC4R (NppiSize oSizeROI, int nLevels[3], int ∗hpBufferSize)

  Scratch-buffer size for nppiHistogramRange_16u_AC4R.

- NppStatus nppiHistogramRange_16u_AC4R (const Npp16u ∗pSrc, int nSrcStep, NppiSize oSize-ROI, Npp32s ∗pHist[3], const Npp32s ∗pLevels[3], int nLevels[3], Npp8u ∗pBuffer)

  4 channel (alpha as a last channel) 16-bit unsigned histogram with bins determined by pLevels.

- NppStatus nppiHistogramRangeGetBufferSize_16s_C1R (NppiSize oSizeROI, int nLevels, int ∗hpBufferSize)

  Scratch-buffer size for nppiHistogramRange_16s_C1R.

- NppStatus nppiHistogramRange_16s_C1R (const Npp16s ∗pSrc, int nSrcStep, NppiSize oSizeROI, Npp32s ∗pHist, const Npp32s ∗pLevels, int nLevels, Npp8u ∗pBuffer)

  16-bit signed histogram with bins determined by pLevels array.

- NppStatus nppiHistogramRangeGetBufferSize_16s_C4R (NppiSize oSizeROI, int nLevels[4], int ∗hpBufferSize)

  Scratch-buffer size for nppiHistogramRange_16s_C4R.

- NppStatus nppiHistogramRange_16s_C4R (const Npp16s ∗pSrc, int nSrcStep, NppiSize oSizeROI, Npp32s ∗pHist[4], const Npp32s ∗pLevels[4], int nLevels[4], Npp8u ∗pBuffer)

  4 channel 16-bit signed histogram with bins determined by pLevels.

- NppStatus nppiHistogramRangeGetBufferSize_16s_AC4R (NppiSize oSizeROI, int nLevels[3], int ∗hpBufferSize)

  Scratch-buffer size for nppiHistogramRange_16s_AC4R.

- NppStatus nppiHistogramRange_16s_AC4R (const Npp16s ∗pSrc, int nSrcStep, NppiSize oSize-ROI, Npp32s ∗pHist[3], const Npp32s ∗pLevels[3], int nLevels[3], Npp8u ∗pBuffer)

  4 channel (alpha as a last channel) 16-bit signed histogram with bins determined by pLevels.

- NppStatus nppiHistogramRangeGetBufferSize_32f_C1R (NppiSize oSizeROI, int nLevels, int ∗hpBufferSize)

  Scratch-buffer size for nppiHistogramRange_32f_C1R.

- NppStatus nppiHistogramRange_32f_C1R (const Npp32f ∗pSrc, int nSrcStep, NppiSize oSizeROI, Npp32s ∗pHist, const Npp32f ∗pLevels, int nLevels, Npp8u ∗pBuffer)

  32-bit float histogram with bins determined by pLevels array.

- NppStatus nppiHistogramRangeGetBufferSize_32f_C4R (NppiSize oSizeROI, int nLevels[4], int ∗hpBufferSize)

  Scratch-buffer size for nppiHistogramRange_32f_C4R.

- NppStatus nppiHistogramRange_32f_C4R (const Npp32f ∗pSrc, int nSrcStep, NppiSize oSizeROI, Npp32s ∗pHist[4], const Npp32f ∗pLevels[4], int nLevels[4], Npp8u ∗pBuffer)

  4 channel 32-bit float histogram with bins determined by pLevels.

- NppStatus nppiHistogramRangeGetBufferSize_32f_AC4R (NppiSize oSizeROI, int nLevels[3], int ∗hpBufferSize)

  Scratch-buffer size for nppiHistogramRange_32f_AC4R.

- NppStatus nppiHistogramRange_32f_AC4R (const Npp32f ∗pSrc, int nSrcStep, NppiSize oSize-ROI, Npp32s ∗pHist[3], const Npp32f ∗pLevels[3], int nLevels[3], Npp8u ∗pBuffer)

  4 channel (alpha as a last channel) 32-bit float histogram with bins determined by pLevels.

JPEG DCT, Quantization and Level Shift Functions

Jpeg standard defines a flow of level shift, DCT and quantization for forward JPEG transform and inverse level shift, IDCT and de-quantization for inverse JPEG transform.

This group has the functions for both forward and inverse functions.

- NppStatus nppiQuantFwdRawTableInit_JPEG_8u (Npp8u ∗pQuantRawTable, int nQualityFactor)

  Converts regular quantization tables with the quality factor.

- NppStatus nppiQuantFwdTableInit_JPEG_8u16u (const Npp8u ∗pQuantRawTable, Npp16u ∗pQuantFwdRawTable)

  Converts raw quantization table to a forward quantization table.

- NppStatus nppiQuantInvTableInit_JPEG_8u16u (const Npp8u ∗pQuantRawTable, Npp16u ∗pQuantFwdRawTable)

  Converts raw quantization table to an inverse quantization table.

- NppStatus nppiSetDefaultQuantTable (Npp8u ∗pQuantRawTable, int tableIndex)

  Fills out the quantization table with either luminance and chrominance tables for JPEG.

- NppStatus nppiDCTQuantInv8x8LS_JPEG_16s8u_C1R (Npp16s ∗pSrc, int nSrcStep, Npp8u ∗pDst, int nDstStep, const Npp16u ∗pQuantInvTable, NppiSize oSizeROI)

  Inverse DCT, de-quantization and level shift part of the JPEG decoding.

- NppStatus nppiDCTQuantFwd8x8LS_JPEG_8u16s_C1R (Npp8u ∗pSrc, int nSrcStep, Npp16s ∗pDst, int nDstStep, const Npp16u ∗pQuantFwdTable, NppiSize oSizeROI)

  Forward DCT, quantization and level shift part of the JPEG encoding.

Sum

Sum of 8 bit images.

- NppStatus nppiReductionGetBufferHostSize_8u_C1R (const NppiSize &oSizeROI, int ∗hpBufferSize)

  Scratch-buffer size for nppiSum_8u_C1R.

- NppStatus nppiReductionGetBufferHostSize_8u_C4R (const NppiSize &oSizeROI, int ∗hpBufferSize)

  Scratch-buffer size for nppiSum_8u_C4R.

- NppStatus nppiSum_8u_C1R (const Npp8u ∗pSrc, int nSrcStep, NppiSize oSizeROI, Npp8u ∗pDeviceBuffer, Npp64f ∗pSum)

  8-bit unsigned image sum.

- NppStatus nppiSum_8u_C4R (const Npp8u ∗pSrc, int nSrcStep, NppiSize oSizeROI, Npp8u ∗pDeviceBuffer, Npp64f aSum[4])

  4 channel 8-bit unsigned image sum.

MinMax

Minimum and maximum of 8-bit images.

- NppStatus nppiMinMaxGetBufferSize_8u_C1R (const NppiSize &oSizeROI, int ∗hpBufferSize)

  Scratch-buffer size for nppiMinManx_8u_C1R.

- NppStatus nppiMinMax_8u_C1R (const Npp8u ∗pSrc, int nSrcStep, NppiSize oSizeROI, Npp8u ∗pMin, Npp8u ∗pMax, Npp8u ∗pDeviceBuffer)

  8-bit unsigned pixel minimum and maximum.

- NppStatus nppiMinMaxGetBufferSize_8u_C4R (const NppiSize &oSizeROI, int ∗hpBufferSize)

  Scratch-buffer size for nppiMinManx_8u_C4R.

- NppStatus nppiMinMax_8u_C4R (const Npp8u ∗pSrc, int nSrcStep, NppiSize oSizeROI, Npp8u aMin[4], Npp8u aMax[4], Npp8u ∗pDeviceBuffer)

  4 channel 8-bit unsigned pixel minimum and maximum.

Resize

Resizes 8 bit images.

Handles C1 and C4 images.

- NppStatus nppiResize_8u_C1R (const Npp8u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcROI, Npp8u ∗pDst, int nDstStep, NppiSize dstROISize, double xFactor, double yFactor, int interpolation)

  8-bit unsigned image resize.

- NppStatus nppiResize_8u_C4R (const Npp8u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcROI, Npp8u ∗pDst, int nDstStep, NppiSize dstROISize, double xFactor, double yFactor, int interpolation)

  4 channel 8-bit unsigned image resize.

Rotate

Rotates an image around the origin (0,0) and then shifts it.

- NppStatus nppiRotate_8u_C1R (const Npp8u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcROI, Npp8u ∗pDst, int nDstStep, NppiRect dstROI, double angle, double xShift, double yShift, int interpolation)

  8-bit unsigned image rotate.

- NppStatus nppiRotate_8u_C4R (const Npp8u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcROI, Npp8u ∗pDst, int nDstStep, NppiRect dstROI, double angle, double xShift, double yShift, int interpolation)

  4 channel 8-bit unsigned image rotate.

Mirror

Mirrors images horizontally, vertically and diagonally.

- NppStatus nppiMirror_8u_C1R (const Npp8u ∗pSrc, int nSrcStep, Npp8u ∗pDst, int nDstStep, NppiSize oROI, NppiAxis flip)

  8-bit unsigned image mirror.

- NppStatus nppiMirror_8u_C4R (const Npp8u ∗pSrc, int nSrcStep, Npp8u ∗pDst, int nDstStep, NppiSize oROI, NppiAxis flip)

  4 channel 8-bit unsigned image mirror.

RGBToYCbCr

RGB to YCbCr color conversion.

- NppStatus nppiRGBToYCbCr_8u_C3R (const Npp8u ∗pSrc, int nSrcStep, Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI)

  3 channel 8-bit unsigned packed RGB to packed YCbCr color conversion.

- NppStatus nppiRGBToYCbCr422_8u_C3C2R (const Npp8u ∗pSrc, int nSrcStep, Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI)

  3 channel 8-bit unsigned RGB to 2 channel chroma packed YCbCr422 color conversion.

- NppStatus nppiRGBToYCbCr420_8u_C3P3R (const Npp8u ∗pSrc, int nSrcStep, Npp8u ∗∗pDst, int nDstStep[3], NppiSize oSizeROI)

3 channel 8-bit unsigned packed RGB to planar YCbCr420 color conversion.

- NppStatus nppiRGBToYCbCr_8u_P3R (const Npp8u ∗const ∗pSrc, int nSrcStep, Npp8u ∗∗pDst, int nDstStep, NppiSize oSizeROI)

  3 channel planar 8-bit unsigned RGB to YCbCr color conversion.

- NppStatus nppiRGBToYCbCr_8u_AC4R (const Npp8u ∗pSrc, int nSrcStep, Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI)

  4 channel 8-bit unsigned RGB to YCbCr color conversion, ignoring Alpha.

YCbCrToRGB

YCbCr to RGB color conversion.

- NppStatus nppiYCbCrToRGB_8u_C3R (const Npp8u ∗pSrc, int nSrcStep, Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI)

  3 channel 8-bit unsigned packed YCbCr to RGB color conversion.

- NppStatus nppiYCbCrToRGB_8u_P3R (const Npp8u ∗const ∗pSrc, int nSrcStep, Npp8u ∗∗pDst, int nDstStep, NppiSize oSizeROI)

  3 channel 8-bit unsigned planar YCbCr to RGB color conversion.

- NppStatus nppiYCbCrToRGB_8u_AC4R (const Npp8u ∗pSrc, int nSrcStep, Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI)

  4 channel 8-bit unsigned packed YCbCr to RGB color conversion, not affecting Alpha.

- NppStatus nppiYCbCr422ToRGB_8u_C2C3R (const Npp8u ∗pSrc, int nSrcStep, Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI)

  2 channel 8-bit unsigned YCbCr422 to 3 channel packed RGB color conversion.

- NppStatus nppiYCbCr420ToRGB_8u_P3C3R (const Npp8u ∗const ∗pSrc, int nSrcStep[3], Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI)

  3 channel 8-bit unsigned planar YCbCr420 to packed RGB color conversion.

Sample Pattern Conversion.

- NppStatus nppiYCbCr422ToYCbCr420_8u_P3R (const Npp8u ∗const ∗pSrc, int nSrcStep[3], Npp8u ∗∗pDst, int nDstStep[3], NppiSize oSizeROI)

  3 channel 8-bit unsigned planar YCbCr:422 to YCbCr:420 resampling.

- NppStatus nppiYCbCr422ToYCbCr411_8u_P3R (const Npp8u ∗const ∗pSrc, int nSrcStep[3], Npp8u ∗∗pDst, int nDstStep[3], NppiSize oSizeROI)

  3 channel 8-bit unsigned planar YCbCr:422 to YCbCr:411 resampling.

- NppStatus nppiYCbCr420ToYCbCr422_8u_P3R (const Npp8u ∗const ∗pSrc, int nSrcStep[3], Npp8u ∗∗pDst, int nDstStep[3], NppiSize oSizeROI)

  3 channel 8-bit unsigned planar YCbCr:420 to YCbCr:422 resampling.

- NppStatus nppiYCbCr420ToYCbCr411_8u_P3P2R (const Npp8u ∗const ∗pSrc, int aSrcStep[3], Npp8u ∗pDstY, int nDstYStep, Npp8u ∗pDstCbCr, int nDstCbCrStep, NppiSize oSizeROI)

  3 channel 8-bit unsigned planar YCbCr:420 to YCbCr:411 resampling.

Color Processing

Color manipuliation functions.

- NppStatus nppiColorTwist32f_8u_C3R (const Npp8u ∗pSrc, int nSrcStep, Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI, const Npp32f twist[3][4])

  3 channel 8-bit unsigned color twist.

- NppStatus nppiColorTwist32f_8u_P3R (const Npp8u ∗const ∗pSrc, int nSrcStep, Npp8u ∗∗pDst, int nDstStep, NppiSize oSizeROI, const Npp32f twist[3][4])

  3 channel planar 8-bit unsigned color twist.

- NppStatus nppiColorTwist32f_8u_AC4R (const Npp8u ∗pSrc, int nSrcStep, Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI, const Npp32f twist[3][4])

  4 channel 8-bit unsigned color twist, not affecting Alpha.

- NppStatus nppiLUT_Linear_8u_C1R (const Npp8u ∗pSrc, int nSrcStep, Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI, const Npp32s ∗pValues, const Npp32s ∗pLevels, int nLevels)

  8-bit unsigned look-up-table color conversion.

- NppStatus nppiLUT_Linear_8u_C3R (const Npp8u ∗pSrc, int nSrcStep, Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI, const Npp32s ∗pValues[3], const Npp32s ∗pLevels[3], int nLevels[3])

  3 channel 8-bit unsigned look-up-table color conversion.

- NppStatus nppiLUT_Linear_8u_AC4R (const Npp8u ∗pSrc, int nSrcStep, Npp8u ∗pDst, int nDstStep, NppiSize oSizeROI, const Npp32s ∗pValues[4], const Npp32s ∗pLevels[4], int nLevels[4])

  4 channel 8-bit unsigned look-up-table color conversion, not affecting Alpha.

Affine warping, affine transform calculation

Affine warping of an image is the transform of image pixel positions, defined by the following formulas:

$$X_{new} = C_{00} * x + C_{01} * y + C_{02} \qquad Y_{new} = C_{10} * x + C_{11} * y + C_{12} \qquad C = \begin{bmatrix} C_{00} & C_{01} & C_{02} \\ C_{10} & C_{11} & C_{12} \end{bmatrix}$$

That is, any pixel with coordinates (X_{new}, Y_{new}) in the transformed image is sourced from coordinates (x, y) in the original image.

The mapping C is completely specified by 6 values $C_{ij}, i = \overline{0,1}, j = \overline{0,2}$. The transform maps parallel lines to parallel lines and preserves ratios of distances of points to lines. Implementation specific properties are discussed in each function's documentation.

- NppStatus nppiGetAffineTransform (NppiRect srcRoi, const double quad[4][2], double coeffs[2][3])

  Calculates affine transform coefficients given source rectangular ROI and its destination quadrangle projection.

- NppStatus nppiGetAffineQuad (NppiRect srcRoi, double quad[4][2], const double coeffs[2][3])

  Calculates affine transform projection of given source rectangular ROI.

- NppStatus nppiGetAffineBound (NppiRect srcRoi, double bound[2][2], const double coeffs[2][3])

  Calculates bounding box of the affine transform projection of the given source rectangular ROI.

- NppStatus nppiWarpAffine_8u_C1R (const Npp8u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp8u ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

  Affine transform of an image (8bit unsigned integer, single channel).

- NppStatus nppiWarpAffine_8u_C3R (const Npp8u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp8u ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

  Affine transform of an image (8bit unsigned integer, three channels).

- NppStatus nppiWarpAffine_8u_C4R (const Npp8u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp8u ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

  Affine transform of an image (8bit unsigned integer, four channels).

- NppStatus nppiWarpAffine_8u_AC4R (const Npp8u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp8u ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

  Affine transform of an image (8bit unsigned integer, four channels RGBA).

- NppStatus nppiWarpAffine_8u_P3R (const Npp8u ∗pSrc[3], NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp8u ∗pDst[3], int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

  Affine transform of an image (8bit unsigned integer, three planes).

- NppStatus nppiWarpAffine_8u_P4R (const Npp8u ∗pSrc[4], NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp8u ∗pDst[4], int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

  Affine transform of an image (8bit unsigned integer, four planes).

- NppStatus nppiWarpAffineBack_8u_C1R (const Npp8u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp8u ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

  Inverse affine transform of an image (8bit unsigned integer, single channel).

- NppStatus nppiWarpAffineBack_8u_C3R (const Npp8u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp8u ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

  Inverse affine transform of an image (8bit unsigned integer, three channels).

- NppStatus nppiWarpAffineBack_8u_C4R (const Npp8u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp8u ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

  Inverse affine transform of an image (8bit unsigned integer, four channels).

- NppStatus nppiWarpAffineBack_8u_AC4R (const Npp8u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp8u ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

Inverse affine transform of an image (8bit unsigned integer, four channels RGBA).

- NppStatus nppiWarpAffineBack_8u_P3R (const Npp8u ∗pSrc[3], NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp8u ∗pDst[3], int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

  Inverse affine transform of an image (8bit unsigned integer, three planes).

- NppStatus nppiWarpAffineBack_8u_P4R (const Npp8u ∗pSrc[4], NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp8u ∗pDst[4], int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

  Inverse affine transform of an image (8bit unsigned integer, four planes).

- NppStatus nppiWarpAffineQuad_8u_C1R (const Npp8u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, const double srcQuad[4][2], Npp8u ∗pDst, int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

  Affine transform of an image (8bit unsigned integer, single channel).

- NppStatus nppiWarpAffineQuad_8u_C3R (const Npp8u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, const double srcQuad[4][2], Npp8u ∗pDst, int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

  Affine transform of an image (8bit unsigned integer, three channels).

- NppStatus nppiWarpAffineQuad_8u_C4R (const Npp8u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, const double srcQuad[4][2], Npp8u ∗pDst, int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

  Affine transform of an image (8bit unsigned integer, four channels).

- NppStatus nppiWarpAffineQuad_8u_AC4R (const Npp8u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, const double srcQuad[4][2], Npp8u ∗pDst, int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

  Affine transform of an image (8bit unsigned integer, four channels RGBA).

- NppStatus nppiWarpAffineQuad_8u_P3R (const Npp8u ∗pSrc[3], NppiSize srcSize, int nSrcStep, NppiRect srcRoi, const double srcQuad[4][2], Npp8u ∗pDst[3], int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

  Affine transform of an image (8bit unsigned integer, three planes).

- NppStatus nppiWarpAffineQuad_8u_P4R (const Npp8u ∗pSrc[4], NppiSize srcSize, int nSrcStep, NppiRect srcRoi, const double srcQuad[4][2], Npp8u ∗pDst[4], int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

  Affine transform of an image (8bit unsigned integer, four planes).

- NppStatus nppiWarpAffine_16u_C1R (const Npp16u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp16u ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

  Affine transform of an image (16bit unsigned integer, single channel).

- NppStatus nppiWarpAffine_16u_C3R (const Npp16u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp16u ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

  Affine transform of an image (16bit unsigned integer, three channels).

- NppStatus nppiWarpAffine_16u_C4R (const Npp16u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp16u ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

  Affine transform of an image (16bit unsigned integer, four channels).

- NppStatus nppiWarpAffine_16u_AC4R (const Npp16u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp16u ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

  Affine transform of an image (16bit unsigned integer, four channels RGBA).

- NppStatus nppiWarpAffine_16u_P3R (const Npp16u ∗pSrc[3], NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp16u ∗pDst[3], int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

  Affine transform of an image (16bit unsigned integer, three planes).

- NppStatus nppiWarpAffine_16u_P4R (const Npp16u ∗pSrc[4], NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp16u ∗pDst[4], int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

  Affine transform of an image (16bit unsigned integer, four planes).

- NppStatus nppiWarpAffineBack_16u_C1R (const Npp16u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp16u ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

  Inverse affine transform of an image (16bit unsigned integer, single channel).

- NppStatus nppiWarpAffineBack_16u_C3R (const Npp16u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp16u ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

  Inverse affine transform of an image (16bit unsigned integer, three channels).

- NppStatus nppiWarpAffineBack_16u_C4R (const Npp16u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp16u ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

  Inverse affine transform of an image (16bit unsigned integer, four channels).

- NppStatus nppiWarpAffineBack_16u_AC4R (const Npp16u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp16u ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

  Inverse affine transform of an image (16bit unsigned integer, four channels RGBA).

- NppStatus nppiWarpAffineBack_16u_P3R (const Npp16u ∗pSrc[3], NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp16u ∗pDst[3], int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

  Inverse affine transform of an image (16bit unsigned integer, three planes).

- NppStatus nppiWarpAffineBack_16u_P4R (const Npp16u ∗pSrc[4], NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp16u ∗pDst[4], int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

  Inverse affine transform of an image (16bit unsigned integer, four planes).

- NppStatus nppiWarpAffineQuad_16u_C1R (const Npp16u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, const double srcQuad[4][2], Npp16u ∗pDst, int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

  Affine transform of an image (16bit unsigned integer, single channel).

- NppStatus nppiWarpAffineQuad_16u_C3R (const Npp16u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, const double srcQuad[4][2], Npp16u ∗pDst, int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

  Affine transform of an image (16bit unsigned integer, three channels).

- NppStatus nppiWarpAffineQuad_16u_C4R (const Npp16u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, const double srcQuad[4][2], Npp16u ∗pDst, int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

  Affine transform of an image (16bit unsigned integer, four channels).

- NppStatus nppiWarpAffineQuad_16u_AC4R (const Npp16u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, const double srcQuad[4][2], Npp16u ∗pDst, int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

  Affine transform of an image (16bit unsigned integer, four channels RGBA).

- NppStatus nppiWarpAffineQuad_16u_P3R (const Npp16u ∗pSrc[3], NppiSize srcSize, int nSrcStep, NppiRect srcRoi, const double srcQuad[4][2], Npp16u ∗pDst[3], int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

  Affine transform of an image (16bit unsigned integer, three planes).

- NppStatus nppiWarpAffineQuad_16u_P4R (const Npp16u ∗pSrc[4], NppiSize srcSize, int nSrcStep, NppiRect srcRoi, const double srcQuad[4][2], Npp16u ∗pDst[4], int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

  Affine transform of an image (16bit unsigned integer, four planes).

- NppStatus nppiWarpAffine_32f_C1R (const Npp32f ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp32f ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

  Affine transform of an image (32bit float, single channel).

- NppStatus nppiWarpAffine_32f_C3R (const Npp32f ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp32f ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

  Affine transform of an image (32bit float, three channels).

- NppStatus nppiWarpAffine_32f_C4R (const Npp32f ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp32f ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

  Affine transform of an image (32bit float, four channels).

- NppStatus nppiWarpAffine_32f_AC4R (const Npp32f ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp32f ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

  Affine transform of an image (32bit float, four channels RGBA).

- NppStatus nppiWarpAffine_32f_P3R (const Npp32f ∗pSrc[3], NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp32f ∗pDst[3], int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

  Affine transform of an image (32bit float, three planes).

- NppStatus nppiWarpAffine_32f_P4R (const Npp32f ∗pSrc[4], NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp32f ∗pDst[4], int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

  Affine transform of an image (32bit float, four planes).

- NppStatus nppiWarpAffineBack_32f_C1R (const Npp32f ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp32f ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

  Inverse affine transform of an image (32bit float, single channel).

- NppStatus nppiWarpAffineBack_32f_C3R (const Npp32f ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp32f ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

  Inverse affine transform of an image (32bit float, three channels).

- NppStatus nppiWarpAffineBack_32f_C4R (const Npp32f ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp32f ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

  Inverse affine transform of an image (32bit float, four channels).

- NppStatus nppiWarpAffineBack_32f_AC4R (const Npp32f ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp32f ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

  Inverse affine transform of an image (32bit float, four channels RGBA).

- NppStatus nppiWarpAffineBack_32f_P3R (const Npp32f ∗pSrc[3], NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp32f ∗pDst[3], int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

  Inverse affine transform of an image (32bit float, three planes).

- NppStatus nppiWarpAffineBack_32f_P4R (const Npp32f ∗pSrc[4], NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp32f ∗pDst[4], int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

  Inverse affine transform of an image (32bit float, four planes).

- NppStatus nppiWarpAffineQuad_32f_C1R (const Npp32f ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, const double srcQuad[4][2], Npp32f ∗pDst, int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

  Affine transform of an image (32bit float, single channel).

- NppStatus nppiWarpAffineQuad_32f_C3R (const Npp32f ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, const double srcQuad[4][2], Npp32f ∗pDst, int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

  Affine transform of an image (32bit float, three channels).

- NppStatus nppiWarpAffineQuad_32f_C4R (const Npp32f ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, const double srcQuad[4][2], Npp32f ∗pDst, int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

    Affine transform of an image (32bit float, four channels).

- NppStatus nppiWarpAffineQuad_32f_AC4R (const Npp32f ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, const double srcQuad[4][2], Npp32f ∗pDst, int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

    Affine transform of an image (32bit float, four channels RGBA).

- NppStatus nppiWarpAffineQuad_32f_P3R (const Npp32f ∗pSrc[3], NppiSize srcSize, int nSrcStep, NppiRect srcRoi, const double srcQuad[4][2], Npp32f ∗pDst[3], int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

    Affine transform of an image (32bit float, three planes).

- NppStatus nppiWarpAffineQuad_32f_P4R (const Npp32f ∗pSrc[4], NppiSize srcSize, int nSrcStep, NppiRect srcRoi, const double srcQuad[4][2], Npp32f ∗pDst[4], int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

    Affine transform of an image (32bit float, four planes).

- NppStatus nppiWarpAffine_32s_C1R (const Npp32s ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp32s ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

    Affine transform of an image (32bit signed integer, single channel).

- NppStatus nppiWarpAffine_32s_C3R (const Npp32s ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp32s ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

    Affine transform of an image (32bit signed integer, three channels).

- NppStatus nppiWarpAffine_32s_C4R (const Npp32s ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp32s ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

    Affine transform of an image (32bit signed integer, four channels).

- NppStatus nppiWarpAffine_32s_AC4R (const Npp32s ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp32s ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

    Affine transform of an image (32bit signed integer, four channels RGBA).

- NppStatus nppiWarpAffine_32s_P3R (const Npp32s ∗pSrc[3], NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp32s ∗pDst[3], int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

    Affine transform of an image (32bit signed integer, three planes).

- NppStatus nppiWarpAffine_32s_P4R (const Npp32s ∗pSrc[4], NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp32s ∗pDst[4], int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

    Affine transform of an image (32bit signed integer, four planes).

- NppStatus nppiWarpAffineBack_32s_C1R (const Npp32s ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp32s ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

  Inverse affine transform of an image (32bit signed integer, single channel).

- NppStatus nppiWarpAffineBack_32s_C3R (const Npp32s ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp32s ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

  Inverse affine transform of an image (32bit signed integer, three channels).

- NppStatus nppiWarpAffineBack_32s_C4R (const Npp32s ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp32s ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

  Inverse affine transform of an image (32bit signed integer, four channels).

- NppStatus nppiWarpAffineBack_32s_AC4R (const Npp32s ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp32s ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

  Inverse affine transform of an image (32bit signed integer, four channels RGBA).

- NppStatus nppiWarpAffineBack_32s_P3R (const Npp32s ∗pSrc[3], NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp32s ∗pDst[3], int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

  Inverse affine transform of an image (32bit signed integer, three planes).

- NppStatus nppiWarpAffineBack_32s_P4R (const Npp32s ∗pSrc[4], NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp32s ∗pDst[4], int nDstStep, NppiRect dstRoi, const double coeffs[2][3], int interpolation)

  Inverse affine transform of an image (32bit signed integer, four planes).

- NppStatus nppiWarpAffineQuad_32s_C1R (const Npp32s ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, const double srcQuad[4][2], Npp32s ∗pDst, int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

  Affine transform of an image (32bit signed integer, single channel).

- NppStatus nppiWarpAffineQuad_32s_C3R (const Npp32s ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, const double srcQuad[4][2], Npp32s ∗pDst, int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

  Affine transform of an image (32bit signed integer, three channels).

- NppStatus nppiWarpAffineQuad_32s_C4R (const Npp32s ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, const double srcQuad[4][2], Npp32s ∗pDst, int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

  Affine transform of an image (32bit signed integer, four channels).

- NppStatus nppiWarpAffineQuad_32s_AC4R (const Npp32s ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, const double srcQuad[4][2], Npp32s ∗pDst, int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

  Affine transform of an image (32bit signed integer, four channels RGBA).

- NppStatus nppiWarpAffineQuad_32s_P3R (const Npp32s ∗pSrc[3], NppiSize srcSize, int nSrcStep, NppiRect srcRoi, const double srcQuad[4][2], Npp32s ∗pDst[3], int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

  Affine transform of an image (32bit signed integer, three planes).

- NppStatus nppiWarpAffineQuad_32s_P4R (const Npp32s ∗pSrc[4], NppiSize srcSize, int nSrcStep, NppiRect srcRoi, const double srcQuad[4][2], Npp32s ∗pDst[4], int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

  Affine transform of an image (32bit signed integer, four planes).

Perspective warping, perspective transform calculation

Perspective warping of an image is the transform of image pixel positions, defined by the following formulas:

$$X_{new} = \frac{C_{00} * x + C_{01} * y + C_{02}}{C_{20} * x + C_{21} * y + C_{22}} \qquad Y_{new} = \frac{C_{10} * x + C_{11} * y + C_{12}}{C_{20} * x + C_{21} * y + C_{22}} \qquad C = \begin{bmatrix} C_{00} & C_{01} & C_{02} \\ C_{10} & C_{11} & C_{12} \\ C_{20} & C_{21} & C_{22} \end{bmatrix}$$

That is, any pixel of the transformed image with coordinates (X_{new}, Y_{new}) has a preimage with coordinates (x, y).

The mapping C is fully defined by 8 values $C_{ij}, (i, j) = \overline{0, 2}$, except of C_{22}, which is a normalizer. The transform has a property of mapping any convex quadrangle to a convex quadrangle, which is used in a group of functions nppiWarpPerspectiveQuad. The NPPI implementation of perspective transform has some issues which are discussed in each function's documentation.

- NppStatus nppiGetPerspectiveTransform (NppiRect srcRoi, const double quad[4][2], double coeffs[3][3])

  Calculates perspective transform coefficients given source rectangular ROI and its destination quadrangle projection.

- NppStatus nppiGetPerspectiveQuad (NppiRect srcRoi, double quad[4][2], const double coeffs[3][3])

  Calculates perspective transform projection of given source rectangular ROI.

- NppStatus nppiGetPerspectiveBound (NppiRect srcRoi, double bound[2][2], const double coeffs[3][3])

  Calculates bounding box of the perspective transform projection of the given source rectangular ROI.

- NppStatus nppiWarpPerspective_8u_C1R (const Npp8u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp8u ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Perspective transform of an image (8bit unsigned integer, single channel).

- NppStatus nppiWarpPerspective_8u_C3R (const Npp8u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp8u ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Perspective transform of an image (8bit unsigned integer, three channels).

- NppStatus nppiWarpPerspective_8u_C4R (const Npp8u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp8u ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Perspective transform of an image (8bit unsigned integer, four channels).

- NppStatus nppiWarpPerspective_8u_AC4R (const Npp8u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp8u ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Perspective transform of an image (8bit unsigned integer, four channels RGBA).

- NppStatus nppiWarpPerspective_8u_P3R (const Npp8u ∗pSrc[3], NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp8u ∗pDst[3], int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Perspective transform of an image (8bit unsigned integer, three planes).

- NppStatus nppiWarpPerspective_8u_P4R (const Npp8u ∗pSrc[4], NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp8u ∗pDst[4], int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Perspective transform of an image (8bit unsigned integer, four planes).

- NppStatus nppiWarpPerspectiveBack_8u_C1R (const Npp8u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp8u ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Inverse perspective transform of an image (8bit unsigned integer, single channel).

- NppStatus nppiWarpPerspectiveBack_8u_C3R (const Npp8u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp8u ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Inverse perspective transform of an image (8bit unsigned integer, three channels).

- NppStatus nppiWarpPerspectiveBack_8u_C4R (const Npp8u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp8u ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Inverse perspective transform of an image (8bit unsigned integer, four channels).

- NppStatus nppiWarpPerspectiveBack_8u_AC4R (const Npp8u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp8u ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Inverse perspective transform of an image (8bit unsigned integer, four channels RGBA).

- NppStatus nppiWarpPerspectiveBack_8u_P3R (const Npp8u ∗pSrc[3], NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp8u ∗pDst[3], int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Inverse perspective transform of an image (8bit unsigned integer, three planes).

- NppStatus nppiWarpPerspectiveBack_8u_P4R (const Npp8u ∗pSrc[4], NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp8u ∗pDst[4], int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Inverse perspective transform of an image (8bit unsigned integer, four planes).

- NppStatus nppiWarpPerspectiveQuad_8u_C1R (const Npp8u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, const double srcQuad[4][2], Npp8u ∗pDst, int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

  Perspective transform of an image (8bit unsigned integer, single channel).

- NppStatus nppiWarpPerspectiveQuad_8u_C3R (const Npp8u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, const double srcQuad[4][2], Npp8u ∗pDst, int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

  Perspective transform of an image (8bit unsigned integer, three channels).

- NppStatus nppiWarpPerspectiveQuad_8u_C4R (const Npp8u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, const double srcQuad[4][2], Npp8u ∗pDst, int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

  Perspective transform of an image (8bit unsigned integer, four channels).

- NppStatus nppiWarpPerspectiveQuad_8u_AC4R (const Npp8u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, const double srcQuad[4][2], Npp8u ∗pDst, int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

  Perspective transform of an image (8bit unsigned integer, four channels RGBA).

- NppStatus nppiWarpPerspectiveQuad_8u_P3R (const Npp8u ∗pSrc[3], NppiSize srcSize, int nSrcStep, NppiRect srcRoi, const double srcQuad[4][2], Npp8u ∗pDst[3], int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

  Perspective transform of an image (8bit unsigned integer, three planes).

- NppStatus nppiWarpPerspectiveQuad_8u_P4R (const Npp8u ∗pSrc[4], NppiSize srcSize, int nSrcStep, NppiRect srcRoi, const double srcQuad[4][2], Npp8u ∗pDst[4], int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

  Perspective transform of an image (8bit unsigned integer, four planes).

- NppStatus nppiWarpPerspective_16u_C1R (const Npp16u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp16u ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Perspective transform of an image (16bit unsigned integer, single channel).

- NppStatus nppiWarpPerspective_16u_C3R (const Npp16u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp16u ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Perspective transform of an image (16bit unsigned integer, three channels).

- NppStatus nppiWarpPerspective_16u_C4R (const Npp16u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp16u ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Perspective transform of an image (16bit unsigned integer, four channels).

- NppStatus nppiWarpPerspective_16u_AC4R (const Npp16u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp16u ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Perspective transform of an image (16bit unsigned integer, four channels RGBA).

- NppStatus nppiWarpPerspective_16u_P3R (const Npp16u ∗pSrc[3], NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp16u ∗pDst[3], int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Perspective transform of an image (16bit unsigned integer, three planes).

- NppStatus nppiWarpPerspective_16u_P4R (const Npp16u ∗pSrc[4], NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp16u ∗pDst[4], int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Perspective transform of an image (16bit unsigned integer, four planes).

- NppStatus nppiWarpPerspectiveBack_16u_C1R (const Npp16u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp16u ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Inverse perspective transform of an image (16bit unsigned integer, single channel).

- NppStatus nppiWarpPerspectiveBack_16u_C3R (const Npp16u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp16u ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Inverse perspective transform of an image (16bit unsigned integer, three channels).

- NppStatus nppiWarpPerspectiveBack_16u_C4R (const Npp16u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp16u ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Inverse perspective transform of an image (16bit unsigned integer, four channels).

- NppStatus nppiWarpPerspectiveBack_16u_AC4R (const Npp16u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp16u ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Inverse perspective transform of an image (16bit unsigned integer, four channels RGBA).

- NppStatus nppiWarpPerspectiveBack_16u_P3R (const Npp16u ∗pSrc[3], NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp16u ∗pDst[3], int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Inverse perspective transform of an image (16bit unsigned integer, three planes).

- NppStatus nppiWarpPerspectiveBack_16u_P4R (const Npp16u ∗pSrc[4], NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp16u ∗pDst[4], int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Inverse perspective transform of an image (16bit unsigned integer, four planes).

- NppStatus nppiWarpPerspectiveQuad_16u_C1R (const Npp16u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, const double srcQuad[4][2], Npp16u ∗pDst, int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

  Perspective transform of an image (16bit unsigned integer, single channel).

- NppStatus nppiWarpPerspectiveQuad_16u_C3R (const Npp16u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, const double srcQuad[4][2], Npp16u ∗pDst, int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

  Perspective transform of an image (16bit unsigned integer, three channels).

- NppStatus nppiWarpPerspectiveQuad_16u_C4R (const Npp16u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, const double srcQuad[4][2], Npp16u ∗pDst, int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

  Perspective transform of an image (16bit unsigned integer, four channels).

- NppStatus nppiWarpPerspectiveQuad_16u_AC4R (const Npp16u ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, const double srcQuad[4][2], Npp16u ∗pDst, int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

  Perspective transform of an image (16bit unsigned integer, four channels RGBA).

- NppStatus nppiWarpPerspectiveQuad_16u_P3R (const Npp16u ∗pSrc[3], NppiSize srcSize, int nSrcStep, NppiRect srcRoi, const double srcQuad[4][2], Npp16u ∗pDst[3], int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

  Perspective transform of an image (16bit unsigned integer, three planes).

- NppStatus nppiWarpPerspectiveQuad_16u_P4R (const Npp16u ∗pSrc[4], NppiSize srcSize, int nSrcStep, NppiRect srcRoi, const double srcQuad[4][2], Npp16u ∗pDst[4], int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

  Perspective transform of an image (16bit unsigned integer, four planes).

- NppStatus nppiWarpPerspective_32f_C1R (const Npp32f ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp32f ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Perspective transform of an image (32bit float, single channel).

- NppStatus nppiWarpPerspective_32f_C3R (const Npp32f ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp32f ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Perspective transform of an image (32bit float, three channels).

- NppStatus nppiWarpPerspective_32f_C4R (const Npp32f ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp32f ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Perspective transform of an image (32bit float, four channels).

- NppStatus nppiWarpPerspective_32f_AC4R (const Npp32f ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp32f ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Perspective transform of an image (32bit float, four channels RGBA).

- NppStatus nppiWarpPerspective_32f_P3R (const Npp32f ∗pSrc[3], NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp32f ∗pDst[3], int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Perspective transform of an image (32bit float, three planes).

- NppStatus nppiWarpPerspective_32f_P4R (const Npp32f ∗pSrc[4], NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp32f ∗pDst[4], int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Perspective transform of an image (32bit float, four planes).

- NppStatus nppiWarpPerspectiveBack_32f_C1R (const Npp32f ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp32f ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Inverse perspective transform of an image (32bit float, single channel).

- NppStatus nppiWarpPerspectiveBack_32f_C3R (const Npp32f ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp32f ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Inverse perspective transform of an image (32bit float, three channels).

- NppStatus nppiWarpPerspectiveBack_32f_C4R (const Npp32f ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp32f ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Inverse perspective transform of an image (32bit float, four channels).

- NppStatus nppiWarpPerspectiveBack_32f_AC4R (const Npp32f ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp32f ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Inverse perspective transform of an image (32bit float, four channels RGBA).

- NppStatus nppiWarpPerspectiveBack_32f_P3R (const Npp32f ∗pSrc[3], NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp32f ∗pDst[3], int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Inverse perspective transform of an image (32bit float, three planes).

- NppStatus nppiWarpPerspectiveBack_32f_P4R (const Npp32f ∗pSrc[4], NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp32f ∗pDst[4], int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Inverse perspective transform of an image (32bit float, four planes).

- NppStatus nppiWarpPerspectiveQuad_32f_C1R (const Npp32f ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, const double srcQuad[4][2], Npp32f ∗pDst, int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

  Perspective transform of an image (32bit float, single channel).

- NppStatus nppiWarpPerspectiveQuad_32f_C3R (const Npp32f ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, const double srcQuad[4][2], Npp32f ∗pDst, int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

  Perspective transform of an image (32bit float, three channels).

- NppStatus nppiWarpPerspectiveQuad_32f_C4R (const Npp32f ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, const double srcQuad[4][2], Npp32f ∗pDst, int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

  Perspective transform of an image (32bit float, four channels).

- NppStatus nppiWarpPerspectiveQuad_32f_AC4R (const Npp32f ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, const double srcQuad[4][2], Npp32f ∗pDst, int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

  Perspective transform of an image (32bit float, four channels RGBA).

- NppStatus nppiWarpPerspectiveQuad_32f_P3R (const Npp32f ∗pSrc[3], NppiSize srcSize, int nSrcStep, NppiRect srcRoi, const double srcQuad[4][2], Npp32f ∗pDst[3], int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

  Perspective transform of an image (32bit float, three planes).

- NppStatus nppiWarpPerspectiveQuad_32f_P4R (const Npp32f ∗pSrc[4], NppiSize srcSize, int nSrcStep, NppiRect srcRoi, const double srcQuad[4][2], Npp32f ∗pDst[4], int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

  Perspective transform of an image (32bit float, four planes).

- NppStatus nppiWarpPerspective_32s_C1R (const Npp32s ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp32s ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Perspective transform of an image (32bit signed integer, single channel).

- NppStatus nppiWarpPerspective_32s_C3R (const Npp32s ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp32s ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Perspective transform of an image (32bit signed integer, three channels).

- NppStatus nppiWarpPerspective_32s_C4R (const Npp32s ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp32s ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Perspective transform of an image (32bit signed integer, four channels).

- NppStatus nppiWarpPerspective_32s_AC4R (const Npp32s ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp32s ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Perspective transform of an image (32bit signed integer, four channels RGBA).

- NppStatus nppiWarpPerspective_32s_P3R (const Npp32s ∗pSrc[3], NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp32s ∗pDst[3], int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Perspective transform of an image (32bit signed integer, three planes).

- NppStatus nppiWarpPerspective_32s_P4R (const Npp32s ∗pSrc[4], NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp32s ∗pDst[4], int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Perspective transform of an image (32bit signed integer, four planes).

- NppStatus nppiWarpPerspectiveBack_32s_C1R (const Npp32s ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp32s ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Inverse perspective transform of an image (32bit signed integer, single channel).

- NppStatus nppiWarpPerspectiveBack_32s_C3R (const Npp32s ∗pSrc, NppiSize srcSize, int nSrcStep, NppiRect srcRoi, Npp32s ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Inverse perspective transform of an image (32bit signed integer, three channels).

- NppStatus nppiWarpPerspectiveBack_32s_C4R (const Npp32s ∗pSrc, NppiSize srcSize, int nSrc-Step, NppiRect srcRoi, Npp32s ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Inverse perspective transform of an image (32bit signed integer, four channels).

- NppStatus nppiWarpPerspectiveBack_32s_AC4R (const Npp32s ∗pSrc, NppiSize srcSize, int nSrc-Step, NppiRect srcRoi, Npp32s ∗pDst, int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Inverse perspective transform of an image (32bit signed integer, four channels RGBA).

- NppStatus nppiWarpPerspectiveBack_32s_P3R (const Npp32s ∗pSrc[3], NppiSize srcSize, int nSrc-Step, NppiRect srcRoi, Npp32s ∗pDst[3], int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Inverse perspective transform of an image (32bit signed integer, three planes).

- NppStatus nppiWarpPerspectiveBack_32s_P4R (const Npp32s ∗pSrc[4], NppiSize srcSize, int nSrc-Step, NppiRect srcRoi, Npp32s ∗pDst[4], int nDstStep, NppiRect dstRoi, const double coeffs[3][3], int interpolation)

  Inverse perspective transform of an image (32bit signed integer, four planes).

- NppStatus nppiWarpPerspectiveQuad_32s_C1R (const Npp32s ∗pSrc, NppiSize srcSize, int nSrc-Step, NppiRect srcRoi, const double srcQuad[4][2], Npp32s ∗pDst, int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

  Perspective transform of an image (32bit signed integer, single channel).

- NppStatus nppiWarpPerspectiveQuad_32s_C3R (const Npp32s ∗pSrc, NppiSize srcSize, int nSrc-Step, NppiRect srcRoi, const double srcQuad[4][2], Npp32s ∗pDst, int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

  Perspective transform of an image (32bit signed integer, three channels).

- NppStatus nppiWarpPerspectiveQuad_32s_C4R (const Npp32s ∗pSrc, NppiSize srcSize, int nSrc-Step, NppiRect srcRoi, const double srcQuad[4][2], Npp32s ∗pDst, int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

  Perspective transform of an image (32bit signed integer, four channels).

- NppStatus nppiWarpPerspectiveQuad_32s_AC4R (const Npp32s ∗pSrc, NppiSize srcSize, int nSrc-Step, NppiRect srcRoi, const double srcQuad[4][2], Npp32s ∗pDst, int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

  Perspective transform of an image (32bit signed integer, four channels RGBA).

- NppStatus nppiWarpPerspectiveQuad_32s_P3R (const Npp32s ∗pSrc[3], NppiSize srcSize, int nSrc-Step, NppiRect srcRoi, const double srcQuad[4][2], Npp32s ∗pDst[3], int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

  Perspective transform of an image (32bit signed integer, three planes).

- NppStatus nppiWarpPerspectiveQuad_32s_P4R (const Npp32s ∗pSrc[4], NppiSize srcSize, int nSrc-Step, NppiRect srcRoi, const double srcQuad[4][2], Npp32s ∗pDst[4], int nDstStep, NppiRect dstRoi, const double dstQuad[4][2], int interpolation)

  Perspective transform of an image (32bit signed integer, four planes).

Image Labeling Techniques

- NppStatus nppiGraphcutGetSize (NppiSize size, int ∗pBufSize)

  Calculates the size of the temporary buffer for graph-cut labeling.

- NppStatus nppiGraphcut_32s8u (Npp32s ∗pTerminals, Npp32s ∗pLeftTransposed, Npp32s ∗pRightTransposed, Npp32s ∗pTop, Npp32s ∗pBottom, int nStep, int nTransposedStep, NppiSize size, Npp8u ∗pLabel, int nLabelStep, Npp8u ∗pBuffer)

  Graphcut of a flow network (32bit signed integer edge capacities).

7.4.1 Function Documentation

7.4.1.1 NppStatus nppiAbsDiff_32f_C1R (const Npp32f ∗ *pSrc1*, int *nSrc1Step*, const Npp32f ∗ *pSrc2*, int *nSrc2Step*, Npp32f ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

32-bit floating point absolute difference.

Compute abs(sourcePixel1 - sourcePixel2).

Parameters:

pSrc1 Source-Image Pointer.

nSrc1Step Source-Image Line Step.

pSrc2 Source-Image Pointer.

nSrc2Step Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.2 NppStatus nppiAbsDiff_32s_C1R (const Npp32s ∗ *pSrc1*, int *nSrc1Step*, const Npp32s ∗ *pSrc2*, int *nSrc2Step*, Npp32s ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

32-bit absolute difference.

Compute abs(sourcePixel1 - sourcePixel2).

Parameters:

pSrc1 Source-Image Pointer.

nSrc1Step Source-Image Line Step.

pSrc2 Source-Image Pointer.

nSrc2Step Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.3 NppStatus nppiAbsDiff_8u_AC4R (const Npp8u ∗ *pSrc1*, int *nSrc1Step*, const Npp8u ∗ *pSrc2*, int *nSrc2Step*, Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

4 channel 8-bit unsigned absolute difference, not affecting Alpha.

Compute abs(sourcePixel1 - sourcePixel2).

Parameters:

pSrc1 Source-Image Pointer.

nSrc1Step Source-Image Line Step.

pSrc2 Source-Image Pointer.

nSrc2Step Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.4 NppStatus nppiAbsDiff_8u_C1R (const Npp8u ∗ *pSrc1*, int *nSrc1Step*, const Npp8u ∗ *pSrc2*, int *nSrc2Step*, Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

8-bit unsigned absolute difference.

Compute abs(sourcePixel1 - sourcePixel2).

Parameters:

pSrc1 Source-Image Pointer.

nSrc1Step Source-Image Line Step.

pSrc2 Source-Image Pointer.

nSrc2Step Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.5 NppStatus nppiAbsDiff_8u_C4R (const Npp8u ∗ *pSrc1*, int *nSrc1Step*, const Npp8u ∗ *pSrc2*, int *nSrc2Step*, Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

4 channel 8-bit unsigned absolute difference.

Compute abs(sourcePixel1 - sourcePixel2).

Parameters:

pSrc1 Source-Image Pointer.

nSrc1Step Source-Image Line Step.

pSrc2 Source-Image Pointer.

nSrc2Step Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.6 NppStatus nppiAbsDiffC_32f_C1R (const Npp32f ∗ *pSrc*, int *nSrcStep*, Npp32f ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*, Npp32f *nValue*)

32-bit floating point image absolute difference from constant.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

nValue Constant.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.7 NppStatus nppiAdd_32f_C1R (const Npp32f ∗ *pSrc1*, int *nSrc1Step*, const Npp32f ∗ *pSrc2*, int *nSrc2Step*, Npp32f ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

32-bit floating point image add.

Add the pixel values of corresponding pixels in the ROI and write them to the output image.

Parameters:

pSrc1 Source-Image Pointer.

nSrc1Step Source-Image Line Step.

pSrc2 Source-Image Pointer.
nSrc2Step Source-Image Line Step.
pDst Destination-Image Pointer.
nDstStep Destination-Image Line Step.
oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.8 NppStatus nppiAdd_32s_C1R (const Npp32s ∗ *pSrc1*, int *nSrc1Step*, const Npp32s ∗ *pSrc2*, int *nSrc2Step*, Npp32s ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

32-bit image add.

Add the pixel values of corresponding pixels in the ROI and write them to the output image.

Parameters:

pSrc1 Source-Image Pointer.
nSrc1Step Source-Image Line Step.
pSrc2 Source-Image Pointer.
nSrc2Step Source-Image Line Step.
pDst Destination-Image Pointer.
nDstStep Destination-Image Line Step.
oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.9 NppStatus nppiAdd_8u_AC4RSfs (const Npp8u ∗ *pSrc1*, int *nSrc1Step*, const Npp8u ∗ *pSrc2*, int *nSrc2Step*, Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*, int *nScaleFactor*)

4 channel 8-bit unsigned image add, not affecting Alpha.

Add the pixel values of corresponding pixels in the ROI and write them to the output image.

Parameters:

pSrc1 Source-Image Pointer.
nSrc1Step Source-Image Line Step.
pSrc2 Source-Image Pointer.
nSrc2Step Source-Image Line Step.
pDst Destination-Image Pointer.
nDstStep Destination-Image Line Step.
oSizeROI Region-of-Interest (ROI).
nScaleFactor Result pixel values are scaled by 2^(-nScaleFactor) and then clamped to [0,255] range.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.10 NppStatus nppiAdd_8u_C1RSfs (const Npp8u $*$ *pSrc1*, int *nSrc1Step*, const Npp8u $*$ *pSrc2*, int *nSrc2Step*, Npp8u $*$ *pDst*, int *nDstStep*, NppiSize *oSizeROI*, int *nScaleFactor*)

8-bit unsigned image add.

Add the pixel values of corresponding pixels in the ROI and write them to the output image.

Parameters:

pSrc1 Source-Image Pointer.

nSrc1Step Source-Image Line Step.

pSrc2 Source-Image Pointer.

nSrc2Step Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

nScaleFactor Result pixel values are scaled by 2^(-nScaleFactor) and then clamped to [0,255] range.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.11 NppStatus nppiAdd_8u_C4RSfs (const Npp8u $*$ *pSrc1*, int *nSrc1Step*, const Npp8u $*$ *pSrc2*, int *nSrc2Step*, Npp8u $*$ *pDst*, int *nDstStep*, NppiSize *oSizeROI*, int *nScaleFactor*)

4 channel 8-bit unsigned image add.

Add the pixel values of corresponding pixels in the ROI and write them to the output image.

Parameters:

pSrc1 Source-Image Pointer.

nSrc1Step Source-Image Line Step.

pSrc2 Source-Image Pointer.

nSrc2Step Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

nScaleFactor Result pixel values are scaled by 2^(-nScaleFactor) and then clamped to [0,255] range.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.12 NppStatus nppiAddC_32f_C1R (const Npp32f $*$ *pSrc*, int *nSrcStep*, Npp32f *nValue*, Npp32f $*$ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

32-bit floating point image add constant.

Parameters:

pSrc Source-Image Pointer.
nSrcStep Source-Image Line Step.
pDst Destination-Image Pointer.
nDstStep Destination-Image Line Step.
oSizeROI Region-of-Interest (ROI).
nValue Constant.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.13 NppStatus nppiAddC_32fc_C1R (const Npp32fc ∗ *pSrc*, int *nSrcStep*, Npp32fc *nValue*, Npp32fc ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

32-bit complex floating point image add constant.

Parameters:

pSrc Source-Image Pointer.
nSrcStep Source-Image Line Step.
pDst Destination-Image Pointer.
nDstStep Destination-Image Line Step.
oSizeROI Region-of-Interest (ROI).
nValue Constant.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.14 NppStatus nppiColorTwist32f_8u_AC4R (const Npp8u ∗ *pSrc*, int *nSrcStep*, Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*, const Npp32f *twist*[3][4])

4 channel 8-bit unsigned color twist, not affecting Alpha.

An input color twist matrix with floating-point pixel values is applied with in ROI. Alpha channel is the last channel and is not processed.

Parameters:

pSrc Source-Image Pointer.
nSrcStep Source-Image Line Step.
pDst Destination-Image Pointer.
nDstStep Destination-Image Line Step.
oSizeROI Region-of-Interest (ROI).
twist The color twist matrix with floating-point pixel values.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.15 NppStatus nppiColorTwist32f_8u_C3R (const Npp8u ∗ *pSrc*, int *nSrcStep*, Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*, const Npp32f *twist*[3][4])

3 channel 8-bit unsigned color twist.

An input color twist matrix with floating-point pixel values is applied within ROI.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

twist The color twist matrix with floating-point pixel values.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.16 NppStatus nppiColorTwist32f_8u_P3R (const Npp8u ∗const ∗ *pSrc*, int *nSrcStep*, Npp8u ∗∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*, const Npp32f *twist*[3][4])

3 channel planar 8-bit unsigned color twist.

An input color twist matrix with floating-point pixel values is applied within ROI.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

twist The color twist matrix with floating-point pixel values.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.17 NppStatus nppiCompare_32f_C1R (const Npp32f ∗ *pSrc1*, int *nSrc1Step*, const Npp32f ∗ *pSrc2*, int *nSrc2Step*, Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*, NppCmpOp *eComparisonOperation*)

32-bit floating point image compare.

Compare pSrc1's pixels with corresponding pixels in pSrc2.

Parameters:

pSrc1 Source-Image Pointer.

nSrc1Step Source-Image Line Step.

pSrc2 Source-Image Pointer.

nSrc2Step Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

eComparisonOperation Specifies the comparison operation to be used in the pixel comparison.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.18 NppStatus nppiCompare_8u_AC4R (const Npp8u ∗ *pSrc1*, int *nSrc1Step*, const Npp8u ∗ *pSrc2*, int *nSrc2Step*, Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*, NppCmpOp *eComparisonOperation*)

4 channel 8-bit unsigned image compare, not affecting Alpha.

Compare pSrc1's pixels with corresponding pixels in pSrc2.

Parameters:

pSrc1 Source-Image Pointer.

nSrc1Step Source-Image Line Step.

pSrc2 Source-Image Pointer.

nSrc2Step Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

eComparisonOperation Specifies the comparison operation to be used in the pixel comparison.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.19 NppStatus nppiCompare_8u_C4R (const Npp8u ∗ *pSrc1*, int *nSrc1Step*, const Npp8u ∗ *pSrc2*, int *nSrc2Step*, Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*, NppCmpOp *eComparisonOperation*)

4 channel 8-bit unsigned image compare.

Compare pSrc1's pixels with corresponding pixels in pSrc2.

Parameters:

pSrc1 Source-Image Pointer.

nSrc1Step Source-Image Line Step.

pSrc2 Source-Image Pointer.

nSrc2Step Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

eComparisonOperation Specifies the comparison operation to be used in the pixel comparison.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.20 NppStatus nppiConvert_16s32f_C1R (const Npp16s ∗ *pSrc*, int *nSrcStep*, Npp32f ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

16-bit singedto 32-bit floating point conversion.

For detailed documentation see nppiConverte_8u16u_C1R().

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.21 NppStatus nppiConvert_16s32s_C1R (const Npp16s ∗ *pSrc*, int *nSrcStep*, Npp32s ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

16-bit to 32-bit conversion.

For detailed documentation see nppiConvert_8u16u_C1R().

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.22 NppStatus nppiConvert_16s8u_AC4R (const Npp16s ∗ *pSrc*, int *nSrcStep*, Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

4 channel 16-bit signed to 8-bit unsigned conversion, not affecting Alpha.

For detailed documentation see nppiConverte_8u16u_C1R().

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.23 NppStatus nppiConvert_16s8u_C1R (const Npp16s ∗ *pSrc*, int *nSrcStep*, Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

4 channel 16-bit signed to 8-bit unsigned conversion.

For detailed documentation see nppiConvert_8u16u_C1R().

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.24 NppStatus nppiConvert_16s8u_C4R (const Npp16s ∗ *pSrc*, int *nSrcStep*, Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

4 channel 16-bit signed to 8-bit unsignedconversion, not affecting Alpha.

For detailed documentation see nppiConvert_8u16u_C1R().

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.25 NppStatus nppiConvert_16u32f_C1R (const Npp16u ∗ *pSrc*, int *nSrcStep*, Npp32f ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

16-bit unsigned to 32-bit floating point conversion.

For detailed documentation see nppiConverte_8u16u_C1R().

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.26 NppStatus nppiConvert_16u32s_C1R (const Npp16u ∗ *pSrc*, int *nSrcStep*, Npp32s ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

16-bit unsigned to 32-bit signed conversion.

For detailed documentation see nppiConverte_8u16u_C1R().

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.27 NppStatus nppiConvert_16u8u_AC4R (const Npp16u ∗ *pSrc*, int *nSrcStep*, Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

4 channel 16-bit unsigned to 8-bit unsigned conversion, not affecting Alpha.

For detailed documentation see nppiConvert_8u16u_C1R().

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.28 NppStatus nppiConvert_16u8u_C1R (const Npp16u ∗ *pSrc*, int *nSrcStep*, Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

16-bit unsigned to 8-bit unsigned conversion.

For detailed documentation see nppiConvert_8u16u_C1R().

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.29 NppStatus nppiConvert_16u8u_C4R (const Npp16u ∗ *pSrc*, int *nSrcStep*, Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

4 channel 16-bit unsigned to 8-bit unsigned conversion.

For detailed documentation see nppiConvert_8u16u_C1R().

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.30 NppStatus nppiConvert_32f16s_C1R (const Npp32f ∗ *pSrc*, int *nSrcStep*, Npp16s ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*, NppRoundMode *eRoundMode*)

32-bit floating point to 16-bit conversion.

For detailed documentation see nppiConverte_8u16u_C1R().

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

eRoundMode Flag specifying how fractional float values are rounded to integer values.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.31 NppStatus nppiConvert_32f16u_C1R (const Npp32f ∗ *pSrc*, int *nSrcStep*, Npp16u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*, NppRoundMode *eRoundMode*)

32-bit floating point to 16-bit unsigned conversion.

For detailed documentation see nppiConverte_8u16u_C1R().

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

eRoundMode Flag specifying how fractional float values are rounded to integer values.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.32 NppStatus nppiConvert_32f8u_C1R (const Npp32f ∗ *pSrc*, int *nSrcStep*, Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*, NppRoundMode *eRoundMode*)

32-bit floating point to 8-bit unsigned conversion.

For detailed documentation see nppiConverte_8u16u_C1R().

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

eRoundMode Flag specifying how fractional float values are rounded to integer values.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.33 NppStatus nppiConvert_8u16s_AC4R (const Npp8u ∗ *pSrc*, int *nSrcStep*, Npp16s ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

4 channel 8-bit unsigned to 16-bit signed conversion, not affecting Alpha.

For detailed documentation see nppiConverte_8u16u_C1R().

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.34 NppStatus nppiConvert_8u16s_C1R (const Npp8u ∗ *pSrc*, int *nSrcStep*, Npp16s ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

8-bit unsigned to 16-bit signed conversion.

For detailed documentation see nppiConvert_8u16u_C1R().

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.35 NppStatus nppiConvert_8u16s_C4R (const Npp8u * *pSrc*, int *nSrcStep*, Npp16s * *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

4 channel 8-bit unsigned to 16-bit signed conversion.

For detailed documentation see nppiConvert_8u16u_C1R().

Parameters:

pSrc Source-Image Pointer.
nSrcStep Source-Image Line Step.
pDst Destination-Image Pointer.
nDstStep Destination-Image Line Step.
oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.36 NppStatus nppiConvert_8u16u_AC4R (const Npp8u * *pSrc*, int *nSrcStep*, Npp16u * *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

4 channel 8-bit unsigned to 16-bit unsigned conversion, not affecting Alpha.

For detailed documentation see nppiConvert_8u16u_C1R().

Parameters:

pSrc Source-Image Pointer.
nSrcStep Source-Image Line Step.
pDst Destination-Image Pointer.
nDstStep Destination-Image Line Step.
oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.37 NppStatus nppiConvert_8u16u_C1R (const Npp8u * *pSrc*, int *nSrcStep*, Npp16u * *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

8-bit unsigned to 16-bit unsigned conversion.

Parameters:

pSrc Source-Image Pointer.
nSrcStep Source-Image Line Step.
pDst Destination-Image Pointer.
nDstStep Destination-Image Line Step.
oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.38 NppStatus nppiConvert_8u16u_C4R (const Npp8u ∗ *pSrc*, int *nSrcStep*, Npp16u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

4 channel 8-bit unsigned to 16-bit unsigned conversion.

For detailed documentation see nppiConvert_8u16u_C1R().

Parameters:

pSrc Source-Image Pointer.
nSrcStep Source-Image Line Step.
pDst Destination-Image Pointer.
nDstStep Destination-Image Line Step.
oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.39 NppStatus nppiConvert_8u32f_C1R (const Npp8u ∗ *pSrc*, int *nSrcStep*, Npp32f ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

8-bit unsigned to 32-bit floating point conversion.

For detailed documentation see nppiConverte_8u16u_C1R().

Parameters:

pSrc Source-Image Pointer.
nSrcStep Source-Image Line Step.
pDst Destination-Image Pointer.
nDstStep Destination-Image Line Step.
oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.40 NppStatus nppiCopy_16s_AC4R (const Npp16s ∗ *pSrc*, int *nSrcStep*, Npp16s ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

4 channel 16-bit image copy, not affecting Alpha.

Parameters:

pSrc Source-Image Pointer.
nSrcStep Source-Image Line Step.
pDst Destination-Image Pointer.
nDstStep Destination-Image Line Step.
oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.41 NppStatus nppiCopy_16s_C1R (const Npp16s ∗ *pSrc*, int *nSrcStep*, Npp16s ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

16-bit image copy.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.42 NppStatus nppiCopy_16s_C4R (const Npp16s ∗ *pSrc*, int *nSrcStep*, Npp16s ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

4 channel 16-bit image copy.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.43 NppStatus nppiCopy_16u_AC4R (const Npp16u ∗ *pSrc*, int *nSrcStep*, Npp16u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

4 channel 16-bit unsigned image copy, not affecting Alpha channel.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.44 NppStatus nppiCopy_16u_C1R (const Npp16u ∗ *pSrc*, int *nSrcStep*, Npp16u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

16-bit unsigned image copy.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.45 NppStatus nppiCopy_16u_C4R (const Npp16u ∗ *pSrc*, int *nSrcStep*, Npp16u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

4 channel 16-bit unsigned image copy.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.46 NppStatus nppiCopy_32f_AC4R (const Npp32f ∗ *pSrc*, int *nSrcStep*, Npp32f ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

4 channel 32-bit floating point image copy, not affecting Alpha.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.47 NppStatus nppiCopy_32f_C1R (const Npp32f $*$ *pSrc*, int *nSrcStep*, Npp32f $*$ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

32-bit floating point image copy.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.48 NppStatus nppiCopy_32f_C4R (const Npp32f $*$ *pSrc*, int *nSrcStep*, Npp32f $*$ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

4 channel 32-bit floating point image copy.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.49 NppStatus nppiCopy_32s_AC4R (const Npp32s $*$ *pSrc*, int *nSrcStep*, Npp32s $*$ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

4 channel 32-bit image copy, not affecting Alpha.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.50 NppStatus nppiCopy_32s_C1R (const Npp32s ∗ *pSrc*, int *nSrcStep*, Npp32s ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

32-bit image copy.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.51 NppStatus nppiCopy_32s_C4R (const Npp32s ∗ *pSrc*, int *nSrcStep*, Npp32s ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

4 channel 32-bit image copy.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.52 NppStatus nppiCopy_8u_AC4R (const Npp8u ∗ *pSrc*, int *nSrcStep*, Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

4 channel 8-bit unsigned image copy, not affecting Alpha channel.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.53 NppStatus nppiCopy_8u_C1R (const Npp8u ∗ *pSrc*, int *nSrcStep*, Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

8-bit unsigned image copy.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.54 NppStatus nppiCopy_8u_C4R (const Npp8u ∗ *pSrc*, int *nSrcStep*, Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

4 channel 8-bit unsigned image copy.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.55 NppStatus nppiCopyConstBorder_32s_C1R (const Npp32s ∗ *pSrc*, int *nSrcStep*, NppiSize *oSrcSizeROI*, Npp32s ∗ *pDst*, int *nDstStep*, NppiSize *oDstSizeROI*, int *nTopBorderHeight*, int *nLeftBorderWidth*, Npp32s *nValue*)

32-bit image copy with constant border color.

See nppiCopyConstBorder_8u_C1R() for detailed documentation.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

oSrcSizeROI Size of the source region-of-interest.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oDstSizeROI Size of the destination region-of-interest.

nTopBorderHeight Height of top border.

nLeftBorderWidth Width of left border.

nValue Border luminance value.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.56 NppStatus nppiCopyConstBorder_8u_AC4R (const Npp8u ∗ *pSrc*, int *nSrcStep*, NppiSize *oSrcSizeROI*, Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *oDstSizeROI*, int *nTopBorderHeight*, int *nLeftBorderWidth*, const Npp8u *aValue*[3])

4 channel 8-bit unsigned image copy with constant border color.

See nppiCopyConstBorder_8u_C1R() for detailed documentation.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

oSrcSizeROI Size of the source region-of-interest.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oDstSizeROI Size of the destination region-of-interest.

nTopBorderHeight Height of top border.

nLeftBorderWidth Width of left border.

aValue Vector of the RGB values of the border pixels. Because this method does not affect the destination image's alpha channel, only three components of the border color are needed.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.57 NppStatus nppiCopyConstBorder_8u_C1R (const Npp8u ∗ *pSrc*, int *nSrcStep*, NppiSize *oSrcSizeROI*, Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *oDstSizeROI*, int *nTopBorderHeight*, int *nLeftBorderWidth*, Npp8u *nValue*)

8-bit unsigned image copy width constant border color.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

oSrcSizeROI Size of the source region of pixels.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oDstSizeROI Size (width, height) of the destination region, i.e. the region that gets filled with data from the source image (inner part) and constant border color (outer part).

nTopBorderHeight Height (in pixels) of the top border. The height of the border at the bottom of the destination ROI is implicitly defined by the size of the source ROI: nBottomBorderHeight = oDstSizeROI.height - nTopBorderHeight - oSrcSizeROI.height.

nLeftBorderWidth Width (in pixels) of the left border. The width of the border at the right side of the destination ROI is implicitly defined by the size of the source ROI: nRightBorderWidth = oDstSizeROI.width - nLeftBorderWidth - oSrcSizeROI.width.

nValue The pixel value to be set for border pixels.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.58 NppStatus nppiCopyConstBorder_8u_C4R (const Npp8u ∗ *pSrc*, int *nSrcStep*, NppiSize *oSrcSizeROI*, Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *oDstSizeROI*, int *nTopBorderHeight*, int *nLeftBorderWidth*, const Npp8u *aValue*[4])

4channel 8-bit unsigned image copy with constant border color.

See nppiCopyConstBorder_8u_C1R() for detailed documentation.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

oSrcSizeROI Size of the source region-of-interest.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oDstSizeROI Size of the destination region-of-interest.

nTopBorderHeight Height of top border.

nLeftBorderWidth Width of left border.

aValue Vector of the RGBA values of the border pixels to be set.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.59 NppStatus nppiDCTQuantFwd8x8LS_JPEG_8u16s_C1R (Npp8u ∗ *pSrc*, int *nSrcStep*, Npp16s ∗ *pDst*, int *nDstStep*, const Npp16u ∗ *pQuantFwdTable*, NppiSize *oSizeROI*)

Forward DCT, quantization and level shift part of the JPEG encoding.

Input is expected in 8x8 macro blocks and output is expected to be in 64x1 macro blocks.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

pQuantFwdTable Forward quantization tables for JPEG encoding.

oSizeROI Region-of-Interest (ROI).

Returns:

Error codes:

- NPP_SIZE_ERROR For negative input height/width or not a multiple of 8 width/height.
- NPP_STEP_ERROR If input image width is not multiple of 8 or does not match ROI.
- NPP_NULL_POINTER_ERROR If the destination pointer is NULL.

7.4.1.60 NppStatus nppiDCTQuantInv8x8LS_JPEG_16s8u_C1R (Npp16s ∗ *pSrc*, int *nSrcStep*, Npp8u ∗ *pDst*, int *nDstStep*, const Npp16u ∗ *pQuantInvTable*, NppiSize *oSizeROI*)

Inverse DCT, de-quantization and level shift part of the JPEG decoding.

Input is expected in 64x1 macro blocks and output is expected to be in 8x8 macro blocks.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

pQuantInvTable Inverse quantization tables for JPEG decoding.

oSizeROI Region-of-Interest (ROI).

Returns:

Error codes:

- NPP_SIZE_ERROR For negative input height/width or not a multiple of 8 width/height.
- NPP_STEP_ERROR If input image width is not multiple of 8 or does not match ROI.
- NPP_NULL_POINTER_ERROR If the destination pointer is NULL.

7.4.1.61 NppStatus nppiDilate_8u_C1R (const Npp8u ∗ *pSrc*, Npp32s *nSrcStep*, Npp8u ∗ *pDst*, Npp32s *nDstStep*, NppiSize *oSizeROI*, const Npp8u ∗ *pMask*, NppiSize *oMaskSize*, NppiPoint *oAnchor*)

8-bit unsigned image dilation.

Dilation computes the output pixel as the maximum pixel value of the pixels under the mask. Pixels who's corresponding mask values are zero to not participate in the maximum search.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).
pMask Pointer to the start address of the mask array
oMaskSize Width and Height mask array.
oAnchor X and Y offsets of the mask origin frame of reference w.r.t the source pixel.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.62 NppStatus nppiDilate_8u_C4R (const Npp8u * *pSrc*, int *nSrcStep*, Npp8u * *pDst*, int *nDstStep*, NppiSize *oSizeROI*, const Npp8u * *pMask*, NppiSize *oMaskSize*, NppiPoint *oAnchor*)

4 channel 8-bit unsigned image dilation.

Dilation computes the output pixel as the maximum pixel value of the pixels under the mask. Pixels who's corresponding mask values are zero to not participate in the maximum search.

Parameters:

pSrc Source-Image Pointer.
nSrcStep Source-Image Line Step.
pDst Destination-Image Pointer.
nDstStep Destination-Image Line Step.
oSizeROI Region-of-Interest (ROI).
pMask Pointer to the start address of the mask array
oMaskSize Width and Height mask array.
oAnchor X and Y offsets of the mask origin frame of reference w.r.t the source pixel.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.63 NppStatus nppiDiv_32f_C1R (const Npp32f * *pSrc1*, int *nSrc1Step*, const Npp32f * *pSrc2*, int *nSrc2Step*, Npp32f * *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

32-bit floating point image division.

Divide pixels in pSrc2 by pSrc1's pixels.

Parameters:

pSrc1 Source-Image Pointer.
nSrc1Step Source-Image Line Step.
pSrc2 Source-Image Pointer.
nSrc2Step Source-Image Line Step.
pDst Destination-Image Pointer.
nDstStep Destination-Image Line Step.
oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.64 NppStatus nppiDiv_32s_C1R (const Npp32s ∗ *pSrc1*, int *nSrc1Step*, const Npp32s ∗ *pSrc2*, int *nSrc2Step*, Npp32s ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

32-bit image division.

Divide pixels in pSrc2 by pSrc1's pixels.

Parameters:

pSrc1 Source-Image Pointer.

nSrc1Step Source-Image Line Step.

pSrc2 Source-Image Pointer.

nSrc2Step Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.65 NppStatus nppiDiv_8u_AC4RSfs (const Npp8u ∗ *pSrc1*, int *nSrc1Step*, const Npp8u ∗ *pSrc2*, int *nSrc2Step*, Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*, int *nScaleFactor*)

4 channel 8-bit unsigned image division, not affecting Alpha.

Divide pixels in pSrc2 by pSrc1's pixels.

Parameters:

pSrc1 Source-Image Pointer.

nSrc1Step Source-Image Line Step.

pSrc2 Source-Image Pointer.

nSrc2Step Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

nScaleFactor Result pixel values are scaled by 2^(-nScaleFactor) and then clamped to [0,255] range.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.66 NppStatus nppiDiv_8u_C1RSfs (const Npp8u ∗ *pSrc1*, int *nSrc1Step*, const Npp8u ∗ *pSrc2*, int *nSrc2Step*, Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*, int *nScaleFactor*)

8-bit unsignedimage division.

Dived pixels in pSrc2 by pSrc1's pixels.

Parameters:

pSrc1 Source-Image Pointer.

nSrc1Step Source-Image Line Step.

pSrc2 Source-Image Pointer.

nSrc2Step Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

nScaleFactor Result pixel values are scaled by 2^(-nScaleFactor) and then clamped to [0,255] range.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.67 NppStatus nppiDiv_8u_C4RSfs (const Npp8u ∗ *pSrc1*, int *nSrc1Step*, const Npp8u ∗ *pSrc2*, int *nSrc2Step*, Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*, int *nScaleFactor*)

4 channel 8-bit unsigned image division.

Divide pixels in pSrc2 by pSrc1's pixels.

Parameters:

pSrc1 Source-Image Pointer.

nSrc1Step Source-Image Line Step.

pSrc2 Source-Image Pointer.

nSrc2Step Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

nScaleFactor Result pixel values are scaled by 2^(-nScaleFactor) and then clamped to [0,255] range.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.68 NppStatus nppiDivC_32f_C1R (const Npp32f ∗ *pSrc*, int *nSrcStep*, Npp32f *nValue*, Npp32f ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

32-bit floating point image divide by constant.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

nValue Constant.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.69 NppStatus nppiDivC_32fc_C1R (const Npp32fc ∗ *pSrc*, int *nSrcStep*, Npp32fc *nValue*, Npp32fc ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

32-bit complex floating point image divide by constant.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

nValue Constant.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.70 NppStatus nppiErode_8u_C1R (const Npp8u ∗ *pSrc*, Npp32s *nSrcStep*, Npp8u ∗ *pDst*, Npp32s *nDstStep*, NppiSize *oSizeROI*, const Npp8u ∗ *pMask*, NppiSize *oMaskSize*, NppiPoint *oAnchor*)

8-bit unsigned image erosion.

Erosion computes the output pixel as the minimum pixel value of the pixels under the mask. Pixels who's corresponding mask values are zero to not participate in the maximum search.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pMask Pointer to the start address of the mask array

oMaskSize Width and Height mask array.

oAnchor X and Y offsets of the mask origin frame of reference w.r.t the source pixel.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.71 NppStatus nppiErode_8u_C4R (const Npp8u ∗ *pSrc*, Npp32s *nSrcStep*, Npp8u ∗ *pDst*, Npp32s *nDstStep*, NppiSize *oSizeROI*, const Npp8u ∗ *pMask*, NppiSize *oMaskSize*, NppiPoint *oAnchor*)

4 channel 8-bit unsigned image erosion.

Erosion computes the output pixel as the minimum pixel value of the pixels under the mask. Pixels who's corresponding mask values are zero to not participate in the maximum search.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pMask Pointer to the start address of the mask array

oMaskSize Width and Height mask array.

oAnchor X and Y offsets of the mask origin frame of reference w.r.t the source pixel.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.72 NppStatus nppiEvenLevelsHost_32s (Npp32s ∗ *hpLevels*, int *nLevels*, Npp32s *nLowerLevel*, Npp32s *nUpperLevel*)

Compute levels with even distribution.

Parameters:

hpLevels A host pointer to array which receives the levels being computed. The array needs to be of size nLevels.

nLevels The number of levels being computed. nLevels must be at least 2, otherwise an NPP_-HISTO_NUMBER_OF_LEVELS_ERROR error is returned.

nLowerLevel Lower boundary value of the lowest level.

nUpperLevel Upper boundary value of the greatest level.

Returns:

Error code.

7.4.1.73 NppStatus nppiExp_32f_C1R (const Npp32f ∗ *pSrc*, int *nSrcStep*, Npp32f ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

32-bit floating point exponentiation.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.74 NppStatus nppiFilter_8u_C1R (const Npp8u ∗ *pSrc*, Npp32s *nSrcStep*, Npp8u ∗ *pDst*, Npp32s *nDstStep*, NppiSize *oSizeROI*, const Npp32s ∗ *pKernel*, NppiSize *oKernelSize*, NppiPoint *oAnchor*, Npp32s *nDivisor*)

8-bit unsigned convolution filter.

Pixels under the mask are multiplied by the respective weights in the mask and the results are summed. Before writing the result pixel the sum is scaled back via division by nDivisor.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pKernel Pointer to the start address of the kernel coefficient array. Coeffcients are expected to be stored in reverse order.

oKernelSize Width and Height of the rectangular kernel.

oAnchor X and Y offsets of the kernel origin frame of reference w.r.t the source pixel.

nDivisor The factor by which the convolved summation from the Filter operation should be divided. If equal to the sum of coefficients, this will keep the maximum result value within full scale.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.75 NppStatus nppiFilter_8u_C4R (const Npp8u ∗ *pSrc*, Npp32s *nSrcStep*, Npp8u ∗ *pDst*, Npp32s *nDstStep*, NppiSize *oSizeROI*, const Npp32s ∗ *pKernel*, NppiSize *oKernelSize*, NppiPoint *oAnchor*, Npp32s *nDivisor*)

4 channel 8-bit unsigned convolution filter.

Pixels under the mask are multiplied by the respective weights in the mask and the results are summed. Before writing the result pixel the sum is scaled back via division by nDivisor.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pKernel Pointer to the start address of the kernel coefficient array. Coeffcients are expected to be stored in reverse order.

oKernelSize Width and Height of the rectangular kernel.

oAnchor X and Y offsets of the kernel origin frame of reference w.r.t the source pixel.

nDivisor The factor by which the convolved summation from the Filter operation should be divided. If equal to the sum of coefficients, this will keep the maximum result value within full scale.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.76 NppStatus nppiFilterBox_8u_C1R (const Npp8u ∗ *pSrc*, Npp32s *nSrcStep*, Npp8u ∗ *pDst*, Npp32s *nDstStep*, NppiSize *oSizeROI*, NppiSize *oMaskSize*, NppiPoint *oAnchor*)

8-bit unsigned box filter.

Computes the average pixel values of the pixels under a rectangular mask.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

oMaskSize Width and Height of the neighborhood region for the local Avg operation.

oAnchor X and Y offsets of the kernel origin frame of reference w.r.t the source pixel.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.77 NppStatus nppiFilterBox_8u_C4R (const Npp8u ∗ *pSrc*, Npp32s *nSrcStep*, Npp8u ∗ *pDst*, Npp32s *nDstStep*, NppiSize *oSizeROI*, NppiSize *oMaskSize*, NppiPoint *oAnchor*)

4 channel 8-bit unsigned box filter.

Computes the average pixel values of the pixels under a rectangular mask.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

oMaskSize Width and Height of the neighborhood region for the local Avg operation.

oAnchor X and Y offsets of the kernel origin frame of reference w.r.t the source pixel.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.78 NppStatus nppiFilterColumn_8u_C1R (const Npp8u ∗ *pSrc*, Npp32s *nSrcStep*, Npp8u ∗ *pDst*, Npp32s *nDstStep*, NppiSize *oROI*, const Npp32s ∗ *pKernel*, Npp32s *nMaskSize*, Npp32s *nAnchor*, Npp32s *nDivisor*)

8-bit unsigned 1D (column) image convolution.

Apply convolution filter with user specified 1D column of weights. Result pixel is equal to the sum of the products between the kernel coefficients (pKernel array) and corresponding neighboring column pixel values in the source image defined by nKernelDim and nAnchorY, divided by nDivisor.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oROI Region-of-Interest (ROI).

pKernel Pointer to the start address of the kernel coefficient array. Coefficients are expected to be stored in reverse order.

nMaskSize Length of the linear kernel array.

nAnchor Y offset of the kernel origin frame of reference w.r.t the source pixel.

nDivisor The factor by which the convolved summation from the Filter operation should be divided. If equal to the sum of coefficients, this will keep the maximum result value within full scale.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.79 NppStatus nppiFilterColumn_8u_C4R (const Npp8u ∗ *pSrc*, Npp32s *nSrcStep*, Npp8u ∗ *pDst*, Npp32s *nDstStep*, NppiSize *oROI*, const Npp32s ∗ *pKernel*, Npp32s *nMaskSize*, Npp32s *nAnchor*, Npp32s *nDivisor*)

4 channel 8-bit unsigned 1D (column) image convolution.

Apply convolution filter with user specified 1D column of weights. Result pixel is equal to the sum of the products between the kernel coefficients (pKernel array) and corresponding neighboring column pixel values in the source image defined by nKernelDim and nAnchorY, divided by nDivisor.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oROI Region-of-Interest (ROI).

pKernel Pointer to the start address of the kernel coefficient array. Coefficients are expected to be stored in reverse order.

nMaskSize Length of the linear kernel array.

nAnchor Y offset of the kernel origin frame of reference w.r.t the source pixel.

nDivisor The factor by which the convolved summation from the Filter operation should be divided. If equal to the sum of coefficients, this will keep the maximum result value within full scale.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.80 NppStatus nppiFilterMax_8u_C1R (const Npp8u ∗ *pSrc*, Npp32s *nSrcStep*, Npp8u ∗ *pDst*, Npp32s *nDstStep*, NppiSize *oSizeROI*, NppiSize *oMaskSize*, NppiPoint *oAnchor*)

8-bit unsigned maximum filter.

Result pixel value is the maximum of pixel values under the rectangular mask region.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

oMaskSize Width and Height of the neighborhood region for the local Max operation.

oAnchor X and Y offsets of the kernel origin frame of reference w.r.t the source pixel.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.81 NppStatus nppiFilterMax_8u_C4R (const Npp8u ∗ *pSrc*, Npp32s *nSrcStep*, Npp8u ∗ *pDst*, Npp32s *nDstStep*, NppiSize *oSizeROI*, NppiSize *oMaskSize*, NppiPoint *oAnchor*)

4 channel 8-bit unsigned maximum filter.

Result pixel value is the maximum of pixel values under the rectangular mask region.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

oMaskSize Width and Height of the neighborhood region for the local Max operation.

oAnchor X and Y offsets of the kernel origin frame of reference w.r.t the source pixel.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.82 NppStatus nppiFilterMin_8u_C1R (const Npp8u ∗ *pSrc*, Npp32s *nSrcStep*, Npp8u ∗ *pDst*, Npp32s *nDstStep*, NppiSize *oSizeROI*, NppiSize *oMaskSize*, NppiPoint *oAnchor*)

8-bit unsigned minimum filter.

Result pixel value is the minimum of pixel values under the rectangular mask region.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

oMaskSize Width and Height of the neighborhood region for the local Max operation.

oAnchor X and Y offsets of the kernel origin frame of reference w.r.t the source pixel.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.83 NppStatus nppiFilterMin_8u_C4R (const Npp8u ∗ *pSrc*, Npp32s *nSrcStep*, Npp8u ∗ *pDst*, Npp32s *nDstStep*, NppiSize *oSizeROI*, NppiSize *oMaskSize*, NppiPoint *oAnchor*)

4 channel 8-bit unsigned minimum filter.

Result pixel value is the minimum of pixel values under the rectangular mask region.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

oMaskSize Width and Height of the neighborhood region for the local Max operation.

oAnchor X and Y offsets of the kernel origin frame of reference w.r.t the source pixel.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.84 NppStatus nppiFilterRow_8u_C1R (const Npp8u ∗ *pSrc*, Npp32s *nSrcStep*, Npp8u ∗ *pDst*, Npp32s *nDstStep*, NppiSize *oROI*, const Npp32s ∗ *pKernel*, Npp32s *nMaskSize*, Npp32s *nAnchor*, Npp32s *nDivisor*)

8-bit unsigned 1D (row) image convolution.

Apply general linear Row convolution filter, with rescaling, in a 1D mask region around each source pixel for 1-channel 8 bit/pixel images. Result pixel is equal to the sum of the products between the kernel coefficients (pKernel array) and corresponding neighboring row pixel values in the source image defined by iKernelDim and iAnchorX, divided by iDivisor.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oROI Region-of-Interest (ROI).

pKernel Pointer to the start address of the kernel coefficient array. Coefficients are expected to be stored in reverse order.

nMaskSize Length of the linear kernel array.

nAnchor X offset of the kernel origin frame of reference w.r.t the source pixel.

nDivisor The factor by which the convolved summation from the Filter operation should be divided. If equal to the sum of coefficients, this will keep the maximum result value within full scale.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.85 NppStatus nppiFilterRow_8u_C4R (const Npp8u ∗ *pSrc*, Npp32s *nSrcStep*, Npp8u ∗ *pDst*, Npp32s *nDstStep*, NppiSize *oROI*, const Npp32s ∗ *pKernel*, Npp32s *nMaskSize*, Npp32s *nAnchor*, Npp32s *nDivisor*)

4 channel 8-bit unsigned 1D (row) image convolution.

Apply general linear Row convolution filter, with rescaling, in a 1D mask region around each source pixel for 1-channel 8 bit/pixel images. Result pixel is equal to the sum of the products between the kernel coefficients (pKernel array) and corresponding neighboring row pixel values in the source image defined by iKernelDim and iAnchorX, divided by iDivisor.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oROI Region-of-Interest (ROI).

pKernel Pointer to the start address of the kernel coefficient array. Coefficients are expected to be stored in reverse order.

nMaskSize Length of the linear kernel array.

nAnchor X offset of the kernel origin frame of reference w.r.t the source pixel.

nDivisor The factor by which the convolved summation from the Filter operation should be divided. If equal to the sum of coefficients, this will keep the maximum result value within full scale.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.86 void nppiFree (void ∗ *pData*)

Free method for any 2D allocated memory.

This method should be used to free memory allocated with any of the nppiMalloc_<modifier> methods.

Parameters:

pData A pointer to memory allocated using nppiMalloc_<modifier>.

7.4.1.87 NppStatus nppiGetAffineBound (NppiRect *srcRoi*, double *bound*[2][2], const double *coeffs*[2][3])

Calculates bounding box of the affine transform projection of the given source rectangular ROI.

Parameters:

srcRoi Source ROI

bound Bounding box of the transformed source ROI

coeffs Affine transform coefficients

Returns:

Error codes:

- NPP_SIZE_ERROR Indicates an error condition if any image dimension has zero or negative value
- NPP_RECT_ERROR Indicates an error condition if width or height of the intersection of the srcRoi and source image is less than or equal to 1
- NPP_COEFF_ERROR Indicates an error condition if coefficient values are invalid

7.4.1.88 NppStatus nppiGetAffineQuad (NppiRect *srcRoi*, double *quad*[4][2], const double *coeffs*[2][3])

Calculates affine transform projection of given source rectangular ROI.

Parameters:

srcRoi Source ROI

quad Destination quadrangle

coeffs Affine transform coefficients

Returns:

Error codes:

- NPP_SIZE_ERROR Indicates an error condition if any image dimension has zero or negative value
- NPP_RECT_ERROR Indicates an error condition if width or height of the intersection of the srcRoi and source image is less than or equal to 1
- NPP_COEFF_ERROR Indicates an error condition if coefficient values are invalid

7.4.1.89 NppStatus nppiGetAffineTransform (NppiRect *srcRoi*, const double *quad*[4][2], double *coeffs*[2][3])

Calculates affine transform coefficients given source rectangular ROI and its destination quadrangle projection.

Parameters:

srcRoi Source ROI

quad Destination quadrangle

coeffs Affine transform coefficients

Returns:

Error codes:

- NPP_SIZE_ERROR Indicates an error condition if any image dimension has zero or negative value
- NPP_RECT_ERROR Indicates an error condition if width or height of the intersection of the srcRoi and source image is less than or equal to 1
- NPP_COEFF_ERROR Indicates an error condition if coefficient values are invalid
- NPP_AFFINE_QUAD_INCORRECT_WARNING Indicates a warning when quad does not conform to the transform properties. Fourth vertex is ignored, internally computed coordinates are used instead

7.4.1.90 NppStatus nppiGetPerspectiveBound (NppiRect *srcRoi*, double *bound*[2][2], const double *coeffs*[3][3])

Calculates bounding box of the perspective transform projection of the given source rectangular ROI.

Parameters:

srcRoi Source ROI

bound Bounding box of the transformed source ROI

coeffs Perspective transform coefficients

Returns:

Error codes:

- NPP_SIZE_ERROR Indicates an error condition if any image dimension has zero or negative value
- NPP_RECT_ERROR Indicates an error condition if width or height of the intersection of the srcRoi and source image is less than or equal to 1
- NPP_COEFF_ERROR Indicates an error condition if coefficient values are invalid

7.4.1.91 NppStatus nppiGetPerspectiveQuad (NppiRect *srcRoi*, double *quad*[4][2], const double *coeffs*[3][3])

Calculates perspective transform projection of given source rectangular ROI.

Parameters:

srcRoi Source ROI

quad Destination quadrangle

coeffs Perspective transform coefficients

Returns:

Error codes:

- NPP_SIZE_ERROR Indicates an error condition if any image dimension has zero or negative value
- NPP_RECT_ERROR Indicates an error condition if width or height of the intersection of the srcRoi and source image is less than or equal to 1
- NPP_COEFF_ERROR Indicates an error condition if coefficient values are invalid

7.4.1.92 NppStatus nppiGetPerspectiveTransform (NppiRect *srcRoi*, const double *quad*[4][2], double *coeffs*[3][3])

Calculates perspective transform coefficients given source rectangular ROI and its destination quadrangle projection.

Parameters:

srcRoi Source ROI

quad Destination quadrangle

coeffs Perspective transform coefficients

Returns:

Error codes:

- NPP_SIZE_ERROR Indicates an error condition if any image dimension has zero or negative value
- NPP_RECT_ERROR Indicates an error condition if width or height of the intersection of the srcRoi and source image is less than or equal to 1
- NPP_COEFF_ERROR Indicates an error condition if coefficient values are invalid

7.4.1.93 NppStatus nppiGraphcut_32s8u (Npp32s ∗ *pTerminals*, Npp32s ∗ *pLeftTransposed*, Npp32s ∗ *pRightTransposed*, Npp32s ∗ *pTop*, Npp32s ∗ *pBottom*, int *nStep*, int *nTransposedStep*, NppiSize *size*, Npp8u ∗ *pLabel*, int *nLabelStep*, Npp8u ∗ *pBuffer*)

Graphcut of a flow network (32bit signed integer edge capacities).

The function computes the minimal cut (graphcut) of a 2D regular 4-connected graph. The inputs are the capacities of the horizontal (in transposed form), vertical and terminal (source and sink) edges. The capacities to source and sink are stored as capacity differences in the terminals array (terminals(x) =

source(x) - sink(x)). The implementation assumes that the edge capacities for boundary edges that would connect to nodes outside the specified domain are set to 0 (for example left(0,∗) == 0). If this is not fulfilled the computed labeling may be wrong! The computed binary labeling is encoded as unsigned 8bit values (0 / 255).

See also:

nppiGraphcutGetSize

Parameters:

pTerminals Pointer to differences of terminal edge capacities (terminal(x) = source(x) - sink(x))

pLeftTransposed Pointer to transposed left edge capacities (left(0,∗) must be 0)

pRightTransposed Pointer to transposed right edge capacities (right(width-1,∗) must be 0)

pTop Pointer to top edge capacities (top(∗,0) must be 0)

pBottom Pointer to bottom edge capacities (bottom(∗,height-1) must be 0)

nStep Step in bytes between any pair of sequential rows of edge capacities

nTransposedStep Step in bytes between any pair of sequential rows of tranposed edge capacities

size Graph size

pLabel Pointer to destination label image

nLabelStep Step in bytes between any pair of sequential rows of label image

pBuffer Pointer to the temporary buffer

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.94 NppStatus nppiGraphcutGetSize (NppiSize *size*, int ∗ *pBufSize*)

Calculates the size of the temporary buffer for graph-cut labeling.

See also:

nppiGraphcut_32s8u

Parameters:

size Graph size

pBufSize Pointer to variable that returns the size of the temporary buffer.

Returns:

NPP_SUCCESS Indicates no error. Any other value indicates an error or a warning
NPP_SIZE_ERROR Indicates an error condition if any image dimension has zero or negative value
NPP_NULL_POINTER_ERROR Indicates an error condition if pBufSize pointer is NULL

7.4.1.95 NppStatus nppiHistogramEven_16s_AC4R (const Npp16s * *pSrc*, int *nSrcStep*, NppiSize *oSizeROI*, Npp32s * *pHist*[3], int *nLevels*[3], Npp32s *nLowerLevel*[3], Npp32s *nUpperLevel*[3], Npp8u * *pBuffer*)

4 channel (alpha as the last channel) 16-bit signed histogram with evenly distributed bins.

Alpha channel is ignored during histogram computation.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pHist Array of pointers which are receiving computed histograms per color channel. Array pointed by pHist[i] be of size nLevels[i]-1.

nLevels Array containing number of levels per color channel.

nLowerLevel Array containing lower-level of lowest bin per color channel.

nUpperLevel Array containing upper-level of highest bin per color channel.

pBuffer Pointer to appropriately sized (nppiHistogramEvenGetBufferSize_16s_AC4R) scratch buffer.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.96 NppStatus nppiHistogramEven_16s_C1R (const Npp16s * *pSrc*, int *nSrcStep*, NppiSize *oSizeROI*, Npp32s * *pHist*, int *nLevels*, Npp32s *nLowerLevel*, Npp32s *nUpperLevel*, Npp8u * *pBuffer*)

16-bit signed histogram with evenly distributed bins.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pHist Pointer to array that receives the computed histogram. The array must be of size nLevels-1.

nLevels Number of levels.

nLowerLevel Lower boundary of lowest level bin.

nUpperLevel Upper boundary of highest level bin.

pBuffer Pointer to appropriately sized (nppiHistogramEvenGetBufferSize_16s_C1R) scratch buffer.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.97 NppStatus nppiHistogramEven_16s_C4R (const Npp16s ∗ *pSrc*, int *nSrcStep*, NppiSize *oSizeROI*, Npp32s ∗ *pHist*[4], int *nLevels*[4], Npp32s *nLowerLevel*[4], Npp32s *nUpperLevel*[4], Npp8u ∗ *pBuffer*)

4 channel 16-bit signed histogram with evenly distributed bins.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pHist Array of pointers which are receiving computed histograms per color channel. Array pointed by pHist[i] be of size nLevels[i]-1.

nLevels Array containing number of levels per color channel.

nLowerLevel Array containing lower-level of lowest bin per color channel.

nUpperLevel Array containing upper-level of highest bin per color channel.

pBuffer Pointer to appropriately sized (nppiHistogramEvenGetBufferSize_16s_C4R) scratch buffer.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.98 NppStatus nppiHistogramEven_16u_AC4R (const Npp16u ∗ *pSrc*, int *nSrcStep*, NppiSize *oSizeROI*, Npp32s ∗ *pHist*[3], int *nLevels*[3], Npp32s *nLowerLevel*[3], Npp32s *nUpperLevel*[3], Npp8u ∗ *pBuffer*)

4 channel (alpha as the last channel) 16-bit unsigned histogram with evenly distributed bins.

Alpha channel is ignored during histogram computation.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pHist Array of pointers which are receiving computed histograms per color channel. Array pointed by pHist[i] be of size nLevels[i]-1.

nLevels Array containing number of levels per color channel.

nLowerLevel Array containing lower-level of lowest bin per color channel.

nUpperLevel Array containing upper-level of highest bin per color channel.

pBuffer Pointer to appropriately sized (nppiHistogramEvenGetBufferSize_16u_AC4R) scratch buffer.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.99 NppStatus nppiHistogramEven_16u_C1R (const Npp16u $*$ *pSrc*, int *nSrcStep*, NppiSize *oSizeROI*, Npp32s $*$ *pHist*, int *nLevels*, Npp32s *nLowerLevel*, Npp32s *nUpperLevel*, Npp8u $*$ *pBuffer*)

16-bit unsigned histogram with evenly distributed bins.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pHist Pointer to array that receives the computed histogram. The array must be of size nLevels-1.

nLevels Number of levels.

nLowerLevel Lower boundary of lowest level bin.

nUpperLevel Upper boundary of highest level bin.

pBuffer Pointer to appropriately sized (nppiHistogramEvenGetBufferSize_16u_C1R) scratch buffer.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.100 NppStatus nppiHistogramEven_16u_C4R (const Npp16u $*$ *pSrc*, int *nSrcStep*, NppiSize *oSizeROI*, Npp32s $*$ *pHist*[4], int *nLevels*[4], Npp32s *nLowerLevel*[4], Npp32s *nUpperLevel*[4], Npp8u $*$ *pBuffer*)

4 channel 16-bit unsigned histogram with evenly distributed bins.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pHist Array of pointers which are receiving computed histograms per color channel. Array pointed by pHist[i] be of size nLevels[i]-1.

nLevels Array containing number of levels per color channel.

nLowerLevel Array containing lower-level of lowest bin per color channel.

nUpperLevel Array containing upper-level of highest bin per color channel.

pBuffer Pointer to appropriately sized (nppiHistogramEvenGetBufferSize_16u_C4R) scratch buffer.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.101 NppStatus nppiHistogramEven_8u_AC4R (const Npp8u $*$ *pSrc*, int *nSrcStep*, NppiSize *oSizeROI*, Npp32s $*$ *pHist*[3], int *nLevels*[3], Npp32s *nLowerLevel*[3], Npp32s *nUpperLevel*[3], Npp8u $*$ *pBuffer*)

4 channel (alpha as the last channel) 8-bit unsigned histogram with evenly distributed bins.

Alpha channel is ignored during histogram computation.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pHist Array of pointers which are receiving computed histograms per color channel. Array pointed by pHist[i] be of size nLevels[i]-1.

nLevels Array containing number of levels per color channel.

nLowerLevel Array containing lower-level of lowest bin per color channel.

nUpperLevel Array containing upper-level of highest bin per color channel.

pBuffer Pointer to appropriately sized (nppiHistogramEvenGetBufferSize_8u_AC4R) scratch buffer.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.102 NppStatus nppiHistogramEven_8u_C1R (const Npp8u ∗ *pSrc*, int *nSrcStep*, NppiSize *oSizeROI*, Npp32s ∗ *pHist*, int *nLevels*, Npp32s *nLowerLevel*, Npp32s *nUpperLevel*, Npp8u ∗ *pBuffer*)

8-bit unsigned histogram with evenly distributed bins.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pHist Pointer to array that receives the computed histogram. The array must be of size nLevels-1.

nLevels Number of levels.

nLowerLevel Lower boundary of lowest level bin.

nUpperLevel Upper boundary of highest level bin.

pBuffer Pointer to appropriately sized (nppiHistogramEvenGetBufferSize_8u_C1R) scratch buffer.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.103 NppStatus nppiHistogramEven_8u_C4R (const Npp8u ∗ *pSrc*, int *nSrcStep*, NppiSize *oSizeROI*, Npp32s ∗ *pHist*[4], int *nLevels*[4], Npp32s *nLowerLevel*[4], Npp32s *nUpperLevel*[4], Npp8u ∗ *pBuffer*)

4 channel 8-bit unsigned histogram with evenly distributed bins.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pHist Array of pointers which are receiving computed histograms per color channel. Array pointed by pHist[i] be of size nLevels[i]-1.

nLevels Array containing number of levels per color channel.

nLowerLevel Array containing lower-level of lowest bin per color channel.

nUpperLevel Array containing upper-level of highest bin per color channel.

pBuffer Pointer to appropriately sized (nppiHistogramEvenGetBufferSize_8u_C4R) scratch buffer.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.104 NppStatus nppiHistogramEvenGetBufferSize_16s_AC4R (NppiSize *oSizeROI*, int *nLevels*[3], int ∗ *hpBufferSize*)

Scratch-buffer size for nppiHistogramEven_16s_AC4R.

Parameters:

oSizeROI ROI size.

nLevels Array containing number of levels per color channel.

hpBufferSize Host pointer where required buffer size is returned.

Returns:

Error Code.

7.4.1.105 NppStatus nppiHistogramEvenGetBufferSize_16s_C1R (NppiSize *oSizeROI*, int *nLevels*, int ∗ *hpBufferSize*)

Scratch-buffer size for nppiHistogramEven_16s_C1R.

Parameters:

oSizeROI Region-of-Interest (ROI).

nLevels Number of levels in the histogram.

hpBufferSize Host pointer where required buffer size is returned.

Returns:

Error Code.

7.4.1.106 NppStatus nppiHistogramEvenGetBufferSize_16s_C4R (NppiSize *oSizeROI*, int *nLevels*[4], int ∗ *hpBufferSize*)

Scratch-buffer size for nppiHistogramEven_16s_C4R.

Parameters:

oSizeROI ROI size.

nLevels Array containing number of levels per color channel.

hpBufferSize Host pointer where required buffer size is returned.

Returns:

Error Code.

7.4.1.107 NppStatus nppiHistogramEvenGetBufferSize_16u_AC4R (NppiSize *oSizeROI*, int *nLevels*[3], int ∗ *hpBufferSize*)

Scratch-buffer size for nppiHistogramEven_16u_AC4R.

Parameters:

oSizeROI ROI size.

nLevels Array containing number of levels per color channel.

hpBufferSize Host pointer where required buffer size is returned.

Returns:

Error Code.

7.4.1.108 NppStatus nppiHistogramEvenGetBufferSize_16u_C1R (NppiSize *oSizeROI*, int *nLevels*, int ∗ *hpBufferSize*)

Scratch-buffer size for nppiHistogramEven_16u_C1R.

Parameters:

oSizeROI Region-of-Interest (ROI).

nLevels Number of levels in the histogram.

hpBufferSize Host pointer where required buffer size is returned.

Returns:

Error Code.

7.4.1.109 NppStatus nppiHistogramEvenGetBufferSize_16u_C4R (NppiSize *oSizeROI*, int *nLevels*[4], int ∗ *hpBufferSize*)

Scratch-buffer size for nppiHistogramEven_16u_C4R.

Parameters:

oSizeROI ROI size.

nLevels Array containing number of levels per color channel.

hpBufferSize Host pointer where required buffer size is returned.

Returns:

Error Code.

7.4.1.110 NppStatus nppiHistogramEvenGetBufferSize_8u_AC4R (NppiSize *oSizeROI*, int *nLevels*[3], int ∗ *hpBufferSize*)

Scratch-buffer size for nppiHistogramEven_8u_AC4R.

Parameters:

oSizeROI ROI size.

nLevels Array containing number of levels per color channel.

hpBufferSize Host pointer where required buffer size is returned.

Returns:

Error Code.

7.4.1.111 NppStatus nppiHistogramEvenGetBufferSize_8u_C1R (NppiSize *oSizeROI*, int *nLevels*, int ∗ *hpBufferSize*)

Scratch-buffer size for nppiHistogramEven_8u_C1R.

Parameters:

oSizeROI Region-of-Interest (ROI).

nLevels Number of levels in the histogram.

hpBufferSize Host pointer where required buffer size is returned.

Returns:

Error Code.

7.4.1.112 NppStatus nppiHistogramEvenGetBufferSize_8u_C4R (NppiSize *oSizeROI*, int *nLevels*[4], int ∗ *hpBufferSize*)

Scratch-buffer size for nppiHistogramEven_8u_C4R.

Parameters:

oSizeROI ROI size.

nLevels Array containing number of levels per color channel.

hpBufferSize Host pointer where required buffer size is returned.

Returns:

Error Code.

7.4.1.113 NppStatus nppiHistogramRange_16s_AC4R (const Npp16s ∗ *pSrc*, int *nSrcStep*, NppiSize *oSizeROI*, Npp32s ∗ *pHist*[3], const Npp32s ∗ *pLevels*[3], int *nLevels*[3], Npp8u ∗ *pBuffer*)

4 channel (alpha as a last channel) 16-bit signed histogram with bins determined by pLevels.

Alpha channel is ignored during the histograms computations.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pHist Array of pointers which are receiving the computed histograms per color channel. Array pointed by pHist[i] must be of size nLevels[i]-1.

nLevels Array containing number of levels per color channel.

pLevels Array containing pointers to level-arrays per color channel. Array pointed by pLevel[i] must be of size nLevels[i].

pBuffer Pointer to appropriately sized (nppiHistogramRangeGetBufferSize_16_AC4R) scratch buffer.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.114 NppStatus nppiHistogramRange_16s_C1R (const Npp16s * *pSrc*, int *nSrcStep*, NppiSize *oSizeROI*, Npp32s * *pHist*, const Npp32s * *pLevels*, int *nLevels*, Npp8u * *pBuffer*)

16-bit signed histogram with bins determined by pLevels array.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pHist Pointer to array that receives the computed histogram. The array must be of size nLevels-1.

pLevels Pointer to array containing the level sizes of the bins. The array must be of size nLevels.

nLevels Number of levels in histogram.

pBuffer Pointer to appropriately sized (nppiHistogramRangeGetBufferSize_16_C1R) scratch buffer.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.115 NppStatus nppiHistogramRange_16s_C4R (const Npp16s * *pSrc*, int *nSrcStep*, NppiSize *oSizeROI*, Npp32s * *pHist*[4], const Npp32s * *pLevels*[4], int *nLevels*[4], Npp8u * *pBuffer*)

4 channel 16-bit signed histogram with bins determined by pLevels.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pHist Array of pointers which are receiving the computed histograms per color channel. Array pointed by pHist[i] must be of size nLevels[i]-1.

nLevels Array containing number of levels per color channel.

pLevels Array containing pointers to level-arrays per color channel. Array pointed by pLevel[i] must be of size nLevels[i].

pBuffer Pointer to appropriately sized (nppiHistogramRangeGetBufferSize_16s_C4R) scratch buffer.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.116 NppStatus nppiHistogramRange_16u_AC4R (const Npp16u ∗ *pSrc*, int *nSrcStep*, NppiSize *oSizeROI*, Npp32s ∗ *pHist*[3], const Npp32s ∗ *pLevels*[3], int *nLevels*[3], Npp8u ∗ *pBuffer*)

4 channel (alpha as a last channel) 16-bit unsigned histogram with bins determined by pLevels.

Alpha channel is ignored during the histograms computations.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pHist Array of pointers which are receiving the computed histograms per color channel. Array pointed by pHist[i] must be of size nLevels[i]-1.

nLevels Array containing number of levels per color channel.

pLevels Array containing pointers to level-arrays per color channel. Array pointed by pLevel[i] must be of size nLevels[i].

pBuffer Pointer to appropriately sized (nppiHistogramRangeGetBufferSize_16u_AC4R) scratch buffer.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.117 NppStatus nppiHistogramRange_16u_C1R (const Npp16u ∗ *pSrc*, int *nSrcStep*, NppiSize *oSizeROI*, Npp32s ∗ *pHist*, const Npp32s ∗ *pLevels*, int *nLevels*, Npp8u ∗ *pBuffer*)

16-bit unsigned histogram with bins determined by pLevels array.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pHist Pointer to array that receives the computed histogram. The array must be of size nLevels-1.

pLevels Pointer to array containing the level sizes of the bins. The array must be of size nLevels.

nLevels Number of levels in histogram.

pBuffer Pointer to appropriately sized (nppiHistogramRangeGetBufferSize_16u_C1R) scratch buffer.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.118 NppStatus nppiHistogramRange_16u_C4R (const Npp16u ∗ *pSrc*, int *nSrcStep*, NppiSize *oSizeROI*, Npp32s ∗ *pHist*[4], const Npp32s ∗ *pLevels*[4], int *nLevels*[4], Npp8u ∗ *pBuffer*)

4 channel 16-bit unsigned histogram with bins determined by pLevels.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pHist Array of pointers which are receiving the computed histograms per color channel. Array pointed by pHist[i] must be of size nLevels[i]-1.

nLevels Array containing number of levels per color channel.

pLevels Array containing pointers to level-arrays per color channel. Array pointed by pLevel[i] must be of size nLevels[i].

pBuffer Pointer to appropriately sized (nppiHistogramRangeGetBufferSize_16u_C4R) scratch buffer.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.119 NppStatus nppiHistogramRange_32f_AC4R (const Npp32f ∗ *pSrc*, int *nSrcStep*, NppiSize *oSizeROI*, Npp32s ∗ *pHist*[3], const Npp32f ∗ *pLevels*[3], int *nLevels*[3], Npp8u ∗ *pBuffer*)

4 channel (alpha as a last channel) 32-bit float histogram with bins determined by pLevels.

Alpha channel is ignored during the histograms computations.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pHist Array of pointers which are receiving the computed histograms per color channel. Array pointed by pHist[i] must be of size nLevels[i]-1.

nLevels Array containing number of levels per color channel.

pLevels Array containing pointers to level-arrays per color channel. Array pointed by pLevel[i] must be of size nLevels[i].

pBuffer Pointer to appropriately sized (nppiHistogramRangeGetBufferSize_32f_AC4R) scratch buffer.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.120 NppStatus nppiHistogramRange_32f_C1R (const Npp32f * *pSrc*, int *nSrcStep*, NppiSize *oSizeROI*, Npp32s * *pHist*, const Npp32f * *pLevels*, int *nLevels*, Npp8u * *pBuffer*)

32-bit float histogram with bins determined by pLevels array.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pHist Pointer to array that receives the computed histogram. The array must be of size nLevels-1.

pLevels Pointer to array containing the level sizes of the bins. The array must be of size nLevels.

nLevels Number of levels in histogram.

pBuffer Pointer to appropriately sized (nppiHistogramRangeGetBufferSize_32f_C1R) scratch buffer.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.121 NppStatus nppiHistogramRange_32f_C4R (const Npp32f * *pSrc*, int *nSrcStep*, NppiSize *oSizeROI*, Npp32s * *pHist*[4], const Npp32f * *pLevels*[4], int *nLevels*[4], Npp8u * *pBuffer*)

4 channel 32-bit float histogram with bins determined by pLevels.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pHist Array of pointers which are receiving the computed histograms per color channel. Array pointed by pHist[i] must be of size nLevels[i]-1.

nLevels Array containing number of levels per color channel.

pLevels Array containing pointers to level-arrays per color channel. Array pointed by pLevel[i] must be of size nLevels[i].

pBuffer Pointer to appropriately sized (nppiHistogramRangeGetBufferSize_32f_C4R) scratch buffer.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.122 NppStatus nppiHistogramRange_8u_AC4R (const Npp8u * *pSrc*, int *nSrcStep*, NppiSize *oSizeROI*, Npp32s * *pHist*[3], const Npp32s * *pLevels*[3], int *nLevels*[3], Npp8u * *pBuffer*)

4 channel (alpha as a last channel) 8-bit unsigned histogram with bins determined by pLevels.

Alpha channel is ignored during the histograms computations.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pHist Array of pointers which are receiving the computed histograms per color channel. Array pointed by pHist[i] must be of size nLevels[i]-1.

nLevels Array containing number of levels per color channel.

pLevels Array containing pointers to level-arrays per color channel. Array pointed by pLevel[i] must be of size nLevels[i].

pBuffer Pointer to appropriately sized (nppiHistogramRangeGetBufferSize_8u_AC4R) scratch buffer.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.123 NppStatus nppiHistogramRange_8u_C1R (const Npp8u $*$ *pSrc*, int *nSrcStep*, NppiSize *oSizeROI*, Npp32s $*$ *pHist*, const Npp32s $*$ *pLevels*, int *nLevels*, Npp8u $*$ *pBuffer*)

8-bit unsigned histogram with bins determined by pLevels array.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pHist Pointer to array that receives the computed histogram. The array must be of size nLevels-1.

pLevels Pointer to array containing the level sizes of the bins. The array must be of size nLevels.

nLevels Number of levels in histogram.

pBuffer Pointer to appropriately sized (nppiHistogramRangeGetBufferSize_8u_C1R) scratch buffer.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.124 NppStatus nppiHistogramRange_8u_C4R (const Npp8u $*$ *pSrc*, int *nSrcStep*, NppiSize *oSizeROI*, Npp32s $*$ *pHist*[4], const Npp32s $*$ *pLevels*[4], int *nLevels*[4], Npp8u $*$ *pBuffer*)

4 channel 8-bit unsigned histogram with bins determined by pLevels.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pHist Array of pointers which are receiving the computed histograms per color channel. Array pointed by pHist[i] must be of size nLevels[i]-1.

nLevels Array containing number of levels per color channel.

pLevels Array containing pointers to level-arrays per color channel. Array pointed by pLevel[i] must be of size nLevels[i].

pBuffer Pointer to appropriately sized (nppiHistogramRangeGetBufferSize_8u_C4R) scratch buffer.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.125 NppStatus nppiHistogramRangeGetBufferSize_16s_AC4R (NppiSize *oSizeROI*, int *nLevels*[3], int * *hpBufferSize*)

Scratch-buffer size for nppiHistogramRange_16s_AC4R.

Parameters:

oSizeROI ROI size.

nLevels Array containing number of levels per color channel.

hpBufferSize Host pointer where required buffer size is returned.

Returns:

Error Code.

7.4.1.126 NppStatus nppiHistogramRangeGetBufferSize_16s_C1R (NppiSize *oSizeROI*, int *nLevels*, int * *hpBufferSize*)

Scratch-buffer size for nppiHistogramRange_16s_C1R.

Parameters:

oSizeROI Region-of-Interest (ROI).

nLevels Number of levels in the histogram.

hpBufferSize Host pointer where required buffer size is returned.

Returns:

Error Code.

7.4.1.127 NppStatus nppiHistogramRangeGetBufferSize_16s_C4R (NppiSize *oSizeROI*, int *nLevels*[4], int * *hpBufferSize*)

Scratch-buffer size for nppiHistogramRange_16s_C4R.

Parameters:

oSizeROI ROI size.

nLevels Array containing number of levels per color channel.

hpBufferSize Host pointer where required buffer size is returned.

Returns:

Error Code.

7.4.1.128 NppStatus nppiHistogramRangeGetBufferSize_16u_AC4R (NppiSize *oSizeROI*, int *nLevels*[3], int ∗ *hpBufferSize*)

Scratch-buffer size for nppiHistogramRange_16u_AC4R.

Parameters:

oSizeROI ROI size.

nLevels Array containing number of levels per color channel.

hpBufferSize Host pointer where required buffer size is returned.

Returns:

Error Code.

7.4.1.129 NppStatus nppiHistogramRangeGetBufferSize_16u_C1R (NppiSize *oSizeROI*, int *nLevels*, int ∗ *hpBufferSize*)

Scratch-buffer size for nppiHistogramRange_16u_C1R.

Parameters:

oSizeROI Region-of-Interest (ROI).

nLevels Number of levels in the histogram.

hpBufferSize Host pointer where required buffer size is returned.

Returns:

Error Code.

7.4.1.130 NppStatus nppiHistogramRangeGetBufferSize_16u_C4R (NppiSize *oSizeROI*, int *nLevels*[4], int ∗ *hpBufferSize*)

Scratch-buffer size for nppiHistogramRange_16u_C4R.

Parameters:

oSizeROI ROI size.

nLevels Array containing number of levels per color channel.

hpBufferSize Host pointer where required buffer size is returned.

Returns:

Error Code.

7.4.1.131 NppStatus nppiHistogramRangeGetBufferSize_32f_AC4R (NppiSize *oSizeROI*, int *nLevels*[3], int ∗ *hpBufferSize*)

Scratch-buffer size for nppiHistogramRange_32f_AC4R.

Parameters:

oSizeROI ROI size.

nLevels Array containing number of levels per color channel.

hpBufferSize Host pointer where required buffer size is returned.

Returns:

Error Code.

7.4.1.132 NppStatus nppiHistogramRangeGetBufferSize_32f_C1R (NppiSize *oSizeROI*, int *nLevels*, int ∗ *hpBufferSize*)

Scratch-buffer size for nppiHistogramRange_32f_C1R.

Parameters:

oSizeROI Region-of-Interest (ROI).

nLevels Number of levels in the histogram.

hpBufferSize Host pointer where required buffer size is returned.

Returns:

Error Code.

7.4.1.133 NppStatus nppiHistogramRangeGetBufferSize_32f_C4R (NppiSize *oSizeROI*, int *nLevels*[4], int ∗ *hpBufferSize*)

Scratch-buffer size for nppiHistogramRange_32f_C4R.

Parameters:

oSizeROI ROI size.

nLevels Array containing number of levels per color channel.

hpBufferSize Host pointer where required buffer size is returned.

Returns:

Error Code.

7.4.1.134 NppStatus nppiHistogramRangeGetBufferSize_8u_AC4R (NppiSize *oSizeROI*, int *nLevels*[3], int ∗ *hpBufferSize*)

Scratch-buffer size for nppiHistogramRange_8u_AC4R.

Parameters:

oSizeROI ROI size.

nLevels Array containing number of levels per color channel.

hpBufferSize Host pointer where required buffer size is returned.

Returns:

Error Code.

7.4.1.135 NppStatus nppiHistogramRangeGetBufferSize_8u_C1R (NppiSize *oSizeROI*, int *nLevels*, int ∗ *hpBufferSize*)

Scratch-buffer size for nppiHistogramRange_8u_C1R.

Parameters:

oSizeROI Region-of-Interest (ROI).

nLevels Number of levels in the histogram.

hpBufferSize Host pointer where required buffer size is returned.

Returns:

Error Code.

7.4.1.136 NppStatus nppiHistogramRangeGetBufferSize_8u_C4R (NppiSize *oSizeROI*, int *nLevels*[4], int ∗ *hpBufferSize*)

Scratch-buffer size for nppiHistogramRange_8u_C4R.

Parameters:

oSizeROI ROI size.

nLevels Array containing number of levels per color channel.

hpBufferSize Host pointer where required buffer size is returned.

Returns:

Error Code.

7.4.1.137 NppStatus nppiLn_32f_C1R (const Npp32f ∗ *pSrc*, int *nSrcStep*, Npp32f ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

32-bit floating point logarithm.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.138 NppStatus nppiLUT_Linear_8u_AC4R (const Npp8u ∗ *pSrc*, int *nSrcStep*, Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*, const Npp32s ∗ *pValues*[4], const Npp32s ∗ *pLevels*[4], int *nLevels*[4])

4 channel 8-bit unsigned look-up-table color conversion, not affecting Alpha.

The LUT is derived from a set of user defined mapping points through linear interpolation. Alpha channel is the last channel and is not processed.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pValues Double pointer to an [4] of arrays of user defined OUTPUT values per CHANNEL

pLevels Double pointer to an [4] of arrays of user defined INPUT values per CHANNEL

nLevels A [4] array of user defined input/output mapping points (levels) per CHANNEL

Returns:

Image Data Related Error Codes, ROI Related Error Codes

- NPP_LUT_NUMBER_OF_LEVELS_ERROR if the number of levels is less than 2.

7.4.1.139 NppStatus nppiLUT_Linear_8u_C1R (const Npp8u ∗ *pSrc*, int *nSrcStep*, Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*, const Npp32s ∗ *pValues*, const Npp32s ∗ *pLevels*, int *nLevels*)

8-bit unsigned look-up-table color conversion.

The LUT is derived from a set of user defined mapping points through linear interpolation.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pValues Pointer to an array of user defined OUTPUT values

pLevels Pointer to an array of user defined INPUT values

nLevels Number of user defined input/output mapping points (levels)

Returns:

Image Data Related Error Codes, ROI Related Error Codes

- NPP_LUT_NUMBER_OF_LEVELS_ERROR if the number of levels is less than 2.

7.4.1.140 NppStatus nppiLUT_Linear_8u_C3R (const Npp8u ∗ *pSrc*, int *nSrcStep*, Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*, const Npp32s ∗ *pValues*[3], const Npp32s ∗ *pLevels*[3], int *nLevels*[3])

3 channel 8-bit unsigned look-up-table color conversion.

The LUT is derived from a set of user defined mapping points through linear interpolation.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pValues Double pointer to an [3] of arrays of user defined OUTPUT values per CHANNEL

pLevels Double pointer to an [3] of arrays of user defined INPUT values per CHANNEL

nLevels A [3] array of user defined input/output mapping points (levels) per CHANNEL

Returns:

Image Data Related Error Codes, ROI Related Error Codes

- NPP_LUT_NUMBER_OF_LEVELS_ERROR if the number of levels is less than 2.

7.4.1.141 NppStatus nppiMagnitude_32fc32f_C1R (const Npp32fc ∗ *pSrc*, int *nSrcStep*, Npp32f ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

32-bit floating point complex to 32-bit floating point magnitude.

Converts complex-number pixel image to single channel image computing the result pixels as the magnitude of the complex values.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.142 NppStatus nppiMagnitudeSqr_32fc32f_C1R (const Npp32fc ∗ *pSrc*, int *nSrcStep*, Npp32f ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

32-bit floating point complex to 32-bit floating point squared magnitude.

Converts complex-number pixel image to single channel image computing the result pixels as the squared magnitude of the complex values.

The squared magnitude is an itermediate result of magnitude computation and can thus be computed faster than actual magnitude. If magnitudes are required for sorting/comparing only, using this function instead of nppiMagnitude_32fc32f_C1R can be a worthwhile performance optimization.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.143 Npp16s∗ nppiMalloc_16s_C1 (int *nWidthPixels*, int *nHeightPixels*, int ∗ *pStepBytes*)

16-bit signed image memory allocator.

Parameters:

nWidthPixels Image width.

nHeightPixels Image height.

pStepBytes Line Step.

Returns:

Pointer to new image data. NULL pointer indicates allocation failure.

7.4.1.144 Npp16s∗ nppiMalloc_16s_C2 (int *nWidthPixels*, int *nHeightPixels*, int ∗ *pStepBytes*)

2 channel 16-bit signed image memory allocator.

Parameters:

nWidthPixels Image width.

nHeightPixels Image height.

pStepBytes Line Step.

Returns:

Pointer to new image data. NULL pointer indicates allocation failure.

7.4.1.145 Npp16s∗ nppiMalloc_16s_C4 (int *nWidthPixels*, int *nHeightPixels*, int ∗ *pStepBytes*)

4 channel 16-bit signed image memory allocator.

Parameters:

nWidthPixels Image width.

nHeightPixels Image height.

pStepBytes Line Step.

Returns:

Pointer to new image data. NULL pointer indicates allocation failure.

7.4.1.146 Npp16u∗ nppiMalloc_16u_C1 (int *nWidthPixels*, int *nHeightPixels*, int ∗ *pStepBytes*)

16-bit unsigned image memory allocator.

Parameters:

nWidthPixels Image width.

nHeightPixels Image height.

pStepBytes Line Step.

Returns:

Pointer to new image data. NULL pointer indicates allocation failure.

7.4.1.147 Npp16u∗ nppiMalloc_16u_C2 (int *nWidthPixels*, int *nHeightPixels*, int ∗ *pStepBytes*)

2 channel 16-bit unsigned image memory allocator.

Parameters:

nWidthPixels Image width.

nHeightPixels Image height.

pStepBytes Line Step.

Returns:

Pointer to new image data. NULL pointer indicates allocation failure.

7.4.1.148 Npp16u∗ nppiMalloc_16u_C3 (int *nWidthPixels*, int *nHeightPixels*, int ∗ *pStepBytes*)

3 channel 16-bit unsigned image memory allocator.

Parameters:

nWidthPixels Image width.

nHeightPixels Image height.

pStepBytes Line Step.

Returns:

Pointer to new image data. NULL pointer indicates allocation failure.

7.4.1.149 Npp16u∗ nppiMalloc_16u_C4 (int *nWidthPixels*, int *nHeightPixels*, int ∗ *pStepBytes*)

4 channel 16-bit unsigned image memory allocator.

Parameters:

nWidthPixels Image width.

nHeightPixels Image height.

pStepBytes Line Step.

Returns:

Pointer to new image data. NULL pointer indicates allocation failure.

7.4.1.150 Npp32f∗ nppiMalloc_32f_C1 (int *nWidthPixels*, int *nHeightPixels*, int ∗ *pStepBytes*)

32-bit floating point image memory allocator.

Parameters:

nWidthPixels Image width.

nHeightPixels Image height.

pStepBytes Line Step.

Returns:

Pointer to new image data. NULL pointer indicates allocation failure.

7.4.1.151 Npp32f∗ nppiMalloc_32f_C2 (int *nWidthPixels*, int *nHeightPixels*, int ∗ *pStepBytes*)

2 channel 32-bit floating point image memory allocator.

Parameters:

nWidthPixels Image width.

nHeightPixels Image height.

pStepBytes Line Step.

Returns:

Pointer to new image data. NULL pointer indicates allocation failure.

7.4.1.152 Npp32f∗ nppiMalloc_32f_C3 (int *nWidthPixels*, int *nHeightPixels*, int ∗ *pStepBytes*)

3 channel 32-bit floating point image memory allocator.

Parameters:

nWidthPixels Image width.

nHeightPixels Image height.

pStepBytes Line Step.

Returns:

Pointer to new image data. NULL pointer indicates allocation failure.

7.4.1.153 Npp32f∗ nppiMalloc_32f_C4 (int *nWidthPixels*, int *nHeightPixels*, int ∗ *pStepBytes*)

4 channel 32-bit floating point image memory allocator.

Parameters:

nWidthPixels Image width.

nHeightPixels Image height.

pStepBytes Line Step.

Returns:

Pointer to new image data. NULL pointer indicates allocation failure.

7.4.1.154 Npp32s∗ nppiMalloc_32s_C1 (int *nWidthPixels*, int *nHeightPixels*, int ∗ *pStepBytes*)

32-bit signed image memory allocator.

Parameters:

nWidthPixels Image width.

nHeightPixels Image height.

pStepBytes Line Step.

Returns:

Pointer to new image data. NULL pointer indicates allocation failure.

7.4.1.155 Npp32s∗ nppiMalloc_32s_C3 (int *nWidthPixels*, int *nHeightPixels*, int ∗ *pStepBytes*)

3 channel 32-bit signed image memory allocator.

Parameters:

nWidthPixels Image width.

nHeightPixels Image height.

pStepBytes Line Step.

Returns:

Pointer to new image data. NULL pointer indicates allocation failure.

7.4.1.156 Npp32s∗ nppiMalloc_32s_C4 (int *nWidthPixels*, int *nHeightPixels*, int ∗ *pStepBytes*)

4 channel 32-bit signed image memory allocator.

Parameters:

nWidthPixels Image width.

nHeightPixels Image height.

pStepBytes Line Step.

Returns:

Pointer to new image data. NULL pointer indicates allocation failure.

7.4.1.157 Npp8u∗ nppiMalloc_8u_C1 (int *nWidthPixels*, int *nHeightPixels*, int ∗ *pStepBytes*)

8-bit unsigned image memory allocator.

Parameters:

nWidthPixels Image width.

nHeightPixels Image height.

pStepBytes Line Step.

Returns:

Pointer to new image data. NULL pointer indicates allocation failure.

7.4.1.158 Npp8u∗ nppiMalloc_8u_C2 (int *nWidthPixels*, int *nHeightPixels*, int ∗ *pStepBytes*)

2 channel 8-bit unsigned image memory allocator.

Parameters:

nWidthPixels Image width.

nHeightPixels Image height.

pStepBytes Line Step.

Returns:

Pointer to new image data. NULL pointer indicates allocation failure.

7.4.1.159 Npp8u∗ nppiMalloc_8u_C3 (int *nWidthPixels*, int *nHeightPixels*, int ∗ *pStepBytes*)

3 channel 8-bit unsigned image memory allocator.

Parameters:

nWidthPixels Image width.

nHeightPixels Image height.

pStepBytes Line Step.

Returns:

Pointer to new image data. NULL pointer indicates allocation failure.

7.4.1.160 Npp8u∗ nppiMalloc_8u_C4 (int *nWidthPixels*, int *nHeightPixels*, int ∗ *pStepBytes*)

4 channel 8-bit unsigned image memory allocator.

Parameters:

nWidthPixels Image width.

nHeightPixels Image height.

pStepBytes Line Step.

Returns:

Pointer to new image data. NULL pointer indicates allocation failure.

7.4.1.161 NppStatus nppiMean_StdDev_8u_C1R (const Npp8u ∗ *pSrc*, int *nSrcStep*, NppiSize *oSizeROI*, Npp64f ∗ *pMean*, Npp64f ∗ *pStdDev*)

8-bit unsigned mean standard deviation.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pMean Contains computed mean. This is a host pointer.

pStdDev Contains computed standard deviation. This is a host pointer.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.162 NppStatus nppiMinMax_8u_C1R (const Npp8u ∗ *pSrc*, int *nSrcStep*, NppiSize *oSizeROI*, Npp8u ∗ *pMin*, Npp8u ∗ *pMax*, Npp8u ∗ *pDeviceBuffer*)

8-bit unsigned pixel minimum and maximum.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pMin Device-memory pointer receiving the minimum result.

pMax Device-memory pointer receiving the maximum result.

pDeviceBuffer Buffer to a scratch memory. Use nppiMinMaxGetBufferSize_8u_C1R to determine the minium number of bytes required.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.163 NppStatus nppiMinMax_8u_C4R (const Npp8u * *pSrc*, int *nSrcStep*, NppiSize *oSizeROI*, Npp8u *aMin*[4], Npp8u *aMax*[4], Npp8u * *pDeviceBuffer*)

4 channel 8-bit unsigned pixel minimum and maximum.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

oSizeROI Region-of-Interest (ROI).

aMin Device-pointer (array) receiving the minimum result.

aMax Device-pointer (array) receiving the maximum result.

pDeviceBuffer Buffer to a scratch memory. Use nppiMinMaxGetBufferSize_8u_C4R to determine the minium number of bytes required.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

Note:

Unlike nppiMinMax_8u_C1R, this primitive returns its results as device pointers.

7.4.1.164 NppStatus nppiMinMaxGetBufferSize_8u_C1R (const NppiSize & *oSizeROI*, int * *hpBufferSize*)

Scratch-buffer size for nppiMinManx_8u_C1R.

Parameters:

oSizeROI ROI size.

hpBufferSize Host pointer where required buffer size is returned.

Returns:

Error Code.

7.4.1.165 NppStatus nppiMinMaxGetBufferSize_8u_C4R (const NppiSize & *oSizeROI*, int * *hpBufferSize*)

Scratch-buffer size for nppiMinManx_8u_C4R.

Parameters:

oSizeROI ROI size.

hpBufferSize Host pointer where required buffer size is returned.

Returns:

Error Code.

7.4.1.166 NppStatus nppiMirror_8u_C1R (const Npp8u ∗ *pSrc*, int *nSrcStep*, Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *oROI*, NppiAxis *flip*)

8-bit unsigned image mirror.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oROI Region-of-Interest (ROI).

flip Specifies the axis about which the image is to be mirrored.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

- NPP_MIRROR_FLIP_ERR if flip has an illegal value.

7.4.1.167 NppStatus nppiMirror_8u_C4R (const Npp8u ∗ *pSrc*, int *nSrcStep*, Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *oROI*, NppiAxis *flip*)

4 channel 8-bit unsigned image mirror.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Distance in bytes between starts of consecutive lines of the destination image.

oROI Region-of-Interest (ROI).

flip Specifies the axis about which the image is to be mirrored.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

- NPP_MIRROR_FLIP_ERR if flip has an illegal value.

7.4.1.168 NppStatus nppiMul_32f_C1R (const Npp32f ∗ *pSrc1*, int *nSrc1Step*, const Npp32f ∗ *pSrc2*, int *nSrc2Step*, Npp32f ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

4 channel 32-bit floating point image multiplication.

Multiply corresponding pixels in ROI.

Parameters:

pSrc1 Source-Image Pointer.

nSrc1Step Source-Image Line Step.

pSrc2 Source-Image Pointer.

nSrc2Step Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.169 NppStatus nppiMul_32s_C1R (const Npp32s ∗ *pSrc1*, int *nSrc1Step*, const Npp32s ∗ *pSrc2*, int *nSrc2Step*, Npp32s ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

4 channel 32-bit image multiplication.

Multiply corresponding pixels in ROI.

Parameters:

pSrc1 Source-Image Pointer.

nSrc1Step Source-Image Line Step.

pSrc2 Source-Image Pointer.

nSrc2Step Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.170 NppStatus nppiMul_8u_AC4RSfs (const Npp8u ∗ *pSrc1*, int *nSrc1Step*, const Npp8u ∗ *pSrc2*, int *nSrc2Step*, Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*, int *nScaleFactor*)

4 channel 8-bit unsigned image multiplication, not affecting Alpha.

Multiply corresponding pixels in ROI.

Parameters:

pSrc1 Source-Image Pointer.

nSrc1Step Source-Image Line Step.

pSrc2 Source-Image Pointer.

nSrc2Step Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

nScaleFactor Result pixel values are scaled by 2^(-nScaleFactor) and then clamped to [0,255] range.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.171 NppStatus nppiMul_8u_C1RSfs (const Npp8u $*$ *pSrc1*, int *nSrc1Step*, const Npp8u $*$ *pSrc2*, int *nSrc2Step*, Npp8u $*$ *pDst*, int *nDstStep*, NppiSize *oSizeROI*, int *nScaleFactor*)

8-bit unsigned image multiplication.

Multiply the pixel values of corresponding pixels in the ROI and write them to the output image.

Parameters:

pSrc1 Source-Image Pointer.

nSrc1Step Source-Image Line Step.

pSrc2 Source-Image Pointer.

nSrc2Step Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

nScaleFactor Result pixel values are scaled by 2^(-nScaleFactor) and then clamped to [0,255] range.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.172 NppStatus nppiMul_8u_C4RSfs (const Npp8u $*$ *pSrc1*, int *nSrc1Step*, const Npp8u $*$ *pSrc2*, int *nSrc2Step*, Npp8u $*$ *pDst*, int *nDstStep*, NppiSize *oSizeROI*, int *nScaleFactor*)

4 channel 8-bit unsigned image multiplication.

Multiply corresponding pixels in ROI.

Parameters:

pSrc1 Source-Image Pointer.

nSrc1Step Source-Image Line Step.

pSrc2 Source-Image Pointer.

nSrc2Step Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

nScaleFactor Result pixel values are scaled by 2^(-nScaleFactor) and then clamped to [0,255] range.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.173 NppStatus nppiMulC_32f_C1R (const Npp32f ∗ *pSrc*, int *nSrcStep*, Npp32f *nValue*, Npp32f ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

32-bit floating point image multiply constant.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

nValue Constant.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.174 NppStatus nppiMulC_32fc_C1R (const Npp32fc ∗ *pSrc*, int *nSrcStep*, Npp32fc *nValue*, Npp32fc ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

32-bit complex floating point image multiply constant.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

nValue Constant.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.175 NppStatus nppiNormDiff_Inf_8u_C1R (const Npp8u ∗ *pSrc1*, int *nSrcStep1*, const Npp8u ∗ *pSrc2*, int *nSrcStep2*, NppiSize *oSizeROI*, Npp64f ∗ *pRetVal*)

8-bit unsigned Infinity Norm of pixel differences.

Parameters:

pSrc1 Source-Image Pointer.
nSrcStep1 Source-Image Line Step.
pSrc2 Source-Image Pointer.
nSrcStep2 Source-Image Line Step.
oSizeROI Region-of-Interest (ROI).
∗*pRetVal* Contains computed L1-norm of differences. This is a host pointer.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.176 NppStatus nppiNormDiff_L1_8u_C1R (const Npp8u ∗ *pSrc1*, int *nSrcStep1*, const Npp8u ∗ *pSrc2*, int *nSrcStep2*, NppiSize *oSizeROI*, Npp64f ∗ *pRetVal*)

8-bit unsigned L1 norm of pixel differences.

Parameters:

pSrc1 Source-Image Pointer.
nSrcStep1 Source-Image Line Step.
pSrc2 Source-Image Pointer.
nSrcStep2 Source-Image Line Step.
oSizeROI Region-of-Interest (ROI).
pRetVal Contains computed L1-norm of differences. This is a host pointer.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.177 NppStatus nppiNormDiff_L2_8u_C1R (const Npp8u ∗ *pSrc1*, int *nSrcStep1*, const Npp8u ∗ *pSrc2*, int *nSrcStep2*, NppiSize *oSizeROI*, Npp64f ∗ *pRetVal*)

8-bit unsigned L2 norm of pixel differences.

Parameters:

pSrc1 Source-Image Pointer.
nSrcStep1 Source-Image Line Step.
pSrc2 Source-Image Pointer.
nSrcStep2 Source-Image Line Step.
oSizeROI Region-of-Interest (ROI).
pRetVal Contains computed L1-norm of differences. This is a host pointer.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.178 NppStatus nppiQuantFwdRawTableInit_JPEG_8u (Npp8u ∗ *pQuantRawTable*, int *nQualityFactor*)

Converts regular quantization tables with the quality factor.

Parameters:

pQuantRawTable	Raw quantization table.
nQualityFactor	Quality factor for the table. Range is [1:100].

Returns:

NPP_NULL_POINTER_ERROR pQuantRawTable is a null pointer.

7.4.1.179 NppStatus nppiQuantFwdTableInit_JPEG_8u16u (const Npp8u ∗ *pQuantRawTable*, Npp16u ∗ *pQuantFwdRawTable*)

Converts raw quantization table to a forward quantization table.

Parameters:

pQuantRawTable	Raw quantization table.
pQuantFwdRawTable	Forward quantization table.

Returns:

NPP_NULL_POINTER_ERROR pQuantRawTable is a null pointer.

7.4.1.180 NppStatus nppiQuantInvTableInit_JPEG_8u16u (const Npp8u ∗ *pQuantRawTable*, Npp16u ∗ *pQuantFwdRawTable*)

Converts raw quantization table to an inverse quantization table.

Parameters:

pQuantRawTable	Raw quantization table.
pQuantFwdRawTable	Inverse quantization table.

Returns:

NPP_NULL_POINTER_ERROR pQuantRawTable or pQuantFwdRawTable is a null pointer.

7.4.1.181 NppStatus nppiRectStdDev_32s32f_C1R (const Npp32s ∗ *pSrc*, int *nSrcStep*, const Npp32f ∗ *pSqr*, int *nSqrStep*, Npp32f ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*, NppiRect *rect*)

RectStdDev Computes the standard deviation of integral images.

Parameters:

pSrc	Source-Image Pointer.

nSrcStep Source-Image Line Step.

pSqr Destination-Image Pointer.

nSqrStep Destination-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

rect rectangular window

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.182 NppStatus nppiReductionGetBufferHostSize_8u_C1R (const NppiSize & *oSizeROI*, int ∗ *hpBufferSize*)

Scratch-buffer size for nppiSum_8u_C1R.

Parameters:

oSizeROI ROI size.

hpBufferSize Host pointer where required buffer size is returned.

Returns:

Error Code.

7.4.1.183 NppStatus nppiReductionGetBufferHostSize_8u_C4R (const NppiSize & *oSizeROI*, int ∗ *hpBufferSize*)

Scratch-buffer size for nppiSum_8u_C4R.

Parameters:

oSizeROI ROI size.

hpBufferSize Host pointer where required buffer size is returned.

Returns:

Error Code.

7.4.1.184 NppStatus nppiResize_8u_C1R (const Npp8u ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcROI*, Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *dstROISize*, double *xFactor*, double *yFactor*, int *interpolation*)

8-bit unsigned image resize.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.
srcSize Size in pixels of the source image
srcROI Region of interest in the source image.
pDst Destination-Image Pointer.
nDstStep Destination-Image Line Step.
dstROISize Size in pixels of the destination image
xFactor Factors by which x dimension is changed
yFactor Factors by which y dimension is changed
interpolation The type of interpolation to perform resampling

Returns:

Image Data Related Error Codes, ROI Related Error Codes

- NPP_WRONG_INTERSECTION_ROI_ERROR indicates an error condition if srcROIRect has no intersection with the source image.
- NPP_RESIZE_NO_OPERATION_ERROR if either destination ROI width or height is less than 1 pixel.
- NPP_RESIZE_FACTOR_ERROR Indicates an error condition if either xFactor or yFactor is less than or equal to zero.
- NPP_INTERPOLATION_ERROR if interpolation has an illegal value.

7.4.1.185 NppStatus nppiResize_8u_C4R (const Npp8u ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcROI*, Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *dstROISize*, double *xFactor*, double *yFactor*, int *interpolation*)

4 channel 8-bit unsigned image resize.

Parameters:

pSrc Source-Image Pointer.
nSrcStep Source-Image Line Step.
srcSize Size in pixels of the source image
srcROI Region of interest in the source image.
pDst Destination-Image Pointer.
nDstStep Destination-Image Line Step.
dstROISize Size in pixels of the destination image
xFactor Factors by which x dimension is changed
yFactor Factors by which y dimension is changed
interpolation The type of interpolation to perform resampling

Returns:

Image Data Related Error Codes, ROI Related Error Codes

- NPP_WRONG_INTERSECTION_ROI_ERROR indicates an error condition if srcROIRect has no intersection with the source image.
- NPP_RESIZE_NO_OPERATION_ERROR if either destination ROI width or height is less than 1 pixel.
- NPP_RESIZE_FACTOR_ERROR Indicates an error condition if either xFactor or yFactor is less than or equal to zero.
- NPP_INTERPOLATION_ERROR if interpolation has an illegal value.

7.4.1.186 NppStatus nppiRGBToYCbCr420_8u_C3P3R (const Npp8u $*$ *pSrc*, int *nSrcStep*, Npp8u $**$ *pDst*, int *nDstStep*[3], NppiSize *oSizeROI*)

3 channel 8-bit unsigned packed RGB to planar YCbCr420 color conversion.

Parameters:

pSrc Source-Image Pointer.
nSrcStep Source-Image Line Step.
pDst Destination-Image Pointer.
nDstStep Destination-Image Line Step.
oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.187 NppStatus nppiRGBToYCbCr422_8u_C3C2R (const Npp8u $*$ *pSrc*, int *nSrcStep*, Npp8u $*$ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

3 channel 8-bit unsigned RGB to 2 channel chroma packed YCbCr422 color conversion.

images.

Parameters:

pSrc Source-Image Pointer.
nSrcStep Source-Image Line Step.
pDst Destination-Image Pointer.
nDstStep Destination-Image Line Step.
oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.188 NppStatus nppiRGBToYCbCr_8u_AC4R (const Npp8u $*$ *pSrc*, int *nSrcStep*, Npp8u $*$ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

4 channel 8-bit unsigned RGB to YCbCr color conversion, ignoring Alpha.

Alpha channel is the last channel and is not processed.

Parameters:

pSrc Source-Image Pointer.
nSrcStep Source-Image Line Step.
pDst Destination-Image Pointer.
nDstStep Destination-Image Line Step.
oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.189 NppStatus nppiRGBToYCbCr_8u_C3R (const Npp8u * *pSrc*, int *nSrcStep*, Npp8u * *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

3 channel 8-bit unsigned packed RGB to packed YCbCr color conversion.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.190 NppStatus nppiRGBToYCbCr_8u_P3R (const Npp8u *const * *pSrc*, int *nSrcStep*, Npp8u ** *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

3 channel planar 8-bit unsigned RGB to YCbCr color conversion.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.191 NppStatus nppiRotate_8u_C1R (const Npp8u * *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcROI*, Npp8u * *pDst*, int *nDstStep*, NppiRect *dstROI*, double *angle*, double *xShift*, double *yShift*, int *interpolation*)

8-bit unsigned image rotate.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

srcSize Size in pixels of the source image

srcROI Region of interest in the source image.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

dstROI Region of interest in the destination image.

angle The angle of rotation in degrees.

xShift Shift along horizontal axis

yShift Shift along vertical axis

interpolation The type of interpolation to perform resampling

Returns:

Image Data Related Error Codes, ROI Related Error Codes

- NPP_INTERPOLATION_ERROR if interpolation has an illegal value.
- NPP_RECT_ERROR Indicates an error condition if width or height of the intersection of the srcRoi and source image is less than or equal to 1.
- NPP_WRONG_INTERSECTION_ROI_ERROR indicates an error condition if srcROIRect has no intersection with the source image.
- NPP_WRONG_INTERSECTION_QUAD_WARNING indicates a warning that no operation is performed if the transformed source ROI does not intersect the destination ROI.

7.4.1.192 NppStatus nppiRotate_8u_C4R (const Npp8u ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcROI*, Npp8u ∗ *pDst*, int *nDstStep*, NppiRect *dstROI*, double *angle*, double *xShift*, double *yShift*, int *interpolation*)

4 channel 8-bit unsigned image rotate.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

srcSize Size in pixels of the source image

srcROI Region of interest in the source image.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

dstROI Region of interest in the destination image.

angle The angle of rotation in degrees.

xShift Shift along horizontal axis

yShift Shift along vertical axis

interpolation The type of interpolation to perform resampling

Returns:

Image Data Related Error Codes, ROI Related Error Codes

- NPP_INTERPOLATION_ERROR if interpolation has an illegal value.
- NPP_RECT_ERROR Indicates an error condition if width or height of the intersection of the srcRoi and source image is less than or equal to 1.
- NPP_WRONG_INTERSECTION_ROI_ERROR indicates an error condition if srcROIRect has no intersection with the source image.
- NPP_WRONG_INTERSECTION_QUAD_WARNING indicates a warning that no operation is performed if the transformed source ROI does not intersect the destination ROI.

7.4.1.193 NppStatus nppiSet_16s_AC4MR (const Npp16s *aValues*[3], Npp16s ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*, const Npp8u ∗ *pMask*, int *nMaskStep*)

Masked 4 channel 16-bit image set method, not affecting Alpha channel.

For RGBA images, this method allows setting of the RGB values without changing the contents of the alpha-channel (fourth channel).

Parameters:

aValues Three-channel array containing the pixel-value to be set.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pMask Pointer to the mask image. This is a single channel 8-bit unsigned int image.

nMaskStep Number of bytes between line starts of successive lines in the mask image.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.194 NppStatus nppiSet_16s_AC4R (const Npp16s *aValues*[3], Npp16s ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

4 channel 16-bit image set method, not affecting Alpha channel.

For RGBA images, this method allows setting of the RGB values without changing the contents of the alpha-channel (fourth channel).

Parameters:

aValues Three-channel array containing the pixel-value to be set.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.195 NppStatus nppiSet_16s_C1MR (Npp16s *nValue*, Npp16s ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*, const Npp8u ∗ *pMask*, int *nMaskStep*)

Masked 16-bit image set.

Parameters:

nValue New pixel value.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pMask Pointer to the mask image. This is a single channel 8-bit unsigned int image.

nMaskStep Number of bytes between line starts of successive lines in the mask image.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.196 NppStatus nppiSet_16s_C1R (Npp16s *nValue*, Npp16s * *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

16-bit image set.

Parameters:

nValue New pixel value.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.197 NppStatus nppiSet_16s_C2R (const Npp16s *aValues*[2], Npp16s * *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

2 channel 16-bit image set.

Parameters:

aValues New pixel value.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.198 NppStatus nppiSet_16s_C4CR (Npp16s *nValue*, Npp16s * *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

4 channel 16-bit unsigned image set affecting only single channel.

For RGBA images, this method allows setting of a single of the four (RGBA) values without changing the contents of the other three channels. The channel is selected via the pDst pointer. The pointer needs to point to the actual first value to be set, e.g. in order to set the R-channel (first channel), one would pass pDst unmodified, since its value actually points to the r channel. If one wanted to modify the B channel (second channel), one would pass pDst + 2 to the function.

Parameters:

nValue The pixel-value to be set.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.199 NppStatus nppiSet_16s_C4MR (const Npp16s *aValues*[4], Npp16s ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*, const Npp8u ∗ *pMask*, int *nMaskStep*)

Masked 4 channel 16-bit image set.

Parameters:

aValues New pixel value.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pMask Pointer to the mask image. This is a single channel 8-bit unsigned int image.

nMaskStep Number of bytes between line starts of successive lines in the mask image.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.200 NppStatus nppiSet_16s_C4R (const Npp16s *aValues*[4], Npp16s ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

4 channel 16-bit image set.

Parameters:

aValues New pixel value.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.201 NppStatus nppiSet_16u_AC4MR (const Npp16u *aValues*[3], Npp16u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*, const Npp8u ∗ *pMask*, int *nMaskStep*)

Masked 4 channel 16-bit unsigned image set method, not affecting Alpha channel.

For RGBA images, this method allows setting of the RGB values without changing the contents of the alpha-channel (fourth channel).

Parameters:

aValues Three-channel array containing the pixel-value to be set.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pMask Pointer to the mask image. This is a single channel 8-bit unsigned int image.

nMaskStep Number of bytes between line starts of successive lines in the mask image.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.202 NppStatus nppiSet_16u_AC4R (const Npp16u *aValues*[3], Npp16u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

4 channel 16-bit unsigned image set method, not affecting Alpha channel.

For RGBA images, this method allows setting of the RGB values without changing the contents of the alpha-channel (fourth channel).

Parameters:

aValues Three-channel array containing the pixel-value to be set.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.203 NppStatus nppiSet_16u_C1MR (Npp16u *nValue*, Npp16u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*, const Npp8u ∗ *pMask*, int *nMaskStep*)

Masked 16-bit unsigned image set.

Parameters:

nValue New pixel value.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pMask Pointer to the mask image. This is a single channel 8-bit unsigned int image.

nMaskStep Number of bytes between line starts of successive lines in the mask image.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.204 NppStatus nppiSet_16u_C1R (Npp16u *nValue*, Npp16u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

16-bit unsigned image set.

Parameters:

nValue New pixel value.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.205 NppStatus nppiSet_16u_C2R (const Npp16u *aValues*[2], Npp16u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

2 channel 16-bit unsigned image set.

Parameters:

aValues New pixel value.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.206 NppStatus nppiSet_16u_C4CR (Npp16u *nValue*, Npp16u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

4 channel 16-bit unsigned image set affecting only single channel.

For RGBA images, this method allows setting of a single of the four (RGBA) values without changing the contents of the other three channels. The channel is selected via the pDst pointer. The pointer needs to point to the actual first value to be set, e.g. in order to set the R-channel (first channel), one would pass pDst unmodified, since its value actually points to the r channel. If one wanted to modify the B channel (second channel), one would pass pDst + 2 to the function.

Parameters:

nValue The pixel-value to be set.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.207 NppStatus nppiSet_16u_C4MR (const Npp16u *aValues*[4], Npp16u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*, const Npp8u ∗ *pMask*, int *nMaskStep*)

Masked 4 channel 16-bit unsigned image set.

Parameters:

aValues New pixel value.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pMask Pointer to the mask image. This is a single channel 8-bit unsigned int image.

nMaskStep Number of bytes between line starts of successive lines in the mask image.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.208 NppStatus nppiSet_16u_C4R (const Npp16u *aValues*[4], Npp16u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

4 channel 16-bit unsigned image set.

Parameters:

aValues New pixel value.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.209 NppStatus nppiSet_32f_AC4MR (const Npp32f *aValues*[3], Npp32f ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*, const Npp8u ∗ *pMask*, int *nMaskStep*)

Masked 4 channel 32-bit floating point image set method, not affecting Alpha channel.

For RGBA images, this method allows setting of the RGB values without changing the contents of the alpha-channel (fourth channel).

Parameters:

aValues Three-channel array containing the pixel-value to be set.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pMask Pointer to the mask image. This is a single channel 8-bit unsigned int image.

nMaskStep Number of bytes between line starts of successive lines in the mask image.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.210 NppStatus nppiSet_32f_AC4R (const Npp32f *aValues*[3], Npp32f ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

4 channel 32-bit floating point image set method, not affecting Alpha channel.

For RGBA images, this method allows setting of the RGB values without changing the contents of the alpha-channel (fourth channel).

Parameters:

aValues Three-channel array containing the pixel-value to be set.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.211 NppStatus nppiSet_32f_C1MR (Npp32f *nValue*, Npp32f ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*, const Npp8u ∗ *pMask*, int *nMaskStep*)

Masked 32-bit floating point image set.

Parameters:

nValue New pixel value.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pMask Pointer to the mask image. This is a single channel 8-bit unsigned int image.

nMaskStep Number of bytes between line starts of successive lines in the mask image.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.212 NppStatus nppiSet_32f_C1R (Npp32f *nValue*, Npp32f ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

32-bit floating point image set.

Parameters:

nValue New pixel value.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.213 NppStatus nppiSet_32f_C4CR (Npp32f *nValue*, Npp32f ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

4 channel 32-bit floating point image set affecting only single channel.

For RGBA images, this method allows setting of a single of the four (RGBA) values without changing the contents of the other three channels. The channel is selected via the pDst pointer. The pointer needs to point to the actual first value to be set, e.g. in order to set the R-channel (first channel), one would pass pDst unmodified, since its value actually points to the r channel. If one wanted to modify the B channel (second channel), one would pass pDst + 2 to the function.

Parameters:

nValue The pixel-value to be set.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.214 NppStatus nppiSet_32f_C4MR (const Npp32f *aValues*[4], Npp32f ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*, const Npp8u ∗ *pMask*, int *nMaskStep*)

Masked 4 channel 32-bit floating point image set.

Parameters:

aValues New pixel value.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pMask Pointer to the mask image. This is a single channel 8-bit unsigned int image.

nMaskStep Number of bytes between line starts of successive lines in the mask image.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.215 NppStatus nppiSet_32f_C4R (const Npp32f *aValues*[4], Npp32f ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

4 channel 32-bit floating point image set.

Parameters:

aValues New pixel value.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.216 NppStatus nppiSet_32s_AC4MR (const Npp32s *aValues*[3], Npp32s ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*, const Npp8u ∗ *pMask*, int *nMaskStep*)

Masked 4 channel 16-bit image set method, not affecting Alpha channel.

For RGBA images, this method allows setting of the RGB values without changing the contents of the alpha-channel (fourth channel).

Parameters:

aValues Three-channel array containing the pixel-value to be set.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pMask Pointer to the mask image. This is a single channel 8-bit unsigned int image.

nMaskStep Number of bytes between line starts of successive lines in the mask image.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.217 NppStatus nppiSet_32s_AC4R (const Npp32s *aValues*[3], Npp32s ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

4 channel 16-bit image set method, not affecting Alpha channel.

For RGBA images, this method allows setting of the RGB values without changing the contents of the alpha-channel (fourth channel).

Parameters:

aValues Three-channel array containing the pixel-value to be set.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.218 NppStatus nppiSet_32s_C1MR (Npp32s *nValue*, Npp32s ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*, const Npp8u ∗ *pMask*, int *nMaskStep*)

Masked 32-bit image set.

Parameters:

nValue New pixel value.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pMask Pointer to the mask image. This is a single channel 8-bit unsigned int image.

nMaskStep Number of bytes between line starts of successive lines in the mask image.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.219 NppStatus nppiSet_32s_C1R (Npp32s *nValue*, Npp32s ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

32-bit image set.

Parameters:

nValue New pixel value.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.220 NppStatus nppiSet_32s_C4CR (Npp32s *nValue*, Npp32s ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

4 channel 32-bit unsigned image set affecting only single channel.

For RGBA images, this method allows setting of a single of the four (RGBA) values without changing the contents of the other three channels. The channel is selected via the pDst pointer. The pointer needs to point to the actual first value to be set, e.g. in order to set the R-channel (first channel), one would pass pDst unmodified, since its value actually points to the r channel. If one wanted to modify the B channel (second channel), one would pass pDst + 2 to the function.

Parameters:

nValue The pixel-value to be set.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.221 NppStatus nppiSet_32s_C4MR (const Npp32s *aValues*[4], Npp32s ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*, const Npp8u ∗ *pMask*, int *nMaskStep*)

Masked 4 channel 32-bit image set.

Parameters:

aValues New pixel value.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pMask Pointer to the mask image. This is a single channel 8-bit unsigned int image.

nMaskStep Number of bytes between line starts of successive lines in the mask image.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.222 NppStatus nppiSet_32s_C4R (const Npp32s *aValues*[4], Npp32s ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

4 channel 32-bit image set.

Parameters:

aValues New pixel value.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.223 NppStatus nppiSet_8u_AC4MR (const Npp8u *aValues*[3], Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*, const Npp8u ∗ *pMask*, int *nMaskStep*)

Masked 4 channel 8-bit unsigned image set method, not affecting Alpha channel.

For RGBA images, this method allows setting of the RGB values without changing the contents of the alpha-channel (fourth channel).

Parameters:

aValues Three-channel array containing the pixel-value to be set.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pMask Pointer to the mask image. This is a single channel 8-bit unsigned int image.

nMaskStep Number of bytes between line starts of successive lines in the mask image.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.224 NppStatus nppiSet_8u_AC4R (const Npp8u *aValues*[3], Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

4 channel 8-bit unsigned image set method, not affecting Alpha channel.

For RGBA images, this method allows setting of the RGB values without changing the contents of the alpha-channel (fourth channel).

Parameters:

aValues Three-channel array containing the pixel-value to be set.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.225 NppStatus nppiSet_8u_C1MR (Npp8u *nValue*, Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*, const Npp8u ∗ *pMask*, int *nMaskStep*)

Masked 8-bit unsigned image set.

The 8-bit mask image affects setting of the respective pixels in the destination image. If the mask value is zero (0) the pixel is not set, if the mask is non-zero, the corresponding destination pixel is set to specified value.

Parameters:

nValue The pixel value to be set.

pDst Pointer Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pMask Pointer to the mask image. This is a single channel 8-bit unsigned int image.

nMaskStep Number of bytes between line starts of successive lines in the mask image.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.226 NppStatus nppiSet_8u_C1R (Npp8u *nValue*, Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

8-bit unsigned image set.

Parameters:

nValue The pixel value to be set.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.227 NppStatus nppiSet_8u_C4CR (Npp8u *nValue*, Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

4 channel 8-bit unsigned image set affecting only single channel.

For RGBA images, this method allows setting of a single of the four (RGBA) values without changing the contents of the other three channels. The channel is selected via the pDst pointer. The pointer needs to point to the actual first value to be set, e.g. in order to set the R-channel (first channel), one would pass pDst unmodified, since its value actually points to the r channel. If one wanted to modify the B channel (second channel), one would pass pDst + 2 to the function.

Parameters:

nValue The pixel-value to be set.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.228 NppStatus nppiSet_8u_C4MR (const Npp8u *aValues*[4], Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*, const Npp8u ∗ *pMask*, int *nMaskStep*)

Masked 4 channel 8-bit unsigned image set.

Parameters:

aValues Four-channel array containing the pixel-value to be set.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pMask Pointer to the mask image. This is a single channel 8-bit unsigned int image.

nMaskStep Number of bytes between line starts of successive lines in the mask image.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.229 NppStatus nppiSet_8u_C4R (const Npp8u *aValues*[4], Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

4 channel 8-bit unsigned image set.

Parameters:

aValues Four-channel array containing the pixel-value to be set.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.230 NppStatus nppiSetDefaultQuantTable (Npp8u ∗ *pQuantRawTable*, int *tableIndex*)

Fills out the quantization table with either luminance and chrominance tables for JPEG.

Parameters:

pQuantRawTable Raw quantization table.

tableIndex Choice for Luminance (tableIndex is 0) or Chrominance component (tableIndex is 1).

Returns:

Error codes:

- NPP_NULL_POINTER_ERROR pQuantRawTable is a null pointer.
- NPP_INVALID_INPUT if tableIndex is not 0 or 1.

7.4.1.231 NppStatus nppiSqrIntegral_8u32s32f_C1R (Npp8u ∗ *pSrc*, int *nSrcStep*, Npp32s ∗ *pDst*, int *nDstStep*, Npp32f ∗ *pSqr*, int *nSqrStep*, NppiSize *srcROI*, Npp32s *val*, Npp32f *valSqr*, Npp32s *integralImageNewHeight*)

SqrIntegral Transforms an image to integral and integral of pixel squares representation.

This function assumes that the integral and integral of squares images.

Parameters:

pSrc Source-Image Pointer.
nSrcStep Source-Image Line Step.
pDst Destination-Image Pointer.
nDstStep Destination-Image Line Step.
pSqr Destination-Image Pointer.
nSqrStep Destination-Image Line Step.
srcROI Region-of-Interest (ROI).
val The value to add to pDst image pixels
valSqr The value to add to pSqr image pixels
integralImageNewHeight Extended height of output surfaces (needed by transpose in primitive)

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.232 NppStatus nppiSub_32f_C1R (const Npp32f ∗ *pSrc1*, int *nSrc1Step*, const Npp32f ∗ *pSrc2*, int *nSrc2Step*, Npp32f ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

32-bit floating point image subtraction.

Subtract pSrc1's pixels from corresponding pixels in pSrc2.

Parameters:

pSrc1 Source-Image Pointer.
nSrc1Step Source-Image Line Step.
pSrc2 Source-Image Pointer.
nSrc2Step Source-Image Line Step.
pDst Destination-Image Pointer.
nDstStep Destination-Image Line Step.
oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.233 NppStatus nppiSub_32s_C1R (const Npp32s $*$ *pSrc1*, int *nSrc1Step*, const Npp32s $*$ *pSrc2*, int *nSrc2Step*, Npp32s $*$ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

32-bit image subtraction.

Subtract pSrc1's pixels from corresponding pixels in pSrc2.

Parameters:

pSrc1 Source-Image Pointer.

nSrc1Step Source-Image Line Step.

pSrc2 Source-Image Pointer.

nSrc2Step Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.234 NppStatus nppiSub_8u_AC4RSfs (const Npp8u $*$ *pSrc1*, int *nSrc1Step*, const Npp8u $*$ *pSrc2*, int *nSrc2Step*, Npp8u $*$ *pDst*, int *nDstStep*, NppiSize *oSizeROI*, int *nScaleFactor*)

4 channel 8-bit unsigned image subtraction, not affecting Alpha.

Subtract pSrc1's pixels from corresponding pixels in pSrc2.

Parameters:

pSrc1 Source-Image Pointer.

nSrc1Step Source-Image Line Step.

pSrc2 Source-Image Pointer.

nSrc2Step Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

nScaleFactor Result pixel values are scaled by 2^(-nScaleFactor) and then clamped to [0,255] range.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.235 NppStatus nppiSub_8u_C1RSfs (const Npp8u $*$ *pSrc1*, int *nSrc1Step*, const Npp8u $*$ *pSrc2*, int *nSrc2Step*, Npp8u $*$ *pDst*, int *nDstStep*, NppiSize *oSizeROI*, int *nScaleFactor*)

8-bit unsigned image subtraction.

Subtract the pixel values of corresponding pixels in the ROI and write them to the output image.

Parameters:

pSrc1 Source-Image Pointer.

nSrc1Step Source-Image Line Step.

pSrc2 Source-Image Pointer.

nSrc2Step Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

nScaleFactor Result pixel values are scaled by 2^(-nScaleFactor) and then clamped to [0,255] range.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.236 NppStatus nppiSub_8u_C4RSfs (const Npp8u ∗ *pSrc1*, int *nSrc1Step*, const Npp8u ∗ *pSrc2*, int *nSrc2Step*, Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*, int *nScaleFactor*)

4 channel 8-bit unsigned image subtraction.

Subtract pSrc1's pixels from corresponding pixels in pSrc2.

Parameters:

pSrc1 Source-Image Pointer.

nSrc1Step Source-Image Line Step.

pSrc2 Source-Image Pointer.

nSrc2Step Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

nScaleFactor Result pixel values are scaled by 2^(-nScaleFactor) and then clamped to [0,255] range.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.237 NppStatus nppiSubC_32f_C1R (const Npp32f ∗ *pSrc*, int *nSrcStep*, Npp32f *nValue*, Npp32f ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

32-bit floating point image subtract constant.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

nValue Constant.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.238 NppStatus nppiSubC_32fc_C1R (const Npp32fc ∗ *pSrc*, int *nSrcStep*, Npp32fc *nValue*, Npp32fc ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

32-bit complex floating point image subtract constant.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

nValue Constant.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.239 NppStatus nppiSum_8u_C1R (const Npp8u ∗ *pSrc*, int *nSrcStep*, NppiSize *oSizeROI*, Npp8u ∗ *pDeviceBuffer*, Npp64f ∗ *pSum*)

8-bit unsigned image sum.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pDeviceBuffer Pointer to the required device memory allocation.

∗pSum Contains computed sum.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.240 NppStatus nppiSum_8u_C4R (const Npp8u ∗ *pSrc*, int *nSrcStep*, NppiSize *oSizeROI*, Npp8u ∗ *pDeviceBuffer*, Npp64f *aSum*[4])

4 channel 8-bit unsigned image sum.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

oSizeROI Region-of-Interest (ROI).

pDeviceBuffer Pointer to the required device memory allocation.

aSum Array contains computed sum for each channel.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.241 NppStatus nppiSumWindowColumn_8u32f_C1R (const Npp8u $*$ *pSrc*, Npp32s *nSrcStep*, Npp32f $*$ *pDst*, Npp32s *nDstStep*, NppiSize *oROI*, Npp32s *nMaskSize*, Npp32s *nAnchor*)

8-bit unsigned 1D (column) sum to 32f.

Apply Column Window Summation filter over a 1D mask region around each source pixel for 1-channel 8 bit/pixel input images with 32-bit floating point output. Result 32-bit floating point pixel is equal to the sum of the corresponding and neighboring column pixel values in a mask region of the source image defined by nMaskSize and nAnchor.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oROI Region-of-Interest (ROI).

nMaskSize Length of the linear kernel array.

nAnchor Y offset of the kernel origin frame of reference w.r.t the source pixel.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.242 NppStatus nppiSumWindowRow_8u32f_C1R (const Npp8u $*$ *pSrc*, Npp32s *nSrcStep*, Npp32f $*$ *pDst*, Npp32s *nDstStep*, NppiSize *oROI*, Npp32s *nMaskSize*, Npp32s *nAnchor*)

8-bit unsigned 1D (row) sum to 32f.

Apply Row Window Summation filter over a 1D mask region around each source pixel for 1-channel 8-bit pixel input images with 32-bit floating point output. Result 32-bit floating point pixel is equal to the sum of the corresponding and neighboring row pixel values in a mask region of the source image defined by iKernelDim and iAnchorX.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oROI Region-of-Interest (ROI).

nMaskSize Length of the linear kernel array.

nAnchor X offset of the kernel origin frame of reference w.r.t the source pixel.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.243 NppStatus nppiSwapChannels_8u_C4IR (Npp8u ∗ *pSrcDst*, int *nSrcDstStep*, NppiSize *oSizeROI*, const int *aDstOrder*[4])

4 channel 8-bit unsigned swap channels, in-place.

Parameters:

pSrcDst In-Place Image Pointer.

nSrcDstStep In-Place Line Step.

oSizeROI Region-of-Interest (ROI).

aDstOrder Integer array describing how channel values are permutated. The n-th entry of the array contains the number of the channel that is stored in the n-th channel of the output image. E.g. Given an RGBA image, aDstOrder = [3,2,1,0] converts this to ABGR channel order.

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.244 NppStatus nppiThreshold_32f_C1R (const Npp32f ∗ *pSrc*, int *nSrcStep*, Npp32f ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*, Npp32f *nThreshold*, NppCmpOp *eComparisonOperation*)

32-bit floating point threshold.

If for a comparison operations OP the predicate (sourcePixel OP nThreshold) is true, the pixel is set to nThreshold, otherwise it is set to sourcePixel.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

nThreshold The threshold value.

eComparisonOperation The type of comparison operation to be used. The only valid values are: NPP_CMP_LESS and NPP_CMP_GREATER.

Returns:

Image Data Related Error Codes, ROI Related Error Codes, or NPP_NOT_SUPPORTED_MODE_-ERROR if an invalid comparison operation type is specified.

7.4.1.245 NppStatus nppiThreshold_8u_AC4R (const Npp8u ∗ *pSrc*, int *nSrcStep*, Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*, const Npp8u *aThresholds*[3], NppCmpOp *eComparisonOperation*)

4 channel 8-bit unsigned image threshold, not affecting Alpha.

If for a comparison operations OP the predicate (sourcePixel.channel OP nThreshold) is true, the channel value is set to nThreshold, otherwise it is set to sourcePixel.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

aThresholds The threshold values, one per color channel.

eComparisonOperation The type of comparison operation to be used. The only valid values are: NPP_CMP_LESS and NPP_CMP_GREATER.

Returns:

Image Data Related Error Codes, ROI Related Error Codes, or NPP_NOT_SUPPORTED_MODE_-ERROR if an invalid comparison operation type is specified.

7.4.1.246 NppStatus nppiTranspose_8u_C1R (const Npp8u ∗ *pSrc*, int *nSrcStep*, Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *oROI*)

8-bit image transpose.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Pointer to the destination ROI.

nDstStep Destination-Image Line Step.

oROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.247 NppStatus nppiWarpAffine_16u_AC4R (const Npp16u ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp16u ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Affine transform of an image (16bit unsigned integer, four channels RGBA).

See also:

nppiWarpAffine_16u_C1R

7.4.1.248 NppStatus nppiWarpAffine_16u_C1R (const Npp16u $*$ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp16u $*$ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Affine transform of an image (16bit unsigned integer, single channel).

This function operates using given transform coefficients that can be obtained by using nppiGetAffineTransform function or set explicitly. The function operates on source and destination regions of interest. The affine warp function transforms the source image pixel coordinates (x, y) according to the following formulas:

$$X_{new} = C_{00} * x + C_{01} * y + C_{02} \qquad Y_{new} = C_{10} * x + C_{11} * y + C_{12}$$

The transformed part of the source image is resampled using the specified interpolation method and written to the destination ROI. The functions nppiGetAffineQuad and nppiGetAffineBound can help with destination ROI specification.

NPPI specific recommendation: The function operates using 2 types of kernels: fast and accurate. The fast method is about 4 times faster than its accurate variant, but does not perform memory access checks and requires the destination ROI to be 64 bytes aligned. Hence any destination ROI is chunked into 3 vertical stripes: the first and the third are processed by accurate kernels and the central one is processed by the fast one. In order to get the maximum available speed of execution, the projection of destination ROI onto image addresses must be 64 bytes aligned. This is always true if the values `(int)((void *)(pDst + dstRoi.x))` and `(int)((void *)(pDst + dstRoi.x + dstRoi.width))` are multiples of 64. Another rule of thumb is to specify destination ROI in such way that left and right sides of the projected image are separated from the ROI by at least 63 bytes from each side. However, this requires the whole ROI to be part of allocated memory. In case when the conditions above are not satisfied, the function may decrease in speed slightly and will return NPP_MISALIGNED_DST_ROI_WARNING warning.

Parameters:

pSrc Source-Image Pointer.

srcSize Size of source image in pixels

nSrcStep Source-Image Line Step.

srcRoi Source ROI

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

dstRoi Destination ROI

coeffs Affine transform coefficients

interpolation Interpolation mode: can be NPPI_INTER_NN, NPPI_INTER_LINEAR or NPPI_-INTER_CUBIC

Returns:

Image Data Related Error Codes, ROI Related Error Codes

- NPP_RECT_ERROR Indicates an error condition if width or height of the intersection of the srcRoi and source image is less than or equal to 1
- NPP_WRONG_INTERSECTION_ROI_ERROR Indicates an error condition if srcRoi has no intersection with the source image
- NPP_INTERPOLATION_ERROR Indicates an error condition if interpolation has an illegal value
- NPP_COEFF_ERROR Indicates an error condition if coefficient values are invalid

- NPP_WRONG_INTERSECTION_QUAD_WARNING Indicates a warning that no operation is performed if the transformed source ROI has no intersection with the destination ROI
- NPP_MISALIGNED_DST_ROI_WARNING Indicates a warning that the speed of primitive execution was reduced due to destination ROI misalignment

7.4.1.249 NppStatus nppiWarpAffine_16u_C3R (const Npp16u ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp16u ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Affine transform of an image (16bit unsigned integer, three channels).

See also:

nppiWarpAffine_16u_C1R

7.4.1.250 NppStatus nppiWarpAffine_16u_C4R (const Npp16u ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp16u ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Affine transform of an image (16bit unsigned integer, four channels).

See also:

nppiWarpAffine_16u_C1R

7.4.1.251 NppStatus nppiWarpAffine_16u_P3R (const Npp16u ∗ *pSrc*[3], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp16u ∗ *pDst*[3], int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Affine transform of an image (16bit unsigned integer, three planes).

See also:

nppiWarpAffine_16u_C1R

7.4.1.252 NppStatus nppiWarpAffine_16u_P4R (const Npp16u ∗ *pSrc*[4], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp16u ∗ *pDst*[4], int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Affine transform of an image (16bit unsigned integer, four planes).

See also:

nppiWarpAffine_16u_C1R

7.4.1.253 NppStatus nppiWarpAffine_32f_AC4R (const Npp32f $*$ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32f $*$ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Affine transform of an image (32bit float, four channels RGBA).

See also:

nppiWarpAffine_32f_C1R

7.4.1.254 NppStatus nppiWarpAffine_32f_C1R (const Npp32f $*$ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32f $*$ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Affine transform of an image (32bit float, single channel).

This function operates using given transform coefficients that can be obtained by using nppiGetAffine-Transform function or set explicitly. The function operates on source and destination regions of interest. The affine warp function transforms the source image pixel coordinates (x, y) according to the following formulas:

$$X_{new} = C_{00} * x + C_{01} * y + C_{02} \qquad Y_{new} = C_{10} * x + C_{11} * y + C_{12}$$

The transformed part of the source image is resampled using the specified interpolation method and written to the destination ROI. The functions nppiGetAffineQuad and nppiGetAffineBound can help with destination ROI specification.

Parameters:

pSrc Source-Image Pointer.

srcSize Size of source image in pixels

nSrcStep Source-Image Line Step.

srcRoi Source ROI

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

dstRoi Destination ROI

coeffs Affine transform coefficients

interpolation Interpolation mode: can be NPPI_INTER_NN, NPPI_INTER_LINEAR or NPPI_-INTER_CUBIC

Returns:

Image Data Related Error Codes, ROI Related Error Codes

- NPP_RECT_ERROR Indicates an error condition if width or height of the intersection of the srcRoi and source image is less than or equal to 1
- NPP_WRONG_INTERSECTION_ROI_ERROR Indicates an error condition if srcRoi has no intersection with the source image
- NPP_INTERPOLATION_ERROR Indicates an error condition if interpolation has an illegal value
- NPP_COEFF_ERROR Indicates an error condition if coefficient values are invalid

- NPP_WRONG_INTERSECTION_QUAD_WARNING Indicates a warning that no operation is performed if the transformed source ROI has no intersection with the destination ROI
- NPP_MISALIGNED_DST_ROI_WARNING Indicates a warning that the speed of primitive execution was reduced due to destination ROI misalignment

7.4.1.255 NppStatus nppiWarpAffine_32f_C3R (const Npp32f ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32f ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Affine transform of an image (32bit float, three channels).

See also:

nppiWarpAffine_32f_C1R

7.4.1.256 NppStatus nppiWarpAffine_32f_C4R (const Npp32f ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32f ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Affine transform of an image (32bit float, four channels).

See also:

nppiWarpAffine_32f_C1R

7.4.1.257 NppStatus nppiWarpAffine_32f_P3R (const Npp32f ∗ *pSrc*[3], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32f ∗ *pDst*[3], int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Affine transform of an image (32bit float, three planes).

See also:

nppiWarpAffine_32f_C1R

7.4.1.258 NppStatus nppiWarpAffine_32f_P4R (const Npp32f ∗ *pSrc*[4], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32f ∗ *pDst*[4], int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Affine transform of an image (32bit float, four planes).

See also:

nppiWarpAffine_32f_C1R

7.4.1.259 NppStatus nppiWarpAffine_32s_AC4R (const Npp32s $*$ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32s $*$ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Affine transform of an image (32bit signed integer, four channels RGBA).

See also:

nppiWarpAffine_32s_C1R

7.4.1.260 NppStatus nppiWarpAffine_32s_C1R (const Npp32s $*$ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32s $*$ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Affine transform of an image (32bit signed integer, single channel).

This function operates using given transform coefficients that can be obtained by using nppiGetAffine-Transform function or set explicitly. The function operates on source and destination regions of interest. The affine warp function transforms the source image pixel coordinates (x, y) according to the following formulas:

$$X_{new} = C_{00} * x + C_{01} * y + C_{02} \qquad Y_{new} = C_{10} * x + C_{11} * y + C_{12}$$

The transformed part of the source image is resampled using the specified interpolation method and written to the destination ROI. The functions nppiGetAffineQuad and nppiGetAffineBound can help with destination ROI specification.

Parameters:

pSrc Source-Image Pointer.

srcSize Size of source image in pixels

nSrcStep Source-Image Line Step.

srcRoi Source ROI

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

dstRoi Destination ROI

coeffs Affine transform coefficients

interpolation Interpolation mode: can be NPPI_INTER_NN, NPPI_INTER_LINEAR or NPPI_-INTER_CUBIC

Returns:

Image Data Related Error Codes, ROI Related Error Codes

- NPP_RECT_ERROR Indicates an error condition if width or height of the intersection of the srcRoi and source image is less than or equal to 1
- NPP_WRONG_INTERSECTION_ROI_ERROR Indicates an error condition if srcRoi has no intersection with the source image
- NPP_INTERPOLATION_ERROR Indicates an error condition if interpolation has an illegal value
- NPP_COEFF_ERROR Indicates an error condition if coefficient values are invalid

- NPP_WRONG_INTERSECTION_QUAD_WARNING Indicates a warning that no operation is performed if the transformed source ROI has no intersection with the destination ROI
- NPP_MISALIGNED_DST_ROI_WARNING Indicates a warning that the speed of primitive execution was reduced due to destination ROI misalignment

7.4.1.261 NppStatus nppiWarpAffine_32s_C3R (const Npp32s $*$ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32s $*$ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Affine transform of an image (32bit signed integer, three channels).

See also:

nppiWarpAffine_32s_C1R

7.4.1.262 NppStatus nppiWarpAffine_32s_C4R (const Npp32s $*$ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32s $*$ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Affine transform of an image (32bit signed integer, four channels).

See also:

nppiWarpAffine_32s_C1R

7.4.1.263 NppStatus nppiWarpAffine_32s_P3R (const Npp32s $*$ *pSrc*[3], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32s $*$ *pDst*[3], int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Affine transform of an image (32bit signed integer, three planes).

See also:

nppiWarpAffine_32s_C1R

7.4.1.264 NppStatus nppiWarpAffine_32s_P4R (const Npp32s $*$ *pSrc*[4], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32s $*$ *pDst*[4], int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Affine transform of an image (32bit signed integer, four planes).

See also:

nppiWarpAffine_32s_C1R

7.4.1.265 NppStatus nppiWarpAffine_8u_AC4R (const Npp8u $*$ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp8u $*$ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Affine transform of an image (8bit unsigned integer, four channels RGBA).

See also:

nppiWarpAffine_8u_C1R

7.4.1.266 NppStatus nppiWarpAffine_8u_C1R (const Npp8u $*$ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp8u $*$ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Affine transform of an image (8bit unsigned integer, single channel).

This function operates using given transform coefficients that can be obtained by using nppiGetAffine-Transform function or set explicitly. The function operates on source and destination regions of interest. The affine warp function transforms the source image pixel coordinates (x, y) according to the following formulas:

$$X_{new} = C_{00} * x + C_{01} * y + C_{02} \qquad Y_{new} = C_{10} * x + C_{11} * y + C_{12}$$

The transformed part of the source image is resampled using the specified interpolation method and written to the destination ROI. The functions nppiGetAffineQuad and nppiGetAffineBound can help with destination ROI specification.

NPPI specific recommendation: The function operates using 2 types of kernels: fast and accurate. The fast method is about 4 times faster than its accurate variant, but does not perform memory access checks and requires the destination ROI to be 64 bytes aligned. Hence any destination ROI is chunked into 3 vertical stripes: the first and the third are processed by accurate kernels and the central one is processed by the fast one. In order to get the maximum available speed of execution, the projection of destination ROI onto image addresses must be 64 bytes aligned. This is always true if the values `(int)((void *)(pDst + dstRoi.x))` and `(int)((void *)(pDst + dstRoi.x + dstRoi.width))` are multiples of 64. Another rule of thumb is to specify destination ROI in such way that left and right sides of the projected image are separated from the ROI by at least 63 bytes from each side. However, this requires the whole ROI to be part of allocated memory. In case when the conditions above are not satisfied, the function may decrease in speed slightly and will return NPP_MISALIGNED_DST_ROI_WARNING warning.

Parameters:

pSrc Source-Image Pointer.

srcSize Size of source image in pixels

nSrcStep Source-Image Line Step.

srcRoi Source ROI

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

dstRoi Destination ROI

coeffs Affine transform coefficients

interpolation Interpolation mode: can be NPPI_INTER_NN, NPPI_INTER_LINEAR or NPPI_-INTER_CUBIC

Returns:

Image Data Related Error Codes, ROI Related Error Codes

- NPP_RECT_ERROR Indicates an error condition if width or height of the intersection of the srcRoi and source image is less than or equal to 1
- NPP_WRONG_INTERSECTION_ROI_ERROR Indicates an error condition if srcRoi has no intersection with the source image
- NPP_INTERPOLATION_ERROR Indicates an error condition if interpolation has an illegal value
- NPP_COEFF_ERROR Indicates an error condition if coefficient values are invalid
- NPP_WRONG_INTERSECTION_QUAD_WARNING Indicates a warning that no operation is performed if the transformed source ROI has no intersection with the destination ROI
- NPP_MISALIGNED_DST_ROI_WARNING Indicates a warning that the speed of primitive execution was reduced due to destination ROI misalignment

7.4.1.267 NppStatus nppiWarpAffine_8u_C3R (const Npp8u ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp8u ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Affine transform of an image (8bit unsigned integer, three channels).

See also:

nppiWarpAffine_8u_C1R

7.4.1.268 NppStatus nppiWarpAffine_8u_C4R (const Npp8u ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp8u ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Affine transform of an image (8bit unsigned integer, four channels).

See also:

nppiWarpAffine_8u_C1R

7.4.1.269 NppStatus nppiWarpAffine_8u_P3R (const Npp8u ∗ *pSrc*[3], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp8u ∗ *pDst*[3], int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Affine transform of an image (8bit unsigned integer, three planes).

See also:

nppiWarpAffine_8u_C1R

7.4.1.270 NppStatus nppiWarpAffine_8u_P4R (const Npp8u * *pSrc*[4], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp8u * *pDst*[4], int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Affine transform of an image (8bit unsigned integer, four planes).

See also:

nppiWarpAffine_8u_C1R

7.4.1.271 NppStatus nppiWarpAffineBack_16u_AC4R (const Npp16u * *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp16u * *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Inverse affine transform of an image (16bit unsigned integer, four channels RGBA).

See also:

nppiWarpAffineBack_16u_C1R

7.4.1.272 NppStatus nppiWarpAffineBack_16u_C1R (const Npp16u * *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp16u * *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Inverse affine transform of an image (16bit unsigned integer, single channel).

This function operates using given transform coefficients that can be obtained by using nppiGetAffineTransform function or set explicitly. Thus there is no need to invert coefficients in your application before calling WarpAffineBack. The function operates on source and destination regions of interest. The affine warp function transforms the source image pixel coordinates (x, y) according to the following formulas:

$$C_{00} * X_{new} + C_{01} * Y_{new} + C_{02} = x \qquad C_{10} * X_{new} + C_{11} * Y_{new} + C_{12} = y$$

The transformed part of the source image is resampled using the specified interpolation method and written to the destination ROI. The functions nppiGetAffineQuad and nppiGetAffineBound can help with destination ROI specification.

NPPI specific recommendation: The function operates using 2 types of kernels: fast and accurate. The fast method is about 4 times faster than its accurate variant, but doesn't perform memory access checks and requires the destination ROI to be 64 bytes aligned. Hence any destination ROI is chunked into 3 vertical stripes: the first and the third are processed by accurate kernels and the central one is processed by the fast one. In order to get the maximum available speed of execution, the projection of destination ROI onto image addresses must be 64 bytes aligned. This is always true if the values `(int)((void *)(pDst + dstRoi.x))` and `(int)((void *)(pDst + dstRoi.x + dstRoi.width))` are multiples of 64. Another rule of thumb is to specify destination ROI in such way that left and right sides of the projected image are separated from the ROI by at least 63 bytes from each side. However, this requires the whole ROI to be part of allocated memory. In case when the conditions above are not satisfied, the function may decrease in speed slightly and will return NPP_MISALIGNED_DST_ROI_WARNING warning.

Parameters:

pSrc Source-Image Pointer.

srcSize Size of source image in pixels

nSrcStep Source-Image Line Step.

srcRoi Source ROI

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

dstRoi Destination ROI

coeffs Affine transform coefficients

interpolation Interpolation mode: can be NPPI_INTER_NN, NPPI_INTER_LINEAR or NPPI_-INTER_CUBIC

Returns:

Image Data Related Error Codes, ROI Related Error Codes

- NPP_RECT_ERROR Indicates an error condition if width or height of the intersection of the srcRoi and source image is less than or equal to 1
- NPP_WRONG_INTERSECTION_ROI_ERROR Indicates an error condition if srcRoi has no intersection with the source image
- NPP_INTERPOLATION_ERROR Indicates an error condition if interpolation has an illegal value
- NPP_COEFF_ERROR Indicates an error condition if coefficient values are invalid
- NPP_WRONG_INTERSECTION_QUAD_WARNING Indicates a warning that no operation is performed if the transformed source ROI has no intersection with the destination ROI
- NPP_MISALIGNED_DST_ROI_WARNING Indicates a warning that the speed of primitive execution was reduced due to destination ROI misalignment

7.4.1.273 NppStatus nppiWarpAffineBack_16u_C3R (const Npp16u ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp16u ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Inverse affine transform of an image (16bit unsigned integer, three channels).

See also:

nppiWarpAffineBack_16u_C1R

7.4.1.274 NppStatus nppiWarpAffineBack_16u_C4R (const Npp16u ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp16u ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Inverse affine transform of an image (16bit unsigned integer, four channels).

See also:

nppiWarpAffineBack_16u_C1R

7.4.1.275 NppStatus nppiWarpAffineBack_16u_P3R (const Npp16u $*$ *pSrc*[3], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp16u $*$ *pDst*[3], int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Inverse affine transform of an image (16bit unsigned integer, three planes).

See also:

nppiWarpAffineBack_16u_C1R

7.4.1.276 NppStatus nppiWarpAffineBack_16u_P4R (const Npp16u $*$ *pSrc*[4], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp16u $*$ *pDst*[4], int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Inverse affine transform of an image (16bit unsigned integer, four planes).

See also:

nppiWarpAffineBack_16u_C1R

7.4.1.277 NppStatus nppiWarpAffineBack_32f_AC4R (const Npp32f $*$ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32f $*$ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Inverse affine transform of an image (32bit float, four channels RGBA).

See also:

nppiWarpAffineBack_32f_C1R

7.4.1.278 NppStatus nppiWarpAffineBack_32f_C1R (const Npp32f $*$ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32f $*$ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Inverse affine transform of an image (32bit float, single channel).

This function operates using given transform coefficients that can be obtained by using nppiGetAffine-Transform function or set explicitly. Thus there is no need to invert coefficients in your application before calling WarpAffineBack. The function operates on source and destination regions of interest. The affine warp function transforms the source image pixel coordinates (x, y) according to the following formulas:

$$C_{00} * X_{new} + C_{01} * Y_{new} + C_{02} = x \qquad C_{10} * X_{new} + C_{11} * Y_{new} + C_{12} = y$$

The transformed part of the source image is resampled using the specified interpolation method and written to the destination ROI. The functions nppiGetAffineQuad and nppiGetAffineBound can help with destination ROI specification.

Parameters:

pSrc Source-Image Pointer.

srcSize Size of source image in pixels

nSrcStep Source-Image Line Step.

srcRoi Source ROI

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

dstRoi Destination ROI

coeffs Affine transform coefficients

interpolation Interpolation mode: can be NPPI_INTER_NN, NPPI_INTER_LINEAR or NPPI_-INTER_CUBIC

Returns:

Image Data Related Error Codes, ROI Related Error Codes

- NPP_RECT_ERROR Indicates an error condition if width or height of the intersection of the srcRoi and source image is less than or equal to 1
- NPP_WRONG_INTERSECTION_ROI_ERROR Indicates an error condition if srcRoi has no intersection with the source image
- NPP_INTERPOLATION_ERROR Indicates an error condition if interpolation has an illegal value
- NPP_COEFF_ERROR Indicates an error condition if coefficient values are invalid
- NPP_WRONG_INTERSECTION_QUAD_WARNING Indicates a warning that no operation is performed if the transformed source ROI has no intersection with the destination ROI
- NPP_MISALIGNED_DST_ROI_WARNING Indicates a warning that the speed of primitive execution was reduced due to destination ROI misalignment

7.4.1.279 NppStatus nppiWarpAffineBack_32f_C3R (const Npp32f ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32f ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Inverse affine transform of an image (32bit float, three channels).

See also:

nppiWarpAffineBack_32f_C1R

7.4.1.280 NppStatus nppiWarpAffineBack_32f_C4R (const Npp32f ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32f ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Inverse affine transform of an image (32bit float, four channels).

See also:

nppiWarpAffineBack_32f_C1R

7.4.1.281 NppStatus nppiWarpAffineBack_32f_P3R (const Npp32f $*$ *pSrc*[3], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32f $*$ *pDst*[3], int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Inverse affine transform of an image (32bit float, three planes).

See also:

nppiWarpAffineBack_32f_C1R

7.4.1.282 NppStatus nppiWarpAffineBack_32f_P4R (const Npp32f $*$ *pSrc*[4], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32f $*$ *pDst*[4], int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Inverse affine transform of an image (32bit float, four planes).

See also:

nppiWarpAffineBack_32f_C1R

7.4.1.283 NppStatus nppiWarpAffineBack_32s_AC4R (const Npp32s $*$ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32s $*$ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Inverse affine transform of an image (32bit signed integer, four channels RGBA).

See also:

nppiWarpAffineBack_32s_C1R

7.4.1.284 NppStatus nppiWarpAffineBack_32s_C1R (const Npp32s $*$ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32s $*$ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Inverse affine transform of an image (32bit signed integer, single channel).

This function operates using given transform coefficients that can be obtained by using nppiGetAffine-Transform function or set explicitly. Thus there is no need to invert coefficients in your application before calling WarpAffineBack. The function operates on source and destination regions of interest. The affine warp function transforms the source image pixel coordinates (x, y) according to the following formulas:

$$C_{00} * X_{new} + C_{01} * Y_{new} + C_{02} = x \qquad C_{10} * X_{new} + C_{11} * Y_{new} + C_{12} = y$$

The transformed part of the source image is resampled using the specified interpolation method and written to the destination ROI. The functions nppiGetAffineQuad and nppiGetAffineBound can help with destination ROI specification.

Parameters:

pSrc Source-Image Pointer.

srcSize Size of source image in pixels

nSrcStep Source-Image Line Step.

srcRoi Source ROI

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

dstRoi Destination ROI

coeffs Affine transform coefficients

interpolation Interpolation mode: can be NPPI_INTER_NN, NPPI_INTER_LINEAR or NPPI_-INTER_CUBIC

Returns:

Image Data Related Error Codes, ROI Related Error Codes

- NPP_RECT_ERROR Indicates an error condition if width or height of the intersection of the srcRoi and source image is less than or equal to 1
- NPP_WRONG_INTERSECTION_ROI_ERROR Indicates an error condition if srcRoi has no intersection with the source image
- NPP_INTERPOLATION_ERROR Indicates an error condition if interpolation has an illegal value
- NPP_COEFF_ERROR Indicates an error condition if coefficient values are invalid
- NPP_WRONG_INTERSECTION_QUAD_WARNING Indicates a warning that no operation is performed if the transformed source ROI has no intersection with the destination ROI
- NPP_MISALIGNED_DST_ROI_WARNING Indicates a warning that the speed of primitive execution was reduced due to destination ROI misalignment

7.4.1.285 NppStatus nppiWarpAffineBack_32s_C3R (const Npp32s ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32s ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Inverse affine transform of an image (32bit signed integer, three channels).

See also:

nppiWarpAffineBack_32s_C1R

7.4.1.286 NppStatus nppiWarpAffineBack_32s_C4R (const Npp32s ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32s ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Inverse affine transform of an image (32bit signed integer, four channels).

See also:

nppiWarpAffineBack_32s_C1R

7.4.1.287 NppStatus nppiWarpAffineBack_32s_P3R (const Npp32s ∗ *pSrc*[3], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32s ∗ *pDst*[3], int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Inverse affine transform of an image (32bit signed integer, three planes).

See also:

nppiWarpAffineBack_32s_C1R

7.4.1.288 NppStatus nppiWarpAffineBack_32s_P4R (const Npp32s ∗ *pSrc*[4], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32s ∗ *pDst*[4], int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Inverse affine transform of an image (32bit signed integer, four planes).

See also:

nppiWarpAffineBack_32s_C1R

7.4.1.289 NppStatus nppiWarpAffineBack_8u_AC4R (const Npp8u ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp8u ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Inverse affine transform of an image (8bit unsigned integer, four channels RGBA).

See also:

nppiWarpAffineBack_8u_C1R

7.4.1.290 NppStatus nppiWarpAffineBack_8u_C1R (const Npp8u ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp8u ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Inverse affine transform of an image (8bit unsigned integer, single channel).

This function operates using given transform coefficients that can be obtained by using nppiGetAffine-Transform function or set explicitly. Thus there is no need to invert coefficients in your application before calling WarpAffineBack. The function operates on source and destination regions of interest. The affine warp function transforms the source image pixel coordinates (x, y) according to the following formulas:

$$C_{00} * X_{new} + C_{01} * Y_{new} + C_{02} = x \qquad C_{10} * X_{new} + C_{11} * Y_{new} + C_{12} = y$$

The transformed part of the source image is resampled using the specified interpolation method and written to the destination ROI. The functions nppiGetAffineQuad and nppiGetAffineBound can help with destination ROI specification.

NPPI specific recommendation: The function operates using 2 types of kernels: fast and accurate. The fast method is about 4 times faster than its accurate variant, but doesn't perform memory access checks and requires the destination ROI to be 64 bytes aligned. Hence any destination ROI is chunked into 3 vertical stripes: the first and the third are processed by accurate kernels and the central one is processed by the fast one. In order to get the maximum available speed of execution, the projection of destination ROI onto image addresses must be 64 bytes aligned. This is always true if the values `(int)((void *)(pDst + dstRoi.x))` and `(int)((void *)(pDst + dstRoi.x + dstRoi.width))` are multiples of 64. Another rule of thumb is to specify destination ROI in such way that left and right sides of the projected image are separated from the ROI by at least 63 bytes from each side. However, this requires the whole ROI to be part of allocated memory. In case when the conditions above are not satisfied, the function may decrease in speed slightly and will return NPP_MISALIGNED_DST_ROI_WARNING warning.

Parameters:

pSrc Source-Image Pointer.

srcSize Size of source image in pixels

nSrcStep Source-Image Line Step.

srcRoi Source ROI

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

dstRoi Destination ROI

coeffs Affine transform coefficients

interpolation Interpolation mode: can be NPPI_INTER_NN, NPPI_INTER_LINEAR or NPPI_-INTER_CUBIC

Returns:

Image Data Related Error Codes, ROI Related Error Codes

- NPP_RECT_ERROR Indicates an error condition if width or height of the intersection of the srcRoi and source image is less than or equal to 1
- NPP_WRONG_INTERSECTION_ROI_ERROR Indicates an error condition if srcRoi has no intersection with the source image
- NPP_INTERPOLATION_ERROR Indicates an error condition if interpolation has an illegal value
- NPP_COEFF_ERROR Indicates an error condition if coefficient values are invalid
- NPP_WRONG_INTERSECTION_QUAD_WARNING Indicates a warning that no operation is performed if the transformed source ROI has no intersection with the destination ROI
- NPP_MISALIGNED_DST_ROI_WARNING Indicates a warning that the speed of primitive execution was reduced due to destination ROI misalignment

7.4.1.291 NppStatus nppiWarpAffineBack_8u_C3R (const Npp8u * *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp8u * *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Inverse affine transform of an image (8bit unsigned integer, three channels).

See also:

nppiWarpAffineBack_8u_C1R

7.4.1.292 NppStatus nppiWarpAffineBack_8u_C4R (const Npp8u ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp8u ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Inverse affine transform of an image (8bit unsigned integer, four channels).

See also:

nppiWarpAffineBack_8u_C1R

7.4.1.293 NppStatus nppiWarpAffineBack_8u_P3R (const Npp8u ∗ *pSrc*[3], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp8u ∗ *pDst*[3], int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Inverse affine transform of an image (8bit unsigned integer, three planes).

See also:

nppiWarpAffineBack_8u_C1R

7.4.1.294 NppStatus nppiWarpAffineBack_8u_P4R (const Npp8u ∗ *pSrc*[4], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp8u ∗ *pDst*[4], int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[2][3], int *interpolation*)

Inverse affine transform of an image (8bit unsigned integer, four planes).

See also:

nppiWarpAffineBack_8u_C1R

7.4.1.295 NppStatus nppiWarpAffineQuad_16u_AC4R (const Npp16u ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp16u ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Affine transform of an image (16bit unsigned integer, four channels RGBA).

See also:

nppiWarpAffineQuad_16u_C1R

7.4.1.296 NppStatus nppiWarpAffineQuad_16u_C1R (const Npp16u ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp16u ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Affine transform of an image (16bit unsigned integer, single channel).

This function performs affine warping of a the specified quadrangle in the source image to the specified quadrangle in the destination image. The function nppiWarpAffineQuad uses the same formulas for pixel mapping as in nppiWarpAffine function. The transform coefficients are computed internally. The transformed part of the source image is resampled using the specified interpolation method and written to the destination ROI.

NPPI specific recommendation: The function operates using 2 types of kernels: fast and accurate. The fast method is about 4 times faster than its accurate variant, but doesn't perform memory access checks and requires the destination ROI to be 64 bytes aligned. Hence any destination ROI is chunked into 3 vertical stripes: the first and the third are processed by accurate kernels and the central one is processed by the fast one. In order to get the maximum available speed of execution, the projection of destination ROI onto image addresses must be 64 bytes aligned. This is always true if the values `(int)((void *)(pDst + dstRoi.x))` and `(int)((void *)(pDst + dstRoi.x + dstRoi.width))` are multiples of 64. Another rule of thumb is to specify destination ROI in such way that left and right sides of the projected image are separated from the ROI by at least 63 bytes from each side. However, this requires the whole ROI to be part of allocated memory. In case when the conditions above are not satisfied, the function may decrease in speed slightly and will return NPP_MISALIGNED_DST_ROI_WARNING warning.

Parameters:

pSrc Source-Image Pointer.

srcSize Size of source image in pixels

nSrcStep Source-Image Line Step.

srcRoi Source ROI

srcQuad Source quadrangle

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

dstRoi Destination ROI

dstQuad Destination quadrangle

interpolation Interpolation mode: can be NPPI_INTER_NN, NPPI_INTER_LINEAR or NPPI_-INTER_CUBIC

Returns:

Image Data Related Error Codes, ROI Related Error Codes

- NPP_RECT_ERROR Indicates an error condition if width or height of the intersection of the srcRoi and source image is less than or equal to 1
- NPP_WRONG_INTERSECTION_ROI_ERROR Indicates an error condition if srcRoi has no intersection with the source image
- NPP_INTERPOLATION_ERROR Indicates an error condition if interpolation has an illegal value
- NPP_COEFF_ERROR Indicates an error condition if coefficient values are invalid
- NPP_WRONG_INTERSECTION_QUAD_WARNING Indicates a warning that no operation is performed if the transformed source ROI has no intersection with the destination ROI
- NPP_MISALIGNED_DST_ROI_WARNING Indicates a warning that the speed of primitive execution was reduced due to destination ROI misalignment

7.4.1.297 NppStatus nppiWarpAffineQuad_16u_C3R (const Npp16u ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp16u ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Affine transform of an image (16bit unsigned integer, three channels).

See also:

nppiWarpAffineQuad_16u_C1R

7.4.1.298 NppStatus nppiWarpAffineQuad_16u_C4R (const Npp16u $*$ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp16u $*$ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Affine transform of an image (16bit unsigned integer, four channels).

See also:

nppiWarpAffineQuad_16u_C1R

7.4.1.299 NppStatus nppiWarpAffineQuad_16u_P3R (const Npp16u $*$ *pSrc*[3], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp16u $*$ *pDst*[3], int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Affine transform of an image (16bit unsigned integer, three planes).

See also:

nppiWarpAffineQuad_16u_C1R

7.4.1.300 NppStatus nppiWarpAffineQuad_16u_P4R (const Npp16u $*$ *pSrc*[4], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp16u $*$ *pDst*[4], int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Affine transform of an image (16bit unsigned integer, four planes).

See also:

nppiWarpAffineQuad_16u_C1R

7.4.1.301 NppStatus nppiWarpAffineQuad_32f_AC4R (const Npp32f $*$ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp32f $*$ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Affine transform of an image (32bit float, four channels RGBA).

See also:

nppiWarpAffineQuad_32f_C1R

7.4.1.302 NppStatus nppiWarpAffineQuad_32f_C1R (const Npp32f $*$ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp32f $*$ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Affine transform of an image (32bit float, single channel).

This function performs affine warping of a the specified quadrangle in the source image to the specified quadrangle in the destination image. The function nppiWarpAffineQuad uses the same formulas for pixel mapping as in nppiWarpAffine function. The transform coefficients are computed internally. The transformed part of the source image is resampled using the specified interpolation method and written to the destination ROI.

Parameters:

pSrc Source-Image Pointer.

srcSize Size of source image in pixels

nSrcStep Source-Image Line Step.

srcRoi Source ROI

srcQuad Source quadrangle

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

dstRoi Destination ROI

dstQuad Destination quadrangle

interpolation Interpolation mode: can be NPPI_INTER_NN, NPPI_INTER_LINEAR or NPPI_-INTER_CUBIC

Returns:

Image Data Related Error Codes, ROI Related Error Codes

- NPP_RECT_ERROR Indicates an error condition if width or height of the intersection of the srcRoi and source image is less than or equal to 1
- NPP_WRONG_INTERSECTION_ROI_ERROR Indicates an error condition if srcRoi has no intersection with the source image
- NPP_INTERPOLATION_ERROR Indicates an error condition if interpolation has an illegal value
- NPP_COEFF_ERROR Indicates an error condition if coefficient values are invalid
- NPP_WRONG_INTERSECTION_QUAD_WARNING Indicates a warning that no operation is performed if the transformed source ROI has no intersection with the destination ROI
- NPP_MISALIGNED_DST_ROI_WARNING Indicates a warning that the speed of primitive execution was reduced due to destination ROI misalignment

7.4.1.303 NppStatus nppiWarpAffineQuad_32f_C3R (const Npp32f ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp32f ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Affine transform of an image (32bit float, three channels).

See also:

nppiWarpAffineQuad_32f_C1R

7.4.1.304 NppStatus nppiWarpAffineQuad_32f_C4R (const Npp32f ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp32f ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Affine transform of an image (32bit float, four channels).

See also:

nppiWarpAffineQuad_32f_C1R

7.4.1.305 NppStatus nppiWarpAffineQuad_32f_P3R (const Npp32f ∗ *pSrc*[3], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp32f ∗ *pDst*[3], int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Affine transform of an image (32bit float, three planes).

See also:

nppiWarpAffineQuad_32f_C1R

7.4.1.306 NppStatus nppiWarpAffineQuad_32f_P4R (const Npp32f ∗ *pSrc*[4], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp32f ∗ *pDst*[4], int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Affine transform of an image (32bit float, four planes).

See also:

nppiWarpAffineQuad_32f_C1R

7.4.1.307 NppStatus nppiWarpAffineQuad_32s_AC4R (const Npp32s ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp32s ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Affine transform of an image (32bit signed integer, four channels RGBA).

See also:

nppiWarpAffineQuad_32s_C1R

7.4.1.308 NppStatus nppiWarpAffineQuad_32s_C1R (const Npp32s ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp32s ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Affine transform of an image (32bit signed integer, single channel).

This function performs affine warping of a the specified quadrangle in the source image to the specified quadrangle in the destination image. The function nppiWarpAffineQuad uses the same formulas for pixel mapping as in nppiWarpAffine function. The transform coefficients are computed internally. The transformed part of the source image is resampled using the specified interpolation method and written to the destination ROI.

Parameters:

pSrc Source-Image Pointer.

srcSize Size of source image in pixels

nSrcStep Source-Image Line Step.

srcRoi Source ROI

srcQuad Source quadrangle

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

dstRoi Destination ROI

dstQuad Destination quadrangle

interpolation Interpolation mode: can be NPPI_INTER_NN, NPPI_INTER_LINEAR or NPPI_-INTER_CUBIC

Returns:

Image Data Related Error Codes, ROI Related Error Codes

- NPP_RECT_ERROR Indicates an error condition if width or height of the intersection of the srcRoi and source image is less than or equal to 1
- NPP_WRONG_INTERSECTION_ROI_ERROR Indicates an error condition if srcRoi has no intersection with the source image
- NPP_INTERPOLATION_ERROR Indicates an error condition if interpolation has an illegal value
- NPP_COEFF_ERROR Indicates an error condition if coefficient values are invalid
- NPP_WRONG_INTERSECTION_QUAD_WARNING Indicates a warning that no operation is performed if the transformed source ROI has no intersection with the destination ROI
- NPP_MISALIGNED_DST_ROI_WARNING Indicates a warning that the speed of primitive execution was reduced due to destination ROI misalignment

7.4.1.309 NppStatus nppiWarpAffineQuad_32s_C3R (const Npp32s ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp32s ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Affine transform of an image (32bit signed integer, three channels).

See also:

nppiWarpAffineQuad_32s_C1R

7.4.1.310 NppStatus nppiWarpAffineQuad_32s_C4R (const Npp32s ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp32s ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Affine transform of an image (32bit signed integer, four channels).

See also:

nppiWarpAffineQuad_32s_C1R

7.4.1.311 NppStatus nppiWarpAffineQuad_32s_P3R (const Npp32s ∗ *pSrc*[3], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp32s ∗ *pDst*[3], int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Affine transform of an image (32bit signed integer, three planes).

See also:

nppiWarpAffineQuad_32s_C1R

7.4.1.312 NppStatus nppiWarpAffineQuad_32s_P4R (const Npp32s ∗ *pSrc*[4], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp32s ∗ *pDst*[4], int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Affine transform of an image (32bit signed integer, four planes).

See also:

nppiWarpAffineQuad_32s_C1R

7.4.1.313 NppStatus nppiWarpAffineQuad_8u_AC4R (const Npp8u ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp8u ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Affine transform of an image (8bit unsigned integer, four channels RGBA).

See also:

nppiWarpAffineQuad_8u_C1R

7.4.1.314 NppStatus nppiWarpAffineQuad_8u_C1R (const Npp8u ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp8u ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Affine transform of an image (8bit unsigned integer, single channel).

This function performs affine warping of a the specified quadrangle in the source image to the specified quadrangle in the destination image. The function nppiWarpAffineQuad uses the same formulas for pixel mapping as in nppiWarpAffine function. The transform coefficients are computed internally. The transformed part of the source image is resampled using the specified interpolation method and written to the destination ROI.

NPPI specific recommendation: The function operates using 2 types of kernels: fast and accurate. The fast method is about 4 times faster than its accurate variant, but does not perform memory access checks and requires the destination ROI to be 64 bytes aligned. Hence any destination ROI is chunked into 3 vertical stripes: the first and the third are processed by accurate kernels and the central one is processed by the fast one. In order to get the maximum available speed of execution, the projection of destination ROI onto image addresses must be 64 bytes aligned. This is always true if the values `(int)((void *)(pDst + dstRoi.x))` and `(int)((void *)(pDst + dstRoi.x + dstRoi.width))` are multiples of 64. Another rule of thumb is to specify destination ROI in such way that left and right sides of the projected image are separated from the ROI by at least 63 bytes from each side. However, this requires the whole ROI to be part of allocated memory. In case when the conditions above are not satisfied, the function may decrease in speed slightly and will return NPP_MISALIGNED_DST_ROI_WARNING warning.

Parameters:

pSrc Source-Image Pointer.

srcSize Size of source image in pixels

nSrcStep Source-Image Line Step.

srcRoi Source ROI

srcQuad Source quadrangle

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

dstRoi Destination ROI

dstQuad Destination quadrangle

interpolation Interpolation mode: can be NPPI_INTER_NN, NPPI_INTER_LINEAR or NPPI_-INTER_CUBIC

Returns:

Image Data Related Error Codes, ROI Related Error Codes

- NPP_RECT_ERROR Indicates an error condition if width or height of the intersection of the srcRoi and source image is less than or equal to 1
- NPP_WRONG_INTERSECTION_ROI_ERROR Indicates an error condition if srcRoi has no intersection with the source image
- NPP_INTERPOLATION_ERROR Indicates an error condition if interpolation has an illegal value
- NPP_COEFF_ERROR Indicates an error condition if coefficient values are invalid
- NPP_WRONG_INTERSECTION_QUAD_WARNING Indicates a warning that no operation is performed if the transformed source ROI has no intersection with the destination ROI
- NPP_MISALIGNED_DST_ROI_WARNING Indicates a warning that the speed of primitive execution was reduced due to destination ROI misalignment ignored, internally computed coordinates are used instead

7.4.1.315 NppStatus nppiWarpAffineQuad_8u_C3R (const Npp8u ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp8u ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Affine transform of an image (8bit unsigned integer, three channels).

See also:

nppiWarpAffineQuad_8u_C1R

7.4.1.316 NppStatus nppiWarpAffineQuad_8u_C4R (const Npp8u ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp8u ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Affine transform of an image (8bit unsigned integer, four channels).

See also:

nppiWarpAffineQuad_8u_C1R

7.4.1.317 NppStatus nppiWarpAffineQuad_8u_P3R (const Npp8u ∗ *pSrc*[3], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp8u ∗ *pDst*[3], int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Affine transform of an image (8bit unsigned integer, three planes).

See also:

nppiWarpAffineQuad_8u_C1R

7.4.1.318 NppStatus nppiWarpAffineQuad_8u_P4R (const Npp8u ∗ *pSrc*[4], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp8u ∗ *pDst*[4], int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Affine transform of an image (8bit unsigned integer, four planes).

See also:

nppiWarpAffineQuad_8u_C1R

7.4.1.319 NppStatus nppiWarpPerspective_16u_AC4R (const Npp16u ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp16u ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Perspective transform of an image (16bit unsigned integer, four channels RGBA).

See also:

nppiWarpPerspective_16u_C1R

7.4.1.320 NppStatus nppiWarpPerspective_16u_C1R (const Npp16u ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp16u ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Perspective transform of an image (16bit unsigned integer, single channel).

This function operates using given transform coefficients that can be obtained by using nppiGetPerspectiveTransform function or set explicitly. The function operates on source and destination regions of interest. The perspective warp function transforms the source image pixel coordinates (x, y) according to the following formulas:

$$X_{new} = \frac{C_{00} * x + C_{01} * y + C_{02}}{C_{20} * x + C_{21} * y + C_{22}} \qquad Y_{new} = \frac{C_{10} * x + C_{11} * y + C_{12}}{C_{20} * x + C_{21} * y + C_{22}}$$

The transformed part of the source image is resampled using the specified interpolation method and written to the destination ROI. The functions nppiGetPerspectiveQuad and nppiGetPerspectiveBound can help with destination ROI specification.

NPPI specific recommendation: The function operates using 2 types of kernels: fast and accurate. The fast method is about 4 times faster than its accurate variant, but does not perform memory access checks and requires the destination ROI to be 64 bytes aligned. Hence any destination ROI is chunked into 3 vertical stripes: the first and the third are processed by accurate kernels and the central one is processed by the fast one. In order to get the maximum available speed of execution, the projection of destination ROI onto image addresses must be 64 bytes aligned. This is always true if the values `(int)((void *)(pDst + dstRoi.x))` and `(int)((void *)(pDst + dstRoi.x + dstRoi.width))` are multiples of 64. Another rule of thumb is to specify destination ROI in such way that left and right sides of the projected image are separated from the ROI by at least 63 bytes from each side. However, this requires the whole ROI to be part of allocated memory. In case when the conditions above are not satisfied, the function may decrease in speed slightly and will return NPP_MISALIGNED_DST_ROI_WARNING warning.

Parameters:

pSrc Source-Image Pointer.

srcSize Size of source image in pixels

nSrcStep Source-Image Line Step.

srcRoi Source ROI

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

dstRoi Destination ROI

coeffs Perspective transform coefficients

interpolation Interpolation mode: can be NPPI_INTER_NN, NPPI_INTER_LINEAR or NPPI_-INTER_CUBIC

Returns:

Image Data Related Error Codes, ROI Related Error Codes

- NPP_RECT_ERROR Indicates an error condition if width or height of the intersection of the srcRoi and source image is less than or equal to 1
- NPP_WRONG_INTERSECTION_ROI_ERROR Indicates an error condition if srcRoi has no intersection with the source image
- NPP_INTERPOLATION_ERROR Indicates an error condition if interpolation has an illegal value
- NPP_COEFF_ERROR Indicates an error condition if coefficient values are invalid
- NPP_WRONG_INTERSECTION_QUAD_WARNING Indicates a warning that no operation is performed if the transformed source ROI has no intersection with the destination ROI
- NPP_MISALIGNED_DST_ROI_WARNING Indicates a warning that the speed of primitive execution was reduced due to destination ROI

7.4.1.321 NppStatus nppiWarpPerspective_16u_C3R (const Npp16u $*$ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp16u $*$ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Perspective transform of an image (16bit unsigned integer, three channels).

See also:

nppiWarpPerspective_16u_C1R

7.4.1.322 NppStatus nppiWarpPerspective_16u_C4R (const Npp16u $*$ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp16u $*$ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Perspective transform of an image (16bit unsigned integer, four channels).

See also:

nppiWarpPerspective_16u_C1R

7.4.1.323 NppStatus nppiWarpPerspective_16u_P3R (const Npp16u $*$ *pSrc*[3], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp16u $*$ *pDst*[3], int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Perspective transform of an image (16bit unsigned integer, three planes).

See also:

nppiWarpPerspective_16u_C1R

7.4.1.324 NppStatus nppiWarpPerspective_16u_P4R (const Npp16u $*$ *pSrc*[4], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp16u $*$ *pDst*[4], int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Perspective transform of an image (16bit unsigned integer, four planes).

See also:

nppiWarpPerspective_16u_C1R

7.4.1.325 NppStatus nppiWarpPerspective_32f_AC4R (const Npp32f $*$ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32f $*$ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Perspective transform of an image (32bit float, four channels RGBA).

See also:

nppiWarpPerspective_32f_C1R

7.4.1.326 NppStatus nppiWarpPerspective_32f_C1R (const Npp32f $*$ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32f $*$ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Perspective transform of an image (32bit float, single channel).

This function operates using given transform coefficients that can be obtained by using nppiGetPerspectiveTransform function or set explicitly. The function operates on source and destination regions of interest. The perspective warp function transforms the source image pixel coordinates (x, y) according to the following formulas:

$$X_{new} = \frac{C_{00} * x + C_{01} * y + C_{02}}{C_{20} * x + C_{21} * y + C_{22}} \qquad Y_{new} = \frac{C_{10} * x + C_{11} * y + C_{12}}{C_{20} * x + C_{21} * y + C_{22}}$$

The transformed part of the source image is resampled using the specified interpolation method and written to the destination ROI. The functions nppiGetPerspectiveQuad and nppiGetPerspectiveBound can help with destination ROI specification.

Parameters:

pSrc Source-Image Pointer.

srcSize Size of source image in pixels

nSrcStep Source-Image Line Step.

srcRoi Source ROI

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

dstRoi Destination ROI

coeffs Perspective transform coefficients

interpolation Interpolation mode: can be NPPI_INTER_NN, NPPI_INTER_LINEAR or NPPI_-INTER_CUBIC

Returns:

Image Data Related Error Codes, ROI Related Error Codes

- NPP_RECT_ERROR Indicates an error condition if width or height of the intersection of the srcRoi and source image is less than or equal to 1
- NPP_WRONG_INTERSECTION_ROI_ERROR Indicates an error condition if srcRoi has no intersection with the source image
- NPP_INTERPOLATION_ERROR Indicates an error condition if interpolation has an illegal value
- NPP_COEFF_ERROR Indicates an error condition if coefficient values are invalid
- NPP_WRONG_INTERSECTION_QUAD_WARNING Indicates a warning that no operation is performed if the transformed source ROI has no intersection with the destination ROI
- NPP_MISALIGNED_DST_ROI_WARNING Indicates a warning that the speed of primitive execution was reduced due to destination ROI

7.4.1.327 NppStatus nppiWarpPerspective_32f_C3R (const Npp32f ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32f ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Perspective transform of an image (32bit float, three channels).

See also:

nppiWarpPerspective_32f_C1R

7.4.1.328 NppStatus nppiWarpPerspective_32f_C4R (const Npp32f ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32f ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Perspective transform of an image (32bit float, four channels).

See also:

nppiWarpPerspective_32f_C1R

7.4.1.329 NppStatus nppiWarpPerspective_32f_P3R (const Npp32f $*$ *pSrc*[3], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32f $*$ *pDst*[3], int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Perspective transform of an image (32bit float, three planes).

See also:

nppiWarpPerspective_32f_C1R

7.4.1.330 NppStatus nppiWarpPerspective_32f_P4R (const Npp32f $*$ *pSrc*[4], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32f $*$ *pDst*[4], int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Perspective transform of an image (32bit float, four planes).

See also:

nppiWarpPerspective_32f_C1R

7.4.1.331 NppStatus nppiWarpPerspective_32s_AC4R (const Npp32s $*$ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32s $*$ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Perspective transform of an image (32bit signed integer, four channels RGBA).

See also:

nppiWarpPerspective_32s_C1R

7.4.1.332 NppStatus nppiWarpPerspective_32s_C1R (const Npp32s $*$ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32s $*$ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Perspective transform of an image (32bit signed integer, single channel).

This function operates using given transform coefficients that can be obtained by using nppiGetPerspectiveTransform function or set explicitly. The function operates on source and destination regions of interest. The perspective warp function transforms the source image pixel coordinates (x, y) according to the following formulas:

$$X_{new} = \frac{C_{00} * x + C_{01} * y + C_{02}}{C_{20} * x + C_{21} * y + C_{22}} \qquad Y_{new} = \frac{C_{10} * x + C_{11} * y + C_{12}}{C_{20} * x + C_{21} * y + C_{22}}$$

The transformed part of the source image is resampled using the specified interpolation method and written to the destination ROI. The functions nppiGetPerspectiveQuad and nppiGetPerspectiveBound can help with destination ROI specification.

Parameters:

pSrc Source-Image Pointer.

srcSize Size of source image in pixels

nSrcStep Source-Image Line Step.

srcRoi Source ROI

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

dstRoi Destination ROI

coeffs Perspective transform coefficients

interpolation Interpolation mode: can be NPPI_INTER_NN, NPPI_INTER_LINEAR or NPPI_-INTER_CUBIC

Returns:

Image Data Related Error Codes, ROI Related Error Codes

- NPP_RECT_ERROR Indicates an error condition if width or height of the intersection of the srcRoi and source image is less than or equal to 1
- NPP_WRONG_INTERSECTION_ROI_ERROR Indicates an error condition if srcRoi has no intersection with the source image
- NPP_INTERPOLATION_ERROR Indicates an error condition if interpolation has an illegal value
- NPP_COEFF_ERROR Indicates an error condition if coefficient values are invalid
- NPP_WRONG_INTERSECTION_QUAD_WARNING Indicates a warning that no operation is performed if the transformed source ROI has no intersection with the destination ROI
- NPP_MISALIGNED_DST_ROI_WARNING Indicates a warning that the speed of primitive execution was reduced due to destination ROI

7.4.1.333 NppStatus nppiWarpPerspective_32s_C3R (const Npp32s ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32s ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Perspective transform of an image (32bit signed integer, three channels).

See also:

nppiWarpPerspective_32s_C1R

7.4.1.334 NppStatus nppiWarpPerspective_32s_C4R (const Npp32s ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32s ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Perspective transform of an image (32bit signed integer, four channels).

See also:

nppiWarpPerspective_32s_C1R

7.4.1.335 NppStatus nppiWarpPerspective_32s_P3R (const Npp32s ∗ *pSrc*[3], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32s ∗ *pDst*[3], int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Perspective transform of an image (32bit signed integer, three planes).

See also:

nppiWarpPerspective_32s_C1R

7.4.1.336 NppStatus nppiWarpPerspective_32s_P4R (const Npp32s ∗ *pSrc*[4], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32s ∗ *pDst*[4], int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Perspective transform of an image (32bit signed integer, four planes).

See also:

nppiWarpPerspective_32s_C1R

7.4.1.337 NppStatus nppiWarpPerspective_8u_AC4R (const Npp8u ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp8u ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Perspective transform of an image (8bit unsigned integer, four channels RGBA).

See also:

nppiWarpPerspective_8u_C1R

7.4.1.338 NppStatus nppiWarpPerspective_8u_C1R (const Npp8u ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp8u ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Perspective transform of an image (8bit unsigned integer, single channel).

This function operates using given transform coefficients that can be obtained by using nppiGetPerspectiveTransform function or set explicitly. The function operates on source and destination regions of interest. The perspective warp function transforms the source image pixel coordinates (x, y) according to the following formulas:

$$X_{new} = \frac{C_{00} * x + C_{01} * y + C_{02}}{C_{20} * x + C_{21} * y + C_{22}} \qquad Y_{new} = \frac{C_{10} * x + C_{11} * y + C_{12}}{C_{20} * x + C_{21} * y + C_{22}}$$

The transformed part of the source image is resampled using the specified interpolation method and written to the destination ROI. The functions nppiGetPerspectiveQuad and nppiGetPerspectiveBound can help with destination ROI specification.

NPPI specific recommendation: The function operates using 2 types of kernels: fast and accurate. The fast method is about 4 times faster than its accurate variant, but does not perform memory access checks and requires the destination ROI to be 64 bytes aligned. Hence any destination ROI is chunked into 3 vertical stripes: the first and the third are processed by accurate kernels and the central one is processed by the fast one. In order to get the maximum available speed of execution, the projection of destination ROI onto image addresses must be 64 bytes aligned. This is always true if the values `(int)((void *)(pDst + dstRoi.x))` and `(int)((void *)(pDst + dstRoi.x + dstRoi.width))` are multiples of 64. Another rule of thumb is to specify destination ROI in such way that left and right sides of the projected image are separated from the ROI by at least 63 bytes from each side. However, this requires the whole ROI to be part of allocated memory. In case when the conditions above are not satisfied, the function may decrease in speed slightly and will return NPP_MISALIGNED_DST_ROI_WARNING warning.

Parameters:

pSrc Source-Image Pointer.

srcSize Size of source image in pixels

nSrcStep Source-Image Line Step.

srcRoi Source ROI

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

dstRoi Destination ROI

coeffs Perspective transform coefficients

interpolation Interpolation mode: can be NPPI_INTER_NN, NPPI_INTER_LINEAR or NPPI_-INTER_CUBIC

Returns:

Image Data Related Error Codes, ROI Related Error Codes

- NPP_RECT_ERROR Indicates an error condition if width or height of the intersection of the srcRoi and source image is less than or equal to 1
- NPP_WRONG_INTERSECTION_ROI_ERROR Indicates an error condition if srcRoi has no intersection with the source image
- NPP_INTERPOLATION_ERROR Indicates an error condition if interpolation has an illegal value
- NPP_COEFF_ERROR Indicates an error condition if coefficient values are invalid
- NPP_WRONG_INTERSECTION_QUAD_WARNING Indicates a warning that no operation is performed if the transformed source ROI has no intersection with the destination ROI
- NPP_MISALIGNED_DST_ROI_WARNING Indicates a warning that the speed of primitive execution was reduced due to destination ROI

7.4.1.339 NppStatus nppiWarpPerspective_8u_C3R (const Npp8u * *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp8u * *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Perspective transform of an image (8bit unsigned integer, three channels).

See also:

nppiWarpPerspective_8u_C1R

7.4.1.340 NppStatus nppiWarpPerspective_8u_C4R (const Npp8u ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp8u ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Perspective transform of an image (8bit unsigned integer, four channels).

See also:

nppiWarpPerspective_8u_C1R

7.4.1.341 NppStatus nppiWarpPerspective_8u_P3R (const Npp8u ∗ *pSrc*[3], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp8u ∗ *pDst*[3], int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Perspective transform of an image (8bit unsigned integer, three planes).

See also:

nppiWarpPerspective_8u_C1R

7.4.1.342 NppStatus nppiWarpPerspective_8u_P4R (const Npp8u ∗ *pSrc*[4], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp8u ∗ *pDst*[4], int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Perspective transform of an image (8bit unsigned integer, four planes).

See also:

nppiWarpPerspective_8u_C1R

7.4.1.343 NppStatus nppiWarpPerspectiveBack_16u_AC4R (const Npp16u ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp16u ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Inverse perspective transform of an image (16bit unsigned integer, four channels RGBA).

See also:

nppiWarpPerspectiveBack_16u_C1R

7.4.1.344 NppStatus nppiWarpPerspectiveBack_16u_C1R (const Npp16u ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp16u ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Inverse perspective transform of an image (16bit unsigned integer, single channel).

This function operates using given transform coefficients that can be obtained by using nppiGetPerspective-Transform function or set explicitly. Thus there is no need to invert coefficients in your application before calling WarpPerspectiveBack. The function operates on source and destination regions of interest. The perspective warp function transforms the source image pixel coordinates (x, y) according to the following formulas:

$$\frac{C_{00} * X_{new} + C_{01} * Y_{new} + C_{02}}{C_{20} * X_{new} + C_{21} * Y_{new} + C_{22}} = x \qquad \frac{C_{10} * X_{new} + C_{11} * Y_{new} + C_{12}}{C_{20} * X_{new} + C_{21} * Y_{new} + C_{22}} = y$$

The transformed part of the source image is resampled using the specified interpolation method and written to the destination ROI. The functions nppiGetPerspectiveQuad and nppiGetPerspectiveBound can help with destination ROI specification.

NPPI specific recommendation: The function operates using 2 types of kernels: fast and accurate. The fast method is about 4 times faster than its accurate variant, but does not perform memory access checks and requires the destination ROI to be 64 bytes aligned. Hence any destination ROI is chunked into 3 vertical stripes: the first and the third are processed by accurate kernels and the central one is processed by the fast one. In order to get the maximum available speed of execution, the projection of destination ROI onto image addresses must be 64 bytes aligned. This is always true if the values `(int)((void *)(pDst + dstRoi.x))` and `(int)((void *)(pDst + dstRoi.x + dstRoi.width))` are multiples of 64. Another rule of thumb is to specify destination ROI in such way that left and right sides of the projected image are separated from the ROI by at least 63 bytes from each side. However, this requires the whole ROI to be part of allocated memory. In case when the conditions above are not satisfied, the function may decrease in speed slightly and will return NPP_MISALIGNED_DST_ROI_WARNING warning.

Parameters:

pSrc Source-Image Pointer.

srcSize Size of source image in pixels

nSrcStep Source-Image Line Step.

srcRoi Source ROI

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

dstRoi Destination ROI

coeffs Perspective transform coefficients

interpolation Interpolation mode: can be NPPI_INTER_NN, NPPI_INTER_LINEAR or NPPI_-INTER_CUBIC

Returns:

Image Data Related Error Codes, ROI Related Error Codes

- NPP_RECT_ERROR Indicates an error condition if width or height of the intersection of the srcRoi and source image is less than or equal to 1
- NPP_WRONG_INTERSECTION_ROI_ERROR Indicates an error condition if srcRoi has no intersection with the source image
- NPP_INTERPOLATION_ERROR Indicates an error condition if interpolation has an illegal value
- NPP_COEFF_ERROR Indicates an error condition if coefficient values are invalid
- NPP_WRONG_INTERSECTION_QUAD_WARNING Indicates a warning that no operation is performed if the transformed source ROI has no intersection with the destination ROI
- NPP_MISALIGNED_DST_ROI_WARNING Indicates a warning that the speed of primitive execution was reduced due to destination ROI

7.4.1.345 NppStatus nppiWarpPerspectiveBack_16u_C3R (const Npp16u ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp16u ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Inverse perspective transform of an image (16bit unsigned integer, three channels).

See also:

nppiWarpPerspectiveBack_16u_C1R

7.4.1.346 NppStatus nppiWarpPerspectiveBack_16u_C4R (const Npp16u ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp16u ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Inverse perspective transform of an image (16bit unsigned integer, four channels).

See also:

nppiWarpPerspectiveBack_16u_C1R

7.4.1.347 NppStatus nppiWarpPerspectiveBack_16u_P3R (const Npp16u ∗ *pSrc*[3], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp16u ∗ *pDst*[3], int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Inverse perspective transform of an image (16bit unsigned integer, three planes).

See also:

nppiWarpPerspectiveBack_16u_C1R

7.4.1.348 NppStatus nppiWarpPerspectiveBack_16u_P4R (const Npp16u ∗ *pSrc*[4], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp16u ∗ *pDst*[4], int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Inverse perspective transform of an image (16bit unsigned integer, four planes).

See also:

nppiWarpPerspectiveBack_16u_C1R

7.4.1.349 NppStatus nppiWarpPerspectiveBack_32f_AC4R (const Npp32f ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32f ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Inverse perspective transform of an image (32bit float, four channels RGBA).

See also:

nppiWarpPerspectiveBack_32f_C1R

7.4.1.350 NppStatus nppiWarpPerspectiveBack_32f_C1R (const Npp32f $*$ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32f $*$ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Inverse perspective transform of an image (32bit float, single channel).

This function operates using given transform coefficients that can be obtained by using nppiGetPerspective-Transform function or set explicitly. Thus there is no need to invert coefficients in your application before calling WarpPerspectiveBack. The function operates on source and destination regions of interest. The perspective warp function transforms the source image pixel coordinates (x, y) according to the following formulas:

$$\frac{C_{00} * X_{new} + C_{01} * Y_{new} + C_{02}}{C_{20} * X_{new} + C_{21} * Y_{new} + C_{22}} = x \qquad \frac{C_{10} * X_{new} + C_{11} * Y_{new} + C_{12}}{C_{20} * X_{new} + C_{21} * Y_{new} + C_{22}} = y$$

The transformed part of the source image is resampled using the specified interpolation method and written to the destination ROI. The functions nppiGetPerspectiveQuad and nppiGetPerspectiveBound can help with destination ROI specification.

Parameters:

pSrc Source-Image Pointer.

srcSize Size of source image in pixels

nSrcStep Source-Image Line Step.

srcRoi Source ROI

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

dstRoi Destination ROI

coeffs Perspective transform coefficients

interpolation Interpolation mode: can be NPPI_INTER_NN, NPPI_INTER_LINEAR or NPPI_-INTER_CUBIC

Returns:

Image Data Related Error Codes, ROI Related Error Codes

- NPP_RECT_ERROR Indicates an error condition if width or height of the intersection of the srcRoi and source image is less than or equal to 1
- NPP_WRONG_INTERSECTION_ROI_ERROR Indicates an error condition if srcRoi has no intersection with the source image
- NPP_INTERPOLATION_ERROR Indicates an error condition if interpolation has an illegal value
- NPP_COEFF_ERROR Indicates an error condition if coefficient values are invalid
- NPP_WRONG_INTERSECTION_QUAD_WARNING Indicates a warning that no operation is performed if the transformed source ROI has no intersection with the destination ROI
- NPP_MISALIGNED_DST_ROI_WARNING Indicates a warning that the speed of primitive execution was reduced due to destination ROI

7.4.1.351 NppStatus nppiWarpPerspectiveBack_32f_C3R (const Npp32f $*$ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32f $*$ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Inverse perspective transform of an image (32bit float, three channels).

See also:

nppiWarpPerspectiveBack_32f_C1R

7.4.1.352 NppStatus nppiWarpPerspectiveBack_32f_C4R (const Npp32f $*$ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32f $*$ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Inverse perspective transform of an image (32bit float, four channels).

See also:

nppiWarpPerspectiveBack_32f_C1R

7.4.1.353 NppStatus nppiWarpPerspectiveBack_32f_P3R (const Npp32f $*$ *pSrc*[3], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32f $*$ *pDst*[3], int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Inverse perspective transform of an image (32bit float, three planes).

See also:

nppiWarpPerspectiveBack_32f_C1R

7.4.1.354 NppStatus nppiWarpPerspectiveBack_32f_P4R (const Npp32f $*$ *pSrc*[4], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32f $*$ *pDst*[4], int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Inverse perspective transform of an image (32bit float, four planes).

See also:

nppiWarpPerspectiveBack_32f_C1R

7.4.1.355 NppStatus nppiWarpPerspectiveBack_32s_AC4R (const Npp32s $*$ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32s $*$ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Inverse perspective transform of an image (32bit signed integer, four channels RGBA).

See also:

nppiWarpPerspectiveBack_32s_C1R

7.4.1.356 NppStatus nppiWarpPerspectiveBack_32s_C1R (const Npp32s ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32s ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Inverse perspective transform of an image (32bit signed integer, single channel).

This function operates using given transform coefficients that can be obtained by using nppiGetPerspective-Transform function or set explicitly. Thus there is no need to invert coefficients in your application before calling WarpPerspectiveBack. The function operates on source and destination regions of interest. The perspective warp function transforms the source image pixel coordinates (x, y) according to the following formulas:

$$\frac{C_{00} * X_{new} + C_{01} * Y_{new} + C_{02}}{C_{20} * X_{new} + C_{21} * Y_{new} + C_{22}} = x \qquad \frac{C_{10} * X_{new} + C_{11} * Y_{new} + C_{12}}{C_{20} * X_{new} + C_{21} * Y_{new} + C_{22}} = y$$

The transformed part of the source image is resampled using the specified interpolation method and written to the destination ROI. The functions nppiGetPerspectiveQuad and nppiGetPerspectiveBound can help with destination ROI specification.

Parameters:

pSrc Source-Image Pointer.

srcSize Size of source image in pixels

nSrcStep Source-Image Line Step.

srcRoi Source ROI

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

dstRoi Destination ROI

coeffs Perspective transform coefficients

interpolation Interpolation mode: can be NPPI_INTER_NN, NPPI_INTER_LINEAR or NPPI_-INTER_CUBIC

Returns:

Image Data Related Error Codes, ROI Related Error Codes

- NPP_RECT_ERROR Indicates an error condition if width or height of the intersection of the srcRoi and source image is less than or equal to 1
- NPP_WRONG_INTERSECTION_ROI_ERROR Indicates an error condition if srcRoi has no intersection with the source image
- NPP_INTERPOLATION_ERROR Indicates an error condition if interpolation has an illegal value
- NPP_COEFF_ERROR Indicates an error condition if coefficient values are invalid
- NPP_WRONG_INTERSECTION_QUAD_WARNING Indicates a warning that no operation is performed if the transformed source ROI has no intersection with the destination ROI
- NPP_MISALIGNED_DST_ROI_WARNING Indicates a warning that the speed of primitive execution was reduced due to destination ROI

7.4.1.357 NppStatus nppiWarpPerspectiveBack_32s_C3R (const Npp32s ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32s ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Inverse perspective transform of an image (32bit signed integer, three channels).

See also:

nppiWarpPerspectiveBack_32s_C1R

7.4.1.358 NppStatus nppiWarpPerspectiveBack_32s_C4R (const Npp32s ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32s ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Inverse perspective transform of an image (32bit signed integer, four channels).

See also:

nppiWarpPerspectiveBack_32s_C1R

7.4.1.359 NppStatus nppiWarpPerspectiveBack_32s_P3R (const Npp32s ∗ *pSrc*[3], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32s ∗ *pDst*[3], int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Inverse perspective transform of an image (32bit signed integer, three planes).

See also:

nppiWarpPerspectiveBack_32s_C1R

7.4.1.360 NppStatus nppiWarpPerspectiveBack_32s_P4R (const Npp32s ∗ *pSrc*[4], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp32s ∗ *pDst*[4], int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Inverse perspective transform of an image (32bit signed integer, four planes).

See also:

nppiWarpPerspectiveBack_32s_C1R

7.4.1.361 NppStatus nppiWarpPerspectiveBack_8u_AC4R (const Npp8u ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp8u ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Inverse perspective transform of an image (8bit unsigned integer, four channels RGBA).

See also:

nppiWarpPerspectiveBack_8u_C1R

7.4.1.362 NppStatus nppiWarpPerspectiveBack_8u_C1R (const Npp8u ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp8u ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Inverse perspective transform of an image (8bit unsigned integer, single channel).

This function operates using given transform coefficients that can be obtained by using nppiGetPerspective-Transform function or set explicitly. Thus there is no need to invert coefficients in your application before calling WarpPerspectiveBack. The function operates on source and destination regions of interest. The perspective warp function transforms the source image pixel coordinates (x, y) according to the following formulas:

$$\frac{C_{00} * X_{new} + C_{01} * Y_{new} + C_{02}}{C_{20} * X_{new} + C_{21} * Y_{new} + C_{22}} = x \qquad \frac{C_{10} * X_{new} + C_{11} * Y_{new} + C_{12}}{C_{20} * X_{new} + C_{21} * Y_{new} + C_{22}} = y$$

The transformed part of the source image is resampled using the specified interpolation method and written to the destination ROI. The functions nppiGetPerspectiveQuad and nppiGetPerspectiveBound can help with destination ROI specification.

NPPI specific recommendation: The function operates using 2 types of kernels: fast and accurate. The fast method is about 4 times faster than its accurate variant, but does not perform memory access checks and requires the destination ROI to be 64 bytes aligned. Hence any destination ROI is chunked into 3 vertical stripes: the first and the third are processed by accurate kernels and the central one is processed by the fast one. In order to get the maximum available speed of execution, the projection of destination ROI onto image addresses must be 64 bytes aligned. This is always true if the values `(int)((void *)(pDst + dstRoi.x))` and `(int)((void *)(pDst + dstRoi.x + dstRoi.width))` are multiples of 64. Another rule of thumb is to specify destination ROI in such way that left and right sides of the projected image are separated from the ROI by at least 63 bytes from each side. However, this requires the whole ROI to be part of allocated memory. In case when the conditions above are not satisfied, the function may decrease in speed slightly and will return NPP_MISALIGNED_DST_ROI_WARNING warning.

Parameters:

pSrc Source-Image Pointer.

srcSize Size of source image in pixels

nSrcStep Source-Image Line Step.

srcRoi Source ROI

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

dstRoi Destination ROI

coeffs Perspective transform coefficients

interpolation Interpolation mode: can be NPPI_INTER_NN, NPPI_INTER_LINEAR or NPPI_-INTER_CUBIC

Returns:

Image Data Related Error Codes, ROI Related Error Codes

- NPP_RECT_ERROR Indicates an error condition if width or height of the intersection of the srcRoi and source image is less than or equal to 1
- NPP_WRONG_INTERSECTION_ROI_ERROR Indicates an error condition if srcRoi has no intersection with the source image
- NPP_INTERPOLATION_ERROR Indicates an error condition if interpolation has an illegal value

- NPP_COEFF_ERROR Indicates an error condition if coefficient values are invalid
- NPP_WRONG_INTERSECTION_QUAD_WARNING Indicates a warning that no operation is performed if the transformed source ROI has no intersection with the destination ROI
- NPP_MISALIGNED_DST_ROI_WARNING Indicates a warning that the speed of primitive execution was reduced due to destination ROI

7.4.1.363 NppStatus nppiWarpPerspectiveBack_8u_C3R (const Npp8u ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp8u ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Inverse perspective transform of an image (8bit unsigned integer, three channels).

See also:

nppiWarpPerspectiveBack_8u_C1R

7.4.1.364 NppStatus nppiWarpPerspectiveBack_8u_C4R (const Npp8u ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp8u ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Inverse perspective transform of an image (8bit unsigned integer, four channels).

See also:

nppiWarpPerspectiveBack_8u_C1R

7.4.1.365 NppStatus nppiWarpPerspectiveBack_8u_P3R (const Npp8u ∗ *pSrc*[3], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp8u ∗ *pDst*[3], int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Inverse perspective transform of an image (8bit unsigned integer, three planes).

See also:

nppiWarpPerspectiveBack_8u_C1R

7.4.1.366 NppStatus nppiWarpPerspectiveBack_8u_P4R (const Npp8u ∗ *pSrc*[4], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, Npp8u ∗ *pDst*[4], int *nDstStep*, NppiRect *dstRoi*, const double *coeffs*[3][3], int *interpolation*)

Inverse perspective transform of an image (8bit unsigned integer, four planes).

See also:

nppiWarpPerspectiveBack_8u_C1R

7.4.1.367 NppStatus nppiWarpPerspectiveQuad_16u_AC4R (const Npp16u ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp16u ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Perspective transform of an image (16bit unsigned integer, four channels RGBA).

See also:

nppiWarpPerspectiveQuad_16u_C1R

7.4.1.368 NppStatus nppiWarpPerspectiveQuad_16u_C1R (const Npp16u ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp16u ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Perspective transform of an image (16bit unsigned integer, single channel).

This function performs perspective warping of a the specified quadrangle in the source image to the specified quadrangle in the destination image. The function nppiWarpPerspectiveQuad uses the same formulas for pixel mapping as in nppiWarpPerspective function. The transform coefficients are computed internally. The transformed part of the source image is resampled using the specified interpolation method and written to the destination ROI.

NPPI specific recommendation: The function operates using 2 types of kernels: fast and accurate. The fast method is about 4 times faster than its accurate variant, but does not perform memory access checks and requires the destination ROI to be 64 bytes aligned. Hence any destination ROI is chunked into 3 vertical stripes: the first and the third are processed by accurate kernels and the central one is processed by the fast one. In order to get the maximum available speed of execution, the projection of destination ROI onto image addresses must be 64 bytes aligned. This is always true if the values `(int)((void *)(pDst + dstRoi.x))` and `(int)((void *)(pDst + dstRoi.x + dstRoi.width))` are multiples of 64. Another rule of thumb is to specify destination ROI in such way that left and right sides of the projected image are separated from the ROI by at least 63 bytes from each side. However, this requires the whole ROI to be part of allocated memory. In case when the conditions above are not satisfied, the function may decrease in speed slightly and will return NPP_MISALIGNED_DST_ROI_WARNING warning.

Parameters:

pSrc Source-Image Pointer.

srcSize Size of source image in pixels

nSrcStep Source-Image Line Step.

srcRoi Source ROI

srcQuad Source quadrangle

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

dstRoi Destination ROI

dstQuad Destination quadrangle

interpolation Interpolation mode: can be NPPI_INTER_NN, NPPI_INTER_LINEAR or NPPI_-INTER_CUBIC

Returns:

Image Data Related Error Codes, ROI Related Error Codes

- NPP_RECT_ERROR Indicates an error condition if width or height of the intersection of the srcRoi and source image is less than or equal to 1
- NPP_WRONG_INTERSECTION_ROI_ERROR Indicates an error condition if srcRoi has no intersection with the source image
- NPP_INTERPOLATION_ERROR Indicates an error condition if interpolation has an illegal value
- NPP_COEFF_ERROR Indicates an error condition if coefficient values are invalid
- NPP_WRONG_INTERSECTION_QUAD_WARNING Indicates a warning that no operation is performed if the transformed source ROI has no intersection with the destination ROI
- NPP_MISALIGNED_DST_ROI_WARNING Indicates a warning that the speed of primitive execution was reduced due to destination ROI

7.4.1.369 NppStatus nppiWarpPerspectiveQuad_16u_C3R (const Npp16u ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp16u ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Perspective transform of an image (16bit unsigned integer, three channels).

See also:

nppiWarpPerspectiveQuad_16u_C1R

7.4.1.370 NppStatus nppiWarpPerspectiveQuad_16u_C4R (const Npp16u ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp16u ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Perspective transform of an image (16bit unsigned integer, four channels).

See also:

nppiWarpPerspectiveQuad_16u_C1R

7.4.1.371 NppStatus nppiWarpPerspectiveQuad_16u_P3R (const Npp16u ∗ *pSrc*[3], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp16u ∗ *pDst*[3], int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Perspective transform of an image (16bit unsigned integer, three planes).

See also:

nppiWarpPerspectiveQuad_16u_C1R

7.4.1.372 NppStatus nppiWarpPerspectiveQuad_16u_P4R (const Npp16u ∗ *pSrc*[4], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp16u ∗ *pDst*[4], int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Perspective transform of an image (16bit unsigned integer, four planes).

See also:

nppiWarpPerspectiveQuad_16u_C1R

7.4.1.373 NppStatus nppiWarpPerspectiveQuad_32f_AC4R (const Npp32f ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp32f ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Perspective transform of an image (32bit float, four channels RGBA).

See also:

nppiWarpPerspectiveQuad_32f_C1R

7.4.1.374 NppStatus nppiWarpPerspectiveQuad_32f_C1R (const Npp32f ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp32f ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Perspective transform of an image (32bit float, single channel).

This function performs perspective warping of a the specified quadrangle in the source image to the specified quadrangle in the destination image. The function nppiWarpPerspectiveQuad uses the same formulas for pixel mapping as in nppiWarpPerspective function. The transform coefficients are computed internally. The transformed part of the source image is resampled using the specified interpolation method and written to the destination ROI.

Parameters:

pSrc Source-Image Pointer.

srcSize Size of source image in pixels

nSrcStep Source-Image Line Step.

srcRoi Source ROI

srcQuad Source quadrangle

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

dstRoi Destination ROI

dstQuad Destination quadrangle

interpolation Interpolation mode: can be NPPI_INTER_NN, NPPI_INTER_LINEAR or NPPI_-INTER_CUBIC

Returns:

Image Data Related Error Codes, ROI Related Error Codes

- NPP_RECT_ERROR Indicates an error condition if width or height of the intersection of the srcRoi and source image is less than or equal to 1
- NPP_WRONG_INTERSECTION_ROI_ERROR Indicates an error condition if srcRoi has no intersection with the source image
- NPP_INTERPOLATION_ERROR Indicates an error condition if interpolation has an illegal value
- NPP_COEFF_ERROR Indicates an error condition if coefficient values are invalid
- NPP_WRONG_INTERSECTION_QUAD_WARNING Indicates a warning that no operation is performed if the transformed source ROI has no intersection with the destination ROI
- NPP_MISALIGNED_DST_ROI_WARNING Indicates a warning that the speed of primitive execution was reduced due to destination ROI

7.4.1.375 NppStatus nppiWarpPerspectiveQuad_32f_C3R (const Npp32f ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp32f ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Perspective transform of an image (32bit float, three channels).

See also:

nppiWarpPerspectiveQuad_32f_C1R

7.4.1.376 NppStatus nppiWarpPerspectiveQuad_32f_C4R (const Npp32f ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp32f ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Perspective transform of an image (32bit float, four channels).

See also:

nppiWarpPerspectiveQuad_32f_C1R

7.4.1.377 NppStatus nppiWarpPerspectiveQuad_32f_P3R (const Npp32f ∗ *pSrc*[3], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp32f ∗ *pDst*[3], int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Perspective transform of an image (32bit float, three planes).

See also:

nppiWarpPerspectiveQuad_32f_C1R

7.4.1.378 NppStatus nppiWarpPerspectiveQuad_32f_P4R (const Npp32f ∗ *pSrc*[4], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp32f ∗ *pDst*[4], int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Perspective transform of an image (32bit float, four planes).

See also:

nppiWarpPerspectiveQuad_32f_C1R

7.4.1.379 NppStatus nppiWarpPerspectiveQuad_32s_AC4R (const Npp32s ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp32s ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Perspective transform of an image (32bit signed integer, four channels RGBA).

See also:

nppiWarpPerspectiveQuad_32s_C1R

7.4.1.380 NppStatus nppiWarpPerspectiveQuad_32s_C1R (const Npp32s ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp32s ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Perspective transform of an image (32bit signed integer, single channel).

This function performs perspective warping of a the specified quadrangle in the source image to the specified quadrangle in the destination image. The function nppiWarpPerspectiveQuad uses the same formulas for pixel mapping as in nppiWarpPerspective function. The transform coefficients are computed internally. The transformed part of the source image is resampled using the specified interpolation method and written to the destination ROI.

Parameters:

pSrc Source-Image Pointer.

srcSize Size of source image in pixels

nSrcStep Source-Image Line Step.

srcRoi Source ROI

srcQuad Source quadrangle

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

dstRoi Destination ROI

dstQuad Destination quadrangle

interpolation Interpolation mode: can be NPPI_INTER_NN, NPPI_INTER_LINEAR or NPPI_-INTER_CUBIC

Returns:

Image Data Related Error Codes, ROI Related Error Codes

- NPP_RECT_ERROR Indicates an error condition if width or height of the intersection of the srcRoi and source image is less than or equal to 1
- NPP_WRONG_INTERSECTION_ROI_ERROR Indicates an error condition if srcRoi has no intersection with the source image
- NPP_INTERPOLATION_ERROR Indicates an error condition if interpolation has an illegal value
- NPP_COEFF_ERROR Indicates an error condition if coefficient values are invalid
- NPP_WRONG_INTERSECTION_QUAD_WARNING Indicates a warning that no operation is performed if the transformed source ROI has no intersection with the destination ROI
- NPP_MISALIGNED_DST_ROI_WARNING Indicates a warning that the speed of primitive execution was reduced due to destination ROI

7.4.1.381 NppStatus nppiWarpPerspectiveQuad_32s_C3R (const Npp32s ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp32s ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Perspective transform of an image (32bit signed integer, three channels).

See also:

nppiWarpPerspectiveQuad_32s_C1R

7.4.1.382 NppStatus nppiWarpPerspectiveQuad_32s_C4R (const Npp32s ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp32s ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Perspective transform of an image (32bit signed integer, four channels).

See also:

nppiWarpPerspectiveQuad_32s_C1R

7.4.1.383 NppStatus nppiWarpPerspectiveQuad_32s_P3R (const Npp32s ∗ *pSrc*[3], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp32s ∗ *pDst*[3], int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Perspective transform of an image (32bit signed integer, three planes).

See also:

nppiWarpPerspectiveQuad_32s_C1R

7.4.1.384 NppStatus nppiWarpPerspectiveQuad_32s_P4R (const Npp32s ∗ *pSrc*[4], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp32s ∗ *pDst*[4], int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Perspective transform of an image (32bit signed integer, four planes).

See also:

nppiWarpPerspectiveQuad_32s_C1R

7.4.1.385 NppStatus nppiWarpPerspectiveQuad_8u_AC4R (const Npp8u ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp8u ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Perspective transform of an image (8bit unsigned integer, four channels RGBA).

See also:

nppiWarpPerspectiveQuad_8u_C1R

7.4.1.386 NppStatus nppiWarpPerspectiveQuad_8u_C1R (const Npp8u ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp8u ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Perspective transform of an image (8bit unsigned integer, single channel).

This function performs perspective warping of a the specified quadrangle in the source image to the specified quadrangle in the destination image. The function nppiWarpPerspectiveQuad uses the same formulas for pixel mapping as in nppiWarpPerspective function. The transform coefficients are computed internally.

The transformed part of the source image is resampled using the specified interpolation method and written to the destination ROI.

NPPI specific recommendation: The function operates using 2 types of kernels: fast and accurate. The fast method is about 4 times faster than its accurate variant, but does not perform memory access checks and requires the destination ROI to be 64 bytes aligned. Hence any destination ROI is chunked into 3 vertical stripes: the first and the third are processed by accurate kernels and the central one is processed by the fast one. In order to get the maximum available speed of execution, the projection of destination ROI onto image addresses must be 64 bytes aligned. This is always true if the values `(int)((void *)(pDst + dstRoi.x))` and `(int)((void *)(pDst + dstRoi.x + dstRoi.width))` are multiples of 64. Another rule of thumb is to specify destination ROI in such way that left and right sides of the projected image are separated from the ROI by at least 63 bytes from each side. However, this requires the whole ROI to be part of allocated memory. In case when the conditions above are not satisfied, the function may decrease in speed slightly and will return NPP_MISALIGNED_DST_ROI_WARNING warning.

Parameters:

pSrc Source-Image Pointer.

srcSize Size of source image in pixels

nSrcStep Source-Image Line Step.

srcRoi Source ROI

srcQuad Source quadrangle

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

dstRoi Destination ROI

dstQuad Destination quadrangle

interpolation Interpolation mode: can be NPPI_INTER_NN, NPPI_INTER_LINEAR or NPPI_-INTER_CUBIC

Returns:

Image Data Related Error Codes, ROI Related Error Codes

- NPP_RECT_ERROR Indicates an error condition if width or height of the intersection of the srcRoi and source image is less than or equal to 1
- NPP_WRONG_INTERSECTION_ROI_ERROR Indicates an error condition if srcRoi has no intersection with the source image
- NPP_INTERPOLATION_ERROR Indicates an error condition if interpolation has an illegal value
- NPP_COEFF_ERROR Indicates an error condition if coefficient values are invalid
- NPP_WRONG_INTERSECTION_QUAD_WARNING Indicates a warning that no operation is performed if the transformed source ROI has no intersection with the destination ROI
- NPP_MISALIGNED_DST_ROI_WARNING Indicates a warning that the speed of primitive execution was reduced due to destination ROI

7.4.1.387 NppStatus nppiWarpPerspectiveQuad_8u_C3R (const Npp8u ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp8u ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Perspective transform of an image (8bit unsigned integer, three channels).

See also:

nppiWarpPerspectiveQuad_8u_C1R

7.4.1.388 NppStatus nppiWarpPerspectiveQuad_8u_C4R (const Npp8u ∗ *pSrc*, NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp8u ∗ *pDst*, int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Perspective transform of an image (8bit unsigned integer, four channels).

See also:

nppiWarpPerspectiveQuad_8u_C1R

7.4.1.389 NppStatus nppiWarpPerspectiveQuad_8u_P3R (const Npp8u ∗ *pSrc*[3], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp8u ∗ *pDst*[3], int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Perspective transform of an image (8bit unsigned integer, three planes).

See also:

nppiWarpPerspectiveQuad_8u_C1R

7.4.1.390 NppStatus nppiWarpPerspectiveQuad_8u_P4R (const Npp8u ∗ *pSrc*[4], NppiSize *srcSize*, int *nSrcStep*, NppiRect *srcRoi*, const double *srcQuad*[4][2], Npp8u ∗ *pDst*[4], int *nDstStep*, NppiRect *dstRoi*, const double *dstQuad*[4][2], int *interpolation*)

Perspective transform of an image (8bit unsigned integer, four planes).

See also:

nppiWarpPerspectiveQuad_8u_C1R

7.4.1.391 NppStatus nppiYCbCr420ToRGB_8u_P3C3R (const Npp8u ∗const ∗ *pSrc*, int *nSrcStep*[3], Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

3 channel 8-bit unsigned planar YCbCr420 to packed RGB color conversion.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.392 NppStatus nppiYCbCr420ToYCbCr411_8u_P3P2R (const Npp8u ∗const ∗ *pSrc*, int *aSrcStep*[3], Npp8u ∗ *pDstY*, int *nDstYStep*, Npp8u ∗ *pDstCbCr*, int *nDstCbCrStep*, NppiSize *oSizeROI*)

3 channel 8-bit unsigned planar YCbCr:420 to YCbCr:411 resampling.

Parameters:

pSrc Array of pointers to the source image planes.

aSrcStep Array with distances in bytes between starts of consecutive lines of the source image planes.

pDstY Destination-Image Pointer. Y-channel.

nDstYStep Destination-Image Line Step. Y-channel.

pDstCbCr Destination-Image Pointer. CbCr image.

nDstCbCrStep Destination-Image Line Step. CbCr image.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.393 NppStatus nppiYCbCr420ToYCbCr422_8u_P3R (const Npp8u ∗const ∗ *pSrc*, int *nSrcStep*[3], Npp8u ∗∗ *pDst*, int *nDstStep*[3], NppiSize *oSizeROI*)

3 channel 8-bit unsigned planar YCbCr:420 to YCbCr:422 resampling.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.394 NppStatus nppiYCbCr422ToRGB_8u_C2C3R (const Npp8u ∗ *pSrc*, int *nSrcStep*, Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

2 channel 8-bit unsigned YCbCr422 to 3 channel packed RGB color conversion.

images.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.395 NppStatus nppiYCbCr422ToYCbCr411_8u_P3R (const Npp8u ∗const ∗ *pSrc*, int *nSrcStep*[3], Npp8u ∗∗ *pDst*, int *nDstStep*[3], NppiSize *oSizeROI*)

3 channel 8-bit unsigned planar YCbCr:422 to YCbCr:411 resampling.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.396 NppStatus nppiYCbCr422ToYCbCr420_8u_P3R (const Npp8u ∗const ∗ *pSrc*, int *nSrcStep*[3], Npp8u ∗∗ *pDst*, int *nDstStep*[3], NppiSize *oSizeROI*)

3 channel 8-bit unsigned planar YCbCr:422 to YCbCr:420 resampling.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.397 NppStatus nppiYCbCrToRGB_8u_AC4R (const Npp8u ∗ *pSrc*, int *nSrcStep*, Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

4 channel 8-bit unsigned packed YCbCr to RGB color conversion, not affecting Alpha.

Alpha channel is the last channel and is not processed.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.398 NppStatus nppiYCbCrToRGB_8u_C3R (const Npp8u ∗ *pSrc*, int *nSrcStep*, Npp8u ∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

3 channel 8-bit unsigned packed YCbCr to RGB color conversion.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.4.1.399 NppStatus nppiYCbCrToRGB_8u_P3R (const Npp8u ∗const ∗ *pSrc*, int *nSrcStep*, Npp8u ∗∗ *pDst*, int *nDstStep*, NppiSize *oSizeROI*)

3 channel 8-bit unsigned planar YCbCr to RGB color conversion.

Parameters:

pSrc Source-Image Pointer.

nSrcStep Source-Image Line Step.

pDst Destination-Image Pointer.

nDstStep Destination-Image Line Step.

oSizeROI Region-of-Interest (ROI).

Returns:

Image Data Related Error Codes, ROI Related Error Codes

7.5 NPP Signal Processing

Memory Allocation

Signal-allocator methods for allocating 1D arrays of data in device memory.

All allocators have size parameters to specify the size of the signal (1D array) being allocated.

The allocator methods return a pointer to the newly allocated memory of appropriate type. If device-memory allocation is not possible due to resource constaints the allocators return 0 (i.e. NULL pointer).

All signal allocators allocate memory aligned such that it is beneficial to the performance of the majority of the signal-processing primitives. It is no mandatory however to use these allocators. Any valid CUDA device-memory pointers can be passed to NPP primitives.

- Npp8u ∗ nppsMalloc_8u (int nSize)

  8-bit unsigned signal allocator.

- Npp16u ∗ nppsMalloc_16u (int nSize)

  16-bit unsigned signal allocator.

- Npp16s ∗ nppsMalloc_16s (int nSize)

  16-bit signal allocator.

- Npp16sc ∗ nppsMalloc_16sc (int nSize)

  16-bit complex-value signal allocator.

- Npp32u ∗ nppsMalloc_32u (int nSize)

  32-bit unsigned signal allocator.

- Npp32s ∗ nppsMalloc_32s (int nSize)

  32-bit integer signal allocator.

- Npp32sc ∗ nppsMalloc_32sc (int nSize)

  32-bit complex integer signal allocator.

- Npp32f ∗ nppsMalloc_32f (int nSize)

  32-bit float signal allocator.

- Npp32fc ∗ nppsMalloc_32fc (int nSize)

  32-bit complex float signal allocator.

- Npp64s ∗ nppsMalloc_64s (int nSize)

  64-bit long integer signal allocator.

- Npp64sc ∗ nppsMalloc_64sc (int nSize)

  64-bit complex long integer signal allocator.

- Npp64f ∗ nppsMalloc_64f (int nSize)

  64-bit float (double) signal allocator.

- Npp64fc ∗ nppsMalloc_64fc (int nSize)

64-bit complex complex signal allocator.

- void nppsFree (void ∗pValues)

  Free method for any 2D allocated memory.

Set

Set methods for 1D vectors of various types.

The copy methods operate on vector data given as a pointer to the underlying data-type (e.g. 8-bit vectors would be passed as pointers to Npp8u type) and length of the vectors, i.e. the number of items.

- NppStatus nppsSet_8u (Npp8u nValue, Npp8u ∗pDst, int nLength)

  8-bit unsigned char, vector set method.

- NppStatus nppsSet_16s (Npp16s nValue, Npp16s ∗pDst, int nLength)

  16-bit integer, vector set method.

- NppStatus nppsSet_16sc (Npp16sc nValue, Npp16sc ∗pDst, int nLength)

  16-bit integer complex, vector set method.

- NppStatus nppsSet_32s (Npp32s nValue, Npp32s ∗pDst, int nLength)

  32-bit integer, vector set method.

- NppStatus nppsSet_32sc (Npp32sc nValue, Npp32sc ∗pDst, int nLength)

  32-bit integer complex, vector set method.

- NppStatus nppsSet_32f (Npp32f nValue, Npp32f ∗pDst, int nLength)

  32-bit float, vector set method.

- NppStatus nppsSet_32fc (Npp32fc nValue, Npp32fc ∗pDst, int nLength)

  32-bit float complex, vector set method.

- NppStatus nppsSet_64s (Npp64s nValue, Npp64s ∗pDst, int nLength)

  64-bit long long integer, vector set method.

- NppStatus nppsSet_64sc (Npp64sc nValue, Npp64sc ∗pDst, int nLength)

  64-bit long long integer complex, vector set method.

- NppStatus nppsSet_64f (Npp64f nValue, Npp64f ∗pDst, int nLength)

  64-bit double, vector set method.

- NppStatus nppsSet_64fc (Npp64fc nValue, Npp64fc ∗pDst, int nLength)

  64-bit double complex, vector set method.

Zero

Set signals to zero.

- NppStatus nppsZero_8u (Npp8u ∗pDst, int nLength)
  8-bit unsigned char, vector zero method.
- NppStatus nppsZero_16s (Npp16s ∗pDst, int nLength)
  16-bit integer, vector zero method.
- NppStatus nppsZero_16sc (Npp16sc ∗pDst, int nLength)
  16-bit integer complex, vector zero method.
- NppStatus nppsZero_32s (Npp32s ∗pDst, int nLength)
  32-bit integer, vector zero method.
- NppStatus nppsZero_32sc (Npp32sc ∗pDst, int nLength)
  32-bit integer complex, vector zero method.
- NppStatus nppsZero_32f (Npp32f ∗pDst, int nLength)
  32-bit float, vector zero method.
- NppStatus nppsZero_32fc (Npp32fc ∗pDst, int nLength)
  32-bit float complex, vector zero method.
- NppStatus nppsZero_64s (Npp64s ∗pDst, int nLength)
  64-bit long long integer, vector zero method.
- NppStatus nppsZero_64sc (Npp64sc ∗pDst, int nLength)
  64-bit long long integer complex, vector zero method.
- NppStatus nppsZero_64f (Npp64f ∗pDst, int nLength)
  64-bit double, vector zero method.
- NppStatus nppsZero_64fc (Npp64fc ∗pDst, int nLength)
  64-bit double complex, vector zero method.

Copy

Copy methods for various type signals.

Copy methods operate on signal data given as a pointer to the underlying data-type (e.g. 8-bit vectors would be passed as pointers to Npp8u type) and length of the vectors, i.e. the number of items.

- NppStatus nppsCopy_8u (const Npp8u ∗pSrc, Npp8u ∗pDst, int len)
  8-bit unsigned char, vector copy method
- NppStatus nppsCopy_16s (const Npp16s ∗pSrc, Npp16s ∗pDst, int len)
  16-bit signed short, vector copy method.

- NppStatus nppsCopy_32s (const Npp32s ∗pSrc, Npp32s ∗pDst, int nLength)

  32-bit signed integer, vector copy method.

- NppStatus nppsCopy_32f (const Npp32f ∗pSrc, Npp32f ∗pDst, int len)

  32-bit float, vector copy method.

- NppStatus nppsCopy_64s (const Npp64s ∗pSrc, Npp64s ∗pDst, int len)

  64-bit signed integer, vector copy method.

- NppStatus nppsCopy_16sc (const Npp16sc ∗pSrc, Npp16sc ∗pDst, int len)

  16-bit complex short, vector copy method.

- NppStatus nppsCopy_32sc (const Npp32sc ∗pSrc, Npp32sc ∗pDst, int len)

  32-bit complex signed integer, vector copy method.

- NppStatus nppsCopy_32fc (const Npp32fc ∗pSrc, Npp32fc ∗pDst, int len)

  32-bit complex float, vector copy method.

- NppStatus nppsCopy_64sc (const Npp64sc ∗pSrc, Npp64sc ∗pDst, int len)

  64-bit complex signed integer, vector copy method.

- NppStatus nppsCopy_64fc (const Npp64fc ∗pSrc, Npp64fc ∗pDst, int len)

  64-bit complex double, vector copy method.

AddC

Adds a constant value to each sample of a signal.

- NppStatus nppsAddC_8u_ISfs (Npp8u nValue, Npp8u ∗pSrcDst, int nLength, int nScaleFactor)

  8-bit unsigned char in place signal add constant, scale, then clamp to saturated value

- NppStatus nppsAddC_8u_Sfs (const Npp8u ∗pSrc, Npp8u nValue, Npp8u ∗pDst, int nLength, int nScaleFactor)

  8-bit unsigned charvector add constant, scale, then clamp to saturated value.

- NppStatus nppsAddC_16u_ISfs (Npp16u nValue, Npp16u ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit unsigned short in place signal add constant, scale, then clamp to saturated value.

- NppStatus nppsAddC_16u_Sfs (const Npp16u ∗pSrc, Npp16u nValue, Npp16u ∗pDst, int nLength, int nScaleFactor)

  16-bit unsigned short vector add constant, scale, then clamp to saturated value.

- NppStatus nppsAddC_16s_ISfs (Npp16s nValue, Npp16s ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit signed short in place signal add constant, scale, then clamp to saturated value.

- NppStatus nppsAddC_16s_Sfs (const Npp16s ∗pSrc, Npp16s nValue, Npp16s ∗pDst, int nLength, int nScaleFactor)

16-bit signed short signal add constant, scale, then clamp to saturated value.

- NppStatus nppsAddC_16sc_ISfs (Npp16sc nValue, Npp16sc ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit integer complex number (16 bit real, 16 bit imaginary) signal add constant, scale, then clamp to saturated value.

- NppStatus nppsAddC_16sc_Sfs (const Npp16sc ∗pSrc, Npp16sc nValue, Npp16sc ∗pDst, int nLength, int nScaleFactor)

  16-bit integer complex number (16 bit real, 16 bit imaginary) signal add constant, scale, then clamp to saturated value.

- NppStatus nppsAddC_32s_ISfs (Npp32s nValue, Npp32s ∗pSrcDst, int nLength, int nScaleFactor)

  32-bit signed integer in place signal add constant and scale.

- NppStatus nppsAddC_32s_Sfs (const Npp32s ∗pSrc, Npp32s nValue, Npp32s ∗pDst, int nLength, int nScaleFactor)

  32-bit signed integersignal add constant and scale.

- NppStatus nppsAddC_32sc_ISfs (Npp32sc nValue, Npp32sc ∗pSrcDst, int nLength, int nScaleFactor)

  32-bit integer complex number (32 bit real, 32 bit imaginary) in place signal add constant and scale.

- NppStatus nppsAddC_32sc_Sfs (const Npp32sc ∗pSrc, Npp32sc nValue, Npp32sc ∗pDst, int nLength, int nScaleFactor)

  32-bit integer complex number (32 bit real, 32 bit imaginary) signal add constant and scale.

- NppStatus nppsAddC_32f_I (Npp32f nValue, Npp32f ∗pSrcDst, int nLength)

  32-bit floating point in place signal add constant.

- NppStatus nppsAddC_32f (const Npp32f ∗pSrc, Npp32f nValue, Npp32f ∗pDst, int nLength)

  32-bit floating point signal add constant.

- NppStatus nppsAddC_32fc_I (Npp32fc nValue, Npp32fc ∗pSrcDst, int nLength)

  32-bit floating point complex number (32 bit real, 32 bit imaginary) in place signal add constant.

- NppStatus nppsAddC_32fc (const Npp32fc ∗pSrc, Npp32fc nValue, Npp32fc ∗pDst, int nLength)

  32-bit floating point complex number (32 bit real, 32 bit imaginary) signal add constant.

- NppStatus nppsAddC_64f_I (Npp64f nValue, Npp64f ∗pSrcDst, int nLength)

  64-bit floating point, in place signal add constant.

- NppStatus nppsAddC_64f (const Npp64f ∗pSrc, Npp64f nValue, Npp64f ∗pDst, int nLength)

  64-bit floating pointsignal add constant.

- NppStatus nppsAddC_64fc_I (Npp64fc nValue, Npp64fc ∗pSrcDst, int nLength)

  64-bit floating point complex number (64 bit real, 64 bit imaginary) in place signal add constant.

- NppStatus nppsAddC_64fc (const Npp64fc ∗pSrc, Npp64fc nValue, Npp64fc ∗pDst, int nLength)

  64-bit floating point complex number (64 bit real, 64 bit imaginary) signal add constant.

AddProductC

Adds product of a constant and each sample of a source signal to each sample of the destination signal.

- NppStatus nppsAddProductC_8u_ISfs (Npp8u nValue, Npp8u ∗pSrcDst, int nLength, int nScaleFactor)

  8-bit unsigned char in place signal add product of signal times constant to destination signal, scale, then clamp to saturated value

- NppStatus nppsAddProductC_8u_Sfs (const Npp8u ∗pSrc, Npp8u nValue, Npp8u ∗pDst, int nLength, int nScaleFactor)

  8-bit unsigned char add product of signal times constant to destination signal, scale, then clamp to saturated value.

- NppStatus nppsAddProductC_16u_ISfs (Npp16u nValue, Npp16u ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit unsigned short in place signal add product of signal times constant to destination signal, scale, then clamp to saturated value.

- NppStatus nppsAddProductC_16u_Sfs (const Npp16u ∗pSrc, Npp16u nValue, Npp16u ∗pDst, int nLength, int nScaleFactor)

  16-bit unsigned short add product of signal times constant to destination signal, scale, then clamp to saturated value.

- NppStatus nppsAddProductC_16s_ISfs (Npp16s nValue, Npp16s ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit signed short in place signal add product of signal times constant to destination signal, scale, then clamp to saturated value.

- NppStatus nppsAddProductC_16s_Sfs (const Npp16s ∗pSrc, Npp16s nValue, Npp16s ∗pDst, int nLength, int nScaleFactor)

  16-bit signed short signal add product of signal times constant to destination signal, scale, then clamp to saturated value.

- NppStatus nppsAddProductC_16sc_ISfs (Npp16sc nValue, Npp16sc ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit integer complex number (16 bit real, 16 bit imaginary)signal add product of signal times constant to destination signal, scale, then clamp to saturated value.

- NppStatus nppsAddProductC_16sc_Sfs (const Npp16sc ∗pSrc, Npp16sc nValue, Npp16sc ∗pDst, int nLength, int nScaleFactor)

  16-bit integer complex number (16 bit real, 16 bit imaginary)signal add product of signal times constant to destination signal, scale, then clamp to saturated value.

- NppStatus nppsAddProductC_32s_ISfs (Npp32s nValue, Npp32s ∗pSrcDst, int nLength, int nScaleFactor)

  32-bit signed integer in place signal add product of signal times constant to destination signal and scale.

- NppStatus nppsAddProductC_32s_Sfs (const Npp32s ∗pSrc, Npp32s nValue, Npp32s ∗pDst, int nLength, int nScaleFactor)

  32-bit signed integer signal add product of signal times constant to destination signal and scale.

- NppStatus nppsAddProductC_32sc_ISfs (Npp32sc nValue, Npp32sc ∗pSrcDst, int nLength, int nScaleFactor)

  32-bit integer complex number (32 bit real, 32 bit imaginary) in place signal add product of signal times constant to destination signal and scale.

- NppStatus nppsAddProductC_32sc_Sfs (const Npp32sc ∗pSrc, Npp32sc nValue, Npp32sc ∗pDst, int nLength, int nScaleFactor)

  32-bit integer complex number (32 bit real, 32 bit imaginary)signal add product of signal times constant to destination signal and scale.

- NppStatus nppsAddProductC_32f_I (Npp32f nValue, Npp32f ∗pSrcDst, int nLength)

  32-bit floating point in place signal add product of signal times constant to destination signal.

- NppStatus nppsAddProductC_32f (const Npp32f ∗pSrc, Npp32f nValue, Npp32f ∗pDst, int nLength)

  32-bit floating point signal add product of signal times constant to destination signal.

MulC

Multiplies each sample of a signal by a constant value.

- NppStatus nppsMulC_8u_ISfs (Npp8u nValue, Npp8u ∗pSrcDst, int nLength, int nScaleFactor)

  8-bit unsigned char in place signal times constant, scale, then clamp to saturated value

- NppStatus nppsMulC_8u_Sfs (const Npp8u ∗pSrc, Npp8u nValue, Npp8u ∗pDst, int nLength, int nScaleFactor)

  8-bit unsigned char signal times constant, scale, then clamp to saturated value.

- NppStatus nppsMulC_16u_ISfs (Npp16u nValue, Npp16u ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit unsigned short in place signal times constant, scale, then clamp to saturated value.

- NppStatus nppsMulC_16u_Sfs (const Npp16u ∗pSrc, Npp16u nValue, Npp16u ∗pDst, int nLength, int nScaleFactor)

  16-bit unsigned short signal times constant, scale, then clamp to saturated value.

- NppStatus nppsMulC_16s_ISfs (Npp16s nValue, Npp16s ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit signed short in place signal times constant, scale, then clamp to saturated value.

- NppStatus nppsMulC_16s_Sfs (const Npp16s ∗pSrc, Npp16s nValue, Npp16s ∗pDst, int nLength, int nScaleFactor)

  16-bit signed short signal times constant, scale, then clamp to saturated value.

- NppStatus nppsMulC_16sc_ISfs (Npp16sc nValue, Npp16sc ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit integer complex number (16 bit real, 16 bit imaginary)signal times constant, scale, then clamp to saturated value.

- NppStatus nppsMulC_16sc_Sfs (const Npp16sc ∗pSrc, Npp16sc nValue, Npp16sc ∗pDst, int nLength, int nScaleFactor)

16-bit integer complex number (16 bit real, 16 bit imaginary)signal times constant, scale, then clamp to saturated value.

- NppStatus nppsMulC_32s_ISfs (Npp32s nValue, Npp32s ∗pSrcDst, int nLength, int nScaleFactor)

  32-bit signed integer in place signal times constant and scale.

- NppStatus nppsMulC_32s_Sfs (const Npp32s ∗pSrc, Npp32s nValue, Npp32s ∗pDst, int nLength, int nScaleFactor)

  32-bit signed integer signal times constant and scale.

- NppStatus nppsMulC_32sc_ISfs (Npp32sc nValue, Npp32sc ∗pSrcDst, int nLength, int nScaleFactor)

  32-bit integer complex number (32 bit real, 32 bit imaginary) in place signal times constant and scale.

- NppStatus nppsMulC_32sc_Sfs (const Npp32sc ∗pSrc, Npp32sc nValue, Npp32sc ∗pDst, int nLength, int nScaleFactor)

  32-bit integer complex number (32 bit real, 32 bit imaginary) signal times constant and scale.

- NppStatus nppsMulC_32f_I (Npp32f nValue, Npp32f ∗pSrcDst, int nLength)

  32-bit floating point in place signal times constant.

- NppStatus nppsMulC_32f (const Npp32f ∗pSrc, Npp32f nValue, Npp32f ∗pDst, int nLength)

  32-bit floating point signal times constant.

- NppStatus nppsMulC_Low_32f16s (const Npp32f ∗pSrc, Npp32f nValue, Npp16s ∗pDst, int nLength)

  32-bit floating point signal times constant with output converted to 16-bit signed integer.

- NppStatus nppsMulC_32f16s_Sfs (const Npp32f ∗pSrc, Npp32f nValue, Npp16s ∗pDst, int nLength, int nScaleFactor)

  32-bit floating point signal times constant with output converted to 16-bit signed integer with scaling and saturation of output result.

- NppStatus nppsMulC_32fc_I (Npp32fc nValue, Npp32fc ∗pSrcDst, int nLength)

  32-bit floating point complex number (32 bit real, 32 bit imaginary) in place signal times constant.

- NppStatus nppsMulC_32fc (const Npp32fc ∗pSrc, Npp32fc nValue, Npp32fc ∗pDst, int nLength)

  32-bit floating point complex number (32 bit real, 32 bit imaginary) signal times constant.

- NppStatus nppsMulC_64f_I (Npp64f nValue, Npp64f ∗pSrcDst, int nLength)

  64-bit floating point, in place signal times constant.

- NppStatus nppsMulC_64f (const Npp64f ∗pSrc, Npp64f nValue, Npp64f ∗pDst, int nLength)

  64-bit floating point signal times constant.

- NppStatus nppsMulC_64f64s_ISfs (Npp64f nValue, Npp64s ∗pDst, int nLength, int nScaleFactor)

  64-bit floating point signal times constant with in place conversion to 64-bit signed integer and with scaling and saturation of output result.

- NppStatus nppsMulC_64fc_I (Npp64fc nValue, Npp64fc ∗pSrcDst, int nLength)

  64-bit floating point complex number (64 bit real, 64 bit imaginary) in place signal times constant.

- NppStatus nppsMulC_64fc (const Npp64fc ∗pSrc, Npp64fc nValue, Npp64fc ∗pDst, int nLength)

  64-bit floating point complex number (64 bit real, 64 bit imaginary) signal times constant.

SubC

Subtracts a constant from each sample of a signal.

- NppStatus nppsSubC_8u_ISfs (Npp8u nValue, Npp8u ∗pSrcDst, int nLength, int nScaleFactor)

  8-bit unsigned char in place signal subtract constant, scale, then clamp to saturated value

- NppStatus nppsSubC_8u_Sfs (const Npp8u ∗pSrc, Npp8u nValue, Npp8u ∗pDst, int nLength, int nScaleFactor)

  8-bit unsigned char signal subtract constant, scale, then clamp to saturated value.

- NppStatus nppsSubC_16u_ISfs (Npp16u nValue, Npp16u ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit unsigned short in place signal subtract constant, scale, then clamp to saturated value.

- NppStatus nppsSubC_16u_Sfs (const Npp16u ∗pSrc, Npp16u nValue, Npp16u ∗pDst, int nLength, int nScaleFactor)

  16-bit unsigned short signal subtract constant, scale, then clamp to saturated value.

- NppStatus nppsSubC_16s_ISfs (Npp16s nValue, Npp16s ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit signed short in place signal subtract constant, scale, then clamp to saturated value.

- NppStatus nppsSubC_16s_Sfs (const Npp16s ∗pSrc, Npp16s nValue, Npp16s ∗pDst, int nLength, int nScaleFactor)

  16-bit signed short signal subtract constant, scale, then clamp to saturated value.

- NppStatus nppsSubC_16sc_ISfs (Npp16sc nValue, Npp16sc ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit integer complex number (16 bit real, 16 bit imaginary) signal subtract constant, scale, then clamp to saturated value.

- NppStatus nppsSubC_16sc_Sfs (const Npp16sc ∗pSrc, Npp16sc nValue, Npp16sc ∗pDst, int nLength, int nScaleFactor)

  16-bit integer complex number (16 bit real, 16 bit imaginary) signal subtract constant, scale, then clamp to saturated value.

- NppStatus nppsSubC_32s_ISfs (Npp32s nValue, Npp32s ∗pSrcDst, int nLength, int nScaleFactor)

  32-bit signed integer in place signal subtract constant and scale.

- NppStatus nppsSubC_32s_Sfs (const Npp32s ∗pSrc, Npp32s nValue, Npp32s ∗pDst, int nLength, int nScaleFactor)

  32-bit signed integer signal subtract constant and scale.

- NppStatus nppsSubC_32sc_ISfs (Npp32sc nValue, Npp32sc ∗pSrcDst, int nLength, int nScaleFactor)

  32-bit integer complex number (32 bit real, 32 bit imaginary) in place signal subtract constant and scale.

- NppStatus nppsSubC_32sc_Sfs (const Npp32sc ∗pSrc, Npp32sc nValue, Npp32sc ∗pDst, int nLength, int nScaleFactor)

  32-bit integer complex number (32 bit real, 32 bit imaginary) signal subtract constant and scale.

- NppStatus nppsSubC_32f_I (Npp32f nValue, Npp32f ∗pSrcDst, int nLength)

  32-bit floating point in place signal subtract constant.

- NppStatus nppsSubC_32f (const Npp32f ∗pSrc, Npp32f nValue, Npp32f ∗pDst, int nLength)

  32-bit floating point signal subtract constant.

- NppStatus nppsSubC_32fc_I (Npp32fc nValue, Npp32fc ∗pSrcDst, int nLength)

  32-bit floating point complex number (32 bit real, 32 bit imaginary) in place signal subtract constant.

- NppStatus nppsSubC_32fc (const Npp32fc ∗pSrc, Npp32fc nValue, Npp32fc ∗pDst, int nLength)

  32-bit floating point complex number (32 bit real, 32 bit imaginary) signal subtract constant.

- NppStatus nppsSubC_64f_I (Npp64f nValue, Npp64f ∗pSrcDst, int nLength)

  64-bit floating point, in place signal subtract constant.

- NppStatus nppsSubC_64f (const Npp64f ∗pSrc, Npp64f nValue, Npp64f ∗pDst, int nLength)

  64-bit floating point signal subtract constant.

- NppStatus nppsSubC_64fc_I (Npp64fc nValue, Npp64fc ∗pSrcDst, int nLength)

  64-bit floating point complex number (64 bit real, 64 bit imaginary) in place signal subtract constant.

- NppStatus nppsSubC_64fc (const Npp64fc ∗pSrc, Npp64fc nValue, Npp64fc ∗pDst, int nLength)

  64-bit floating point complex number (64 bit real, 64 bit imaginary) signal subtract constant.

SubCRev

Subtracts each sample of a signal from a constant.

- NppStatus nppsSubCRev_8u_ISfs (Npp8u nValue, Npp8u ∗pSrcDst, int nLength, int nScaleFactor)

  8-bit unsigned char in place signal subtract from constant, scale, then clamp to saturated value

- NppStatus nppsSubCRev_8u_Sfs (const Npp8u ∗pSrc, Npp8u nValue, Npp8u ∗pDst, int nLength, int nScaleFactor)

  8-bit unsigned char signal subtract from constant, scale, then clamp to saturated value.

- NppStatus nppsSubCRev_16u_ISfs (Npp16u nValue, Npp16u ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit unsigned short in place signal subtract from constant, scale, then clamp to saturated value.

- NppStatus nppsSubCRev_16u_Sfs (const Npp16u ∗pSrc, Npp16u nValue, Npp16u ∗pDst, int nLength, int nScaleFactor)

  16-bit unsigned short signal subtract from constant, scale, then clamp to saturated value.

- NppStatus nppsSubCRev_16s_ISfs (Npp16s nValue, Npp16s ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit signed short in place signal subtract from constant, scale, then clamp to saturated value.

- NppStatus nppsSubCRev_16s_Sfs (const Npp16s ∗pSrc, Npp16s nValue, Npp16s ∗pDst, int nLength, int nScaleFactor)

  16-bit signed short signal subtract from constant, scale, then clamp to saturated value.

- NppStatus nppsSubCRev_16sc_ISfs (Npp16sc nValue, Npp16sc ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit integer complex number (16 bit real, 16 bit imaginary) signal subtract from constant, scale, then clamp to saturated value.

- NppStatus nppsSubCRev_16sc_Sfs (const Npp16sc ∗pSrc, Npp16sc nValue, Npp16sc ∗pDst, int nLength, int nScaleFactor)

  16-bit integer complex number (16 bit real, 16 bit imaginary) signal subtract from constant, scale, then clamp to saturated value.

- NppStatus nppsSubCRev_32s_ISfs (Npp32s nValue, Npp32s ∗pSrcDst, int nLength, int nScaleFactor)

  32-bit signed integer in place signal subtract from constant and scale.

- NppStatus nppsSubCRev_32s_Sfs (const Npp32s ∗pSrc, Npp32s nValue, Npp32s ∗pDst, int nLength, int nScaleFactor)

  32-bit signed integersignal subtract from constant and scale.

- NppStatus nppsSubCRev_32sc_ISfs (Npp32sc nValue, Npp32sc ∗pSrcDst, int nLength, int nScaleFactor)

  32-bit integer complex number (32 bit real, 32 bit imaginary) in place signal subtract from constant and scale.

- NppStatus nppsSubCRev_32sc_Sfs (const Npp32sc ∗pSrc, Npp32sc nValue, Npp32sc ∗pDst, int nLength, int nScaleFactor)

  32-bit integer complex number (32 bit real, 32 bit imaginary) signal subtract from constant and scale.

- NppStatus nppsSubCRev_32f_I (Npp32f nValue, Npp32f ∗pSrcDst, int nLength)

  32-bit floating point in place signal subtract from constant.

- NppStatus nppsSubCRev_32f (const Npp32f ∗pSrc, Npp32f nValue, Npp32f ∗pDst, int nLength)

  32-bit floating point signal subtract from constant.

- NppStatus nppsSubCRev_32fc_I (Npp32fc nValue, Npp32fc ∗pSrcDst, int nLength)

  32-bit floating point complex number (32 bit real, 32 bit imaginary) in place signal subtract from constant.

- NppStatus nppsSubCRev_32fc (const Npp32fc ∗pSrc, Npp32fc nValue, Npp32fc ∗pDst, int nLength)

  32-bit floating point complex number (32 bit real, 32 bit imaginary) signal subtract from constant.

- NppStatus nppsSubCRev_64f_I (Npp64f nValue, Npp64f ∗pSrcDst, int nLength)

  64-bit floating point, in place signal subtract from constant.

- NppStatus nppsSubCRev_64f (const Npp64f ∗pSrc, Npp64f nValue, Npp64f ∗pDst, int nLength)

  64-bit floating point signal subtract from constant.

- NppStatus nppsSubCRev_64fc_I (Npp64fc nValue, Npp64fc ∗pSrcDst, int nLength)

  64-bit floating point complex number (64 bit real, 64 bit imaginary) in place signal subtract from constant.

- NppStatus nppsSubCRev_64fc (const Npp64fc ∗pSrc, Npp64fc nValue, Npp64fc ∗pDst, int nLength)

  64-bit floating point complex number (64 bit real, 64 bit imaginary) signal subtract from constant.

DivC

Divides each sample of a signal by a constant.

- NppStatus nppsDivC_8u_ISfs (Npp8u nValue, Npp8u ∗pSrcDst, int nLength, int nScaleFactor)

  8-bit unsigned char in place signal divided by constant, scale, then clamp to saturated value

- NppStatus nppsDivC_8u_Sfs (const Npp8u ∗pSrc, Npp8u nValue, Npp8u ∗pDst, int nLength, int nScaleFactor)

  8-bit unsigned char signal divided by constant, scale, then clamp to saturated value.

- NppStatus nppsDivC_16u_ISfs (Npp16u nValue, Npp16u ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit unsigned short in place signal divided by constant, scale, then clamp to saturated value.

- NppStatus nppsDivC_16u_Sfs (const Npp16u ∗pSrc, Npp16u nValue, Npp16u ∗pDst, int nLength, int nScaleFactor)

  16-bit unsigned short signal divided by constant, scale, then clamp to saturated value.

- NppStatus nppsDivC_16s_ISfs (Npp16s nValue, Npp16s ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit signed short in place signal divided by constant, scale, then clamp to saturated value.

- NppStatus nppsDivC_16s_Sfs (const Npp16s ∗pSrc, Npp16s nValue, Npp16s ∗pDst, int nLength, int nScaleFactor)

  16-bit signed short signal divided by constant, scale, then clamp to saturated value.

- NppStatus nppsDivC_16sc_ISfs (Npp16sc nValue, Npp16sc ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit integer complex number (16 bit real, 16 bit imaginary) signal divided by constant, scale, then clamp to saturated value.

- NppStatus nppsDivC_16sc_Sfs (const Npp16sc ∗pSrc, Npp16sc nValue, Npp16sc ∗pDst, int nLength, int nScaleFactor)

  16-bit integer complex number (16 bit real, 16 bit imaginary) signal divided by constant, scale, then clamp to saturated value.

- NppStatus nppsDivC_32s_ISfs (Npp32s nValue, Npp32s ∗pSrcDst, int nLength, int nScaleFactor)

  32-bit signed integer in place signal divided by constant and scale.

- NppStatus nppsDivC_32s_Sfs (const Npp32s ∗pSrc, Npp32s nValue, Npp32s ∗pDst, int nLength, int nScaleFactor)

  32-bit signed integer signal divided by constant and scale.

- NppStatus nppsDivC_32sc_ISfs (Npp32sc nValue, Npp32sc ∗pSrcDst, int nLength, int nScaleFactor)

  32-bit integer complex number (32 bit real, 32 bit imaginary) in place signal divided by constant and scale.

- NppStatus nppsDivC_32sc_Sfs (const Npp32sc ∗pSrc, Npp32sc nValue, Npp32sc ∗pDst, int nLength, int nScaleFactor)

  32-bit integer complex number (32 bit real, 32 bit imaginary) signal divided by constant and scale.

- NppStatus nppsDivC_32f_I (Npp32f nValue, Npp32f ∗pSrcDst, int nLength)

  32-bit floating point in place signal divided by constant.

- NppStatus nppsDivC_32f (const Npp32f ∗pSrc, Npp32f nValue, Npp32f ∗pDst, int nLength)

  32-bit floating point signal divided by constant.

- NppStatus nppsDivC_32fc_I (Npp32fc nValue, Npp32fc ∗pSrcDst, int nLength)

  32-bit floating point complex number (32 bit real, 32 bit imaginary) in place signal divided by constant.

- NppStatus nppsDivC_32fc (const Npp32fc ∗pSrc, Npp32fc nValue, Npp32fc ∗pDst, int nLength)

  32-bit floating point complex number (32 bit real, 32 bit imaginary) signal divided by constant.

- NppStatus nppsDivC_64f_I (Npp64f nValue, Npp64f ∗pSrcDst, int nLength)

  64-bit floating point in place signal divided by constant.

- NppStatus nppsDivC_64f (const Npp64f ∗pSrc, Npp64f nValue, Npp64f ∗pDst, int nLength)

  64-bit floating point signal divided by constant.

- NppStatus nppsDivC_64fc_I (Npp64fc nValue, Npp64fc ∗pSrcDst, int nLength)

  64-bit floating point complex number (64 bit real, 64 bit imaginary) in place signal divided by constant.

- NppStatus nppsDivC_64fc (const Npp64fc ∗pSrc, Npp64fc nValue, Npp64fc ∗pDst, int nLength)

  64-bit floating point complex number (64 bit real, 64 bit imaginary) signal divided by constant.

DivCRev

Divides a constant by each sample of a signal.

- NppStatus nppsDivCRev_8u_I (Npp8u nValue, Npp8u ∗pSrcDst, int nLength)

  8-bit unsigned char signal in place constant divided by signal, scale, then clamp to saturated value

- NppStatus nppsDivCRev_8u (const Npp8u ∗pSrc, Npp8u nValue, Npp8u ∗pDst, int nLength)

  8-bit unsigned char signal divided by constant, then clamp to saturated value.

- NppStatus nppsDivCRev_16u_I (Npp16u nValue, Npp16u ∗pSrcDst, int nLength)

  16-bit unsigned short in place constant divided by signal, then clamp to saturated value.

- NppStatus nppsDivCRev_16u (const Npp16u ∗pSrc, Npp16u nValue, Npp16u ∗pDst, int nLength)
  *16-bit unsigned short vector divided by constant, then clamp to saturated value.
- NppStatus nppsDivCRev_16s_I (Npp16s nValue, Npp16s ∗pSrcDst, int nLength)
  16-bit signed short in place constant divided by signal, then clamp to saturated value.
- NppStatus nppsDivCRev_16s (const Npp16s ∗pSrc, Npp16s nValue, Npp16s ∗pDst, int nLength)
  16-bit signed short constant divided by signal, then clamp to saturated value.
- NppStatus nppsDivCRev_32s_I (Npp32s nValue, Npp32s ∗pSrcDst, int nLength)
  32-bit signed integer in place constant divided by signal.
- NppStatus nppsDivCRev_32s (const Npp32s ∗pSrc, Npp32s nValue, Npp32s ∗pDst, int nLength)
  32-bit signed integer constant divided by signal.
- NppStatus nppsDivCRev_32f_I (Npp32f nValue, Npp32f ∗pSrcDst, int nLength)
  32-bit floating point in place constant divided by signal.
- NppStatus nppsDivCRev_32f (const Npp32f ∗pSrc, Npp32f nValue, Npp32f ∗pDst, int nLength)
  32-bit floating point constant divided by signal.
- NppStatus nppsDivCRev_64f_I (Npp64f nValue, Npp64f ∗pSrcDst, int nLength)
  64-bit floating point in place constant divided by signal.
- NppStatus nppsDivCRev_64f (const Npp64f ∗pSrc, Npp64f nValue, Npp64f ∗pDst, int nLength)
  64-bit floating point constant divided by signal.

Add Signal

Sample by sample addition of two signals.

- NppStatus nppsAdd_16s (const Npp16s ∗pSrc1, const Npp16s ∗pSrc2, Npp16s ∗pDst, int nLength)
  16-bit signed short signal add signal, then clamp to saturated value.
- NppStatus nppsAdd_16u (const Npp16u ∗pSrc1, const Npp16u ∗pSrc2, Npp16u ∗pDst, int nLength)
  16-bit unsigned short signal add signal, then clamp to saturated value.
- NppStatus nppsAdd_32u (const Npp32u ∗pSrc1, const Npp32u ∗pSrc2, Npp32u ∗pDst, int nLength)
  32-bit unsigned int signal add signal, then clamp to saturated value.
- NppStatus nppsAdd_32f (const Npp32f ∗pSrc1, const Npp32f ∗pSrc2, Npp32f ∗pDst, int nLength)
  32-bit floating point signal add signal, then clamp to saturated value.
- NppStatus nppsAdd_64f (const Npp64f ∗pSrc1, const Npp64f ∗pSrc2, Npp64f ∗pDst, int nLength)
  64-bit floating point signal add signal, then clamp to saturated value.

- NppStatus nppsAdd_32fc (const Npp32fc ∗pSrc1, const Npp32fc ∗pSrc2, Npp32fc ∗pDst, int nLength)

  32-bit complex floating point signal add signal, then clamp to saturated value.

- NppStatus nppsAdd_64fc (const Npp64fc ∗pSrc1, const Npp64fc ∗pSrc2, Npp64fc ∗pDst, int nLength)

  64-bit complex floating point signal add signal, then clamp to saturated value.

- NppStatus nppsAdd_8u16u (const Npp8u ∗pSrc1, const Npp8u ∗pSrc2, Npp16u ∗pDst, int nLength)

  8-bit unsigned char signal add signal with 16-bit unsigned result, then clamp to saturated value.

- NppStatus nppsAdd_16s32f (const Npp16s ∗pSrc1, const Npp16s ∗pSrc2, Npp32f ∗pDst, int nLength)

  16-bit signed short signal add signal with 32-bit floating point result, then clamp to saturated value.

- NppStatus nppsAdd_8u_Sfs (const Npp8u ∗pSrc1, const Npp8u ∗pSrc2, Npp8u ∗pDst, int nLength, int nScaleFactor)

  8-bit unsigned char add signal, scale, then clamp to saturated value.

- NppStatus nppsAdd_16u_Sfs (const Npp16u ∗pSrc1, const Npp16u ∗pSrc2, Npp16u ∗pDst, int nLength, int nScaleFactor)

  16-bit unsigned short add signal, scale, then clamp to saturated value.

- NppStatus nppsAdd_16s_Sfs (const Npp16s ∗pSrc1, const Npp16s ∗pSrc2, Npp16s ∗pDst, int nLength, int nScaleFactor)

  16-bit signed short add signal, scale, then clamp to saturated value.

- NppStatus nppsAdd_32s_Sfs (const Npp32s ∗pSrc1, const Npp32s ∗pSrc2, Npp32s ∗pDst, int nLength, int nScaleFactor)

  32-bit signed integer add signal, scale, then clamp to saturated value.

- NppStatus nppsAdd_64s_Sfs (const Npp64s ∗pSrc1, const Npp64s ∗pSrc2, Npp64s ∗pDst, int nLength, int nScaleFactor)

  64-bit signed integer add signal, scale, then clamp to saturated value.

- NppStatus nppsAdd_16sc_Sfs (const Npp16sc ∗pSrc1, const Npp16sc ∗pSrc2, Npp16sc ∗pDst, int nLength, int nScaleFactor)

  16-bit signed complex short add signal, scale, then clamp to saturated value.

- NppStatus nppsAdd_32sc_Sfs (const Npp32sc ∗pSrc1, const Npp32sc ∗pSrc2, Npp32sc ∗pDst, int nLength, int nScaleFactor)

  32-bit signed complex integer add signal, scale, then clamp to saturated value.

- NppStatus nppsAdd_16s_I (const Npp16s ∗pSrc, Npp16s ∗pSrcDst, int nLength)

  16-bit signed short in place signal add signal, then clamp to saturated value.

- NppStatus nppsAdd_32f_I (const Npp32f ∗pSrc, Npp32f ∗pSrcDst, int nLength)

  32-bit floating point in place signal add signal, then clamp to saturated value.

- NppStatus nppsAdd_64f_I (const Npp64f ∗pSrc, Npp64f ∗pSrcDst, int nLength)

64-bit floating point in place signal add signal, then clamp to saturated value.

- NppStatus nppsAdd_32fc_I (const Npp32fc ∗pSrc, Npp32fc ∗pSrcDst, int nLength)

  32-bit complex floating point in place signal add signal, then clamp to saturated value.

- NppStatus nppsAdd_64fc_I (const Npp64fc ∗pSrc, Npp64fc ∗pSrcDst, int nLength)

  64-bit complex floating point in place signal add signal, then clamp to saturated value.

- NppStatus nppsAdd_16s32s_I (const Npp16s ∗pSrc, Npp32s ∗pSrcDst, int nLength)

  16/32-bit signed short in place signal add signal with 32-bit signed integer results, then clamp to saturated value.

- NppStatus nppsAdd_8u_ISfs (const Npp8u ∗pSrc, Npp8u ∗pSrcDst, int nLength, int nScaleFactor)

  8-bit unsigned char in place signal add signal, with scaling, then clamp to saturated value.

- NppStatus nppsAdd_16u_ISfs (const Npp16u ∗pSrc, Npp16u ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit unsigned short in place signal add signal, with scaling, then clamp to saturated value.

- NppStatus nppsAdd_16s_ISfs (const Npp16s ∗pSrc, Npp16s ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit signed short in place signal add signal, with scaling, then clamp to saturated value.

- NppStatus nppsAdd_32s_ISfs (const Npp32s ∗pSrc, Npp32s ∗pSrcDst, int nLength, int nScaleFactor)

  32-bit signed integer in place signal add signal, with scaling, then clamp to saturated value.

- NppStatus nppsAdd_16sc_ISfs (const Npp16sc ∗pSrc, Npp16sc ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit complex signed short in place signal add signal, with scaling, then clamp to saturated value.

- NppStatus nppsAdd_32sc_ISfs (const Npp32sc ∗pSrc, Npp32sc ∗pSrcDst, int nLength, int nScaleFactor)

  32-bit complex signed integer in place signal add signal, with scaling, then clamp to saturated value.

AddProduct Signal

Adds sample by sample product of two signals to the destination signal.

- NppStatus nppsAddProduct_32f (const Npp32f ∗pSrc1, const Npp32f ∗pSrc2, Npp32f ∗pDst, int nLength)

  32-bit floating point signal add product of source signal times destination signal to destination signal, then clamp to saturated value.

- NppStatus nppsAddProduct_64f (const Npp64f ∗pSrc1, const Npp64f ∗pSrc2, Npp64f ∗pDst, int nLength)

  64-bit floating point signal add product of source signal times destination signal to destination signal, then clamp to saturated value.

- NppStatus nppsAddProduct_32fc (const Npp32fc ∗pSrc1, const Npp32fc ∗pSrc2, Npp32fc ∗pDst, int nLength)

  32-bit complex floating point signal add product of source signal times destination signal to destination signal, then clamp to saturated value.

- NppStatus nppsAddProduct_64fc (const Npp64fc ∗pSrc1, const Npp64fc ∗pSrc2, Npp64fc ∗pDst, int nLength)

  64-bit complex floating point signal add product of source signal times destination signal to destination signal, then clamp to saturated value.

- NppStatus nppsAddProduct_16s_Sfs (const Npp16s ∗pSrc1, const Npp16s ∗pSrc2, Npp16s ∗pDst, int nLength, int nScaleFactor)

  16-bit signed short signal add product of source signal1 times source signal2 to destination signal, with scaling, then clamp to saturated value.

- NppStatus nppsAddProduct_32s_Sfs (const Npp32s ∗pSrc1, const Npp32s ∗pSrc2, Npp32s ∗pDst, int nLength, int nScaleFactor)

  32-bit signed short signal add product of source signal1 times source signal2 to destination signal, with scaling, then clamp to saturated value.

- NppStatus nppsAddProduct_16s32s_Sfs (const Npp16s ∗pSrc1, const Npp16s ∗pSrc2, Npp32s ∗pDst, int nLength, int nScaleFactor)

  16-bit signed short signal add product of source signal1 times source signal2 to 32-bit signed integer destination signal, with scaling, then clamp to saturated value.

Mul Signal

Sample by sample multiplication the samples of two signals.

- NppStatus nppsMul_16s (const Npp16s ∗pSrc1, const Npp16s ∗pSrc2, Npp16s ∗pDst, int nLength)

  16-bit signed short signal times signal, then clamp to saturated value.

- NppStatus nppsMul_32f (const Npp32f ∗pSrc1, const Npp32f ∗pSrc2, Npp32f ∗pDst, int nLength)

  32-bit floating point signal times signal, then clamp to saturated value.

- NppStatus nppsMul_64f (const Npp64f ∗pSrc1, const Npp64f ∗pSrc2, Npp64f ∗pDst, int nLength)

  64-bit floating point signal times signal, then clamp to saturated value.

- NppStatus nppsMul_32fc (const Npp32fc ∗pSrc1, const Npp32fc ∗pSrc2, Npp32fc ∗pDst, int nLength)

  32-bit complex floating point signal times signal, then clamp to saturated value.

- NppStatus nppsMul_64fc (const Npp64fc ∗pSrc1, const Npp64fc ∗pSrc2, Npp64fc ∗pDst, int nLength)

  64-bit complex floating point signal times signal, then clamp to saturated value.

- NppStatus nppsMul_8u16u (const Npp8u ∗pSrc1, const Npp8u ∗pSrc2, Npp16u ∗pDst, int nLength)

  8-bit unsigned char signal times signal with 16-bit unsigned result, then clamp to saturated value.

- NppStatus nppsMul_16s32f (const Npp16s ∗pSrc1, const Npp16s ∗pSrc2, Npp32f ∗pDst, int nLength)

    16-bit signed short signal times signal with 32-bit floating point result, then clamp to saturated value.

- NppStatus nppsMul_32f32fc (const Npp32f ∗pSrc1, const Npp32fc ∗pSrc2, Npp32fc ∗pDst, int nLength)

    32-bit floating point signal times 32-bit complex floating point signal with complex 32-bit floating point result, then clamp to saturated value.

- NppStatus nppsMul_8u_Sfs (const Npp8u ∗pSrc1, const Npp8u ∗pSrc2, Npp8u ∗pDst, int nLength, int nScaleFactor)

    8-bit unsigned char signal times signal, scale, then clamp to saturated value.

- NppStatus nppsMul_16u_Sfs (const Npp16u ∗pSrc1, const Npp16u ∗pSrc2, Npp16u ∗pDst, int nLength, int nScaleFactor)

    16-bit unsigned short signal time signal, scale, then clamp to saturated value.

- NppStatus nppsMul_16s_Sfs (const Npp16s ∗pSrc1, const Npp16s ∗pSrc2, Npp16s ∗pDst, int nLength, int nScaleFactor)

    16-bit signed short signal times signal, scale, then clamp to saturated value.

- NppStatus nppsMul_32s_Sfs (const Npp32s ∗pSrc1, const Npp32s ∗pSrc2, Npp32s ∗pDst, int nLength, int nScaleFactor)

    32-bit signed integer signal times signal, scale, then clamp to saturated value.

- NppStatus nppsMul_16sc_Sfs (const Npp16sc ∗pSrc1, const Npp16sc ∗pSrc2, Npp16sc ∗pDst, int nLength, int nScaleFactor)

    16-bit signed complex short signal times signal, scale, then clamp to saturated value.

- NppStatus nppsMul_32sc_Sfs (const Npp32sc ∗pSrc1, const Npp32sc ∗pSrc2, Npp32sc ∗pDst, int nLength, int nScaleFactor)

    32-bit signed complex integer signal times signal, scale, then clamp to saturated value.

- NppStatus nppsMul_16u16s_Sfs (const Npp16u ∗pSrc1, const Npp16s ∗pSrc2, Npp16s ∗pDst, int nLength, int nScaleFactor)

    16-bit unsigned short signal times 16-bit signed short signal, scale, then clamp to 16-bit signed saturated value.

- NppStatus nppsMul_16s32s_Sfs (const Npp16s ∗pSrc1, const Npp16s ∗pSrc2, Npp32s ∗pDst, int nLength, int nScaleFactor)

    16-bit signed short signal times signal, scale, then clamp to 32-bit signed saturated value.

- NppStatus nppsMul_32s32sc_Sfs (const Npp32s ∗pSrc1, const Npp32sc ∗pSrc2, Npp32sc ∗pDst, int nLength, int nScaleFactor)

    32-bit signed integer signal times 32-bit complex signed integer signal, scale, then clamp to 32-bit complex integer saturated value.

- NppStatus nppsMul_Low_32s_Sfs (const Npp32s ∗pSrc1, const Npp32s ∗pSrc2, Npp32s ∗pDst, int nLength, int nScaleFactor)

    32-bit signed integer signal times signal, scale, then clamp to saturated value.

- NppStatus nppsMul_16s_I (const Npp16s ∗pSrc, Npp16s ∗pSrcDst, int nLength)

  16-bit signed short in place signal times signal, then clamp to saturated value.

- NppStatus nppsMul_32f_I (const Npp32f ∗pSrc, Npp32f ∗pSrcDst, int nLength)

  32-bit floating point in place signal times signal, then clamp to saturated value.

- NppStatus nppsMul_64f_I (const Npp64f ∗pSrc, Npp64f ∗pSrcDst, int nLength)

  64-bit floating point in place signal times signal, then clamp to saturated value.

- NppStatus nppsMul_32fc_I (const Npp32fc ∗pSrc, Npp32fc ∗pSrcDst, int nLength)

  32-bit complex floating point in place signal times signal, then clamp to saturated value.

- NppStatus nppsMul_64fc_I (const Npp64fc ∗pSrc, Npp64fc ∗pSrcDst, int nLength)

  64-bit complex floating point in place signal times signal, then clamp to saturated value.

- NppStatus nppsMul_32f32fc_I (const Npp32f ∗pSrc, Npp32fc ∗pSrcDst, int nLength)

  32-bit complex floating point in place signal times 32-bit floating point signal, then clamp to 32-bit complex floating point saturated value.

- NppStatus nppsMul_8u_ISfs (const Npp8u ∗pSrc, Npp8u ∗pSrcDst, int nLength, int nScaleFactor)

  8-bit unsigned char in place signal times signal, with scaling, then clamp to saturated value.

- NppStatus nppsMul_16u_ISfs (const Npp16u ∗pSrc, Npp16u ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit unsigned short in place signal times signal, with scaling, then clamp to saturated value.

- NppStatus nppsMul_16s_ISfs (const Npp16s ∗pSrc, Npp16s ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit signed short in place signal times signal, with scaling, then clamp to saturated value.

- NppStatus nppsMul_32s_ISfs (const Npp32s ∗pSrc, Npp32s ∗pSrcDst, int nLength, int nScaleFactor)

  32-bit signed integer in place signal times signal, with scaling, then clamp to saturated value.

- NppStatus nppsMul_16sc_ISfs (const Npp16sc ∗pSrc, Npp16sc ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit complex signed short in place signal times signal, with scaling, then clamp to saturated value.

- NppStatus nppsMul_32sc_ISfs (const Npp32sc ∗pSrc, Npp32sc ∗pSrcDst, int nLength, int nScaleFactor)

  32-bit complex signed integer in place signal times signal, with scaling, then clamp to saturated value.

- NppStatus nppsMul_32s32sc_ISfs (const Npp32s ∗pSrc, Npp32sc ∗pSrcDst, int nLength, int nScaleFactor)

  32-bit complex signed integer in place signal times 32-bit signed integer signal, with scaling, then clamp to saturated value.

Sub Signal

Sample by sample subtraction of the samples of two signals.

- NppStatus nppsSub_16s (const Npp16s ∗pSrc1, const Npp16s ∗pSrc2, Npp16s ∗pDst, int nLength)

  16-bit signed short signal subtract signal, then clamp to saturated value.

- NppStatus nppsSub_32f (const Npp32f ∗pSrc1, const Npp32f ∗pSrc2, Npp32f ∗pDst, int nLength)

  32-bit floating point signal subtract signal, then clamp to saturated value.

- NppStatus nppsSub_64f (const Npp64f ∗pSrc1, const Npp64f ∗pSrc2, Npp64f ∗pDst, int nLength)

  64-bit floating point signal subtract signal, then clamp to saturated value.

- NppStatus nppsSub_32fc (const Npp32fc ∗pSrc1, const Npp32fc ∗pSrc2, Npp32fc ∗pDst, int nLength)

  32-bit complex floating point signal subtract signal, then clamp to saturated value.

- NppStatus nppsSub_64fc (const Npp64fc ∗pSrc1, const Npp64fc ∗pSrc2, Npp64fc ∗pDst, int nLength)

  64-bit complex floating point signal subtract signal, then clamp to saturated value.

- NppStatus nppsSub_16s32f (const Npp16s ∗pSrc1, const Npp16s ∗pSrc2, Npp32f ∗pDst, int nLength)

  16-bit signed short signal subtract 16-bit signed short signal, then clamp and convert to 32-bit floating point saturated value.

- NppStatus nppsSub_8u_Sfs (const Npp8u ∗pSrc1, const Npp8u ∗pSrc2, Npp8u ∗pDst, int nLength, int nScaleFactor)

  8-bit unsigned char signal subtract signal, scale, then clamp to saturated value.

- NppStatus nppsSub_16u_Sfs (const Npp16u ∗pSrc1, const Npp16u ∗pSrc2, Npp16u ∗pDst, int nLength, int nScaleFactor)

  16-bit unsigned short signal subtract signal, scale, then clamp to saturated value.

- NppStatus nppsSub_16s_Sfs (const Npp16s ∗pSrc1, const Npp16s ∗pSrc2, Npp16s ∗pDst, int nLength, int nScaleFactor)

  16-bit signed short signal subtract signal, scale, then clamp to saturated value.

- NppStatus nppsSub_32s_Sfs (const Npp32s ∗pSrc1, const Npp32s ∗pSrc2, Npp32s ∗pDst, int nLength, int nScaleFactor)

  32-bit signed integer signal subtract signal, scale, then clamp to saturated value.

- NppStatus nppsSub_16sc_Sfs (const Npp16sc ∗pSrc1, const Npp16sc ∗pSrc2, Npp16sc ∗pDst, int nLength, int nScaleFactor)

  16-bit signed complex short signal subtract signal, scale, then clamp to saturated value.

- NppStatus nppsSub_32sc_Sfs (const Npp32sc ∗pSrc1, const Npp32sc ∗pSrc2, Npp32sc ∗pDst, int nLength, int nScaleFactor)

  32-bit signed complex integer signal subtract signal, scale, then clamp to saturated value.

- NppStatus nppsSub_16s_I (const Npp16s ∗pSrc, Npp16s ∗pSrcDst, int nLength)

16-bit signed short in place signal subtract signal, then clamp to saturated value.

- NppStatus nppsSub_32f_I (const Npp32f ∗pSrc, Npp32f ∗pSrcDst, int nLength)

  32-bit floating point in place signal subtract signal, then clamp to saturated value.

- NppStatus nppsSub_64f_I (const Npp64f ∗pSrc, Npp64f ∗pSrcDst, int nLength)

  64-bit floating point in place signal subtract signal, then clamp to saturated value.

- NppStatus nppsSub_32fc_I (const Npp32fc ∗pSrc, Npp32fc ∗pSrcDst, int nLength)

  32-bit complex floating point in place signal subtract signal, then clamp to saturated value.

- NppStatus nppsSub_64fc_I (const Npp64fc ∗pSrc, Npp64fc ∗pSrcDst, int nLength)

  64-bit complex floating point in place signal subtract signal, then clamp to saturated value.

- NppStatus nppsSub_8u_ISfs (const Npp8u ∗pSrc, Npp8u ∗pSrcDst, int nLength, int nScaleFactor)

  8-bit unsigned char in place signal subtract signal, with scaling, then clamp to saturated value.

- NppStatus nppsSub_16u_ISfs (const Npp16u ∗pSrc, Npp16u ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit unsigned short in place signal subtract signal, with scaling, then clamp to saturated value.

- NppStatus nppsSub_16s_ISfs (const Npp16s ∗pSrc, Npp16s ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit signed short in place signal subtract signal, with scaling, then clamp to saturated value.

- NppStatus nppsSub_32s_ISfs (const Npp32s ∗pSrc, Npp32s ∗pSrcDst, int nLength, int nScaleFactor)

  32-bit signed integer in place signal subtract signal, with scaling, then clamp to saturated value.

- NppStatus nppsSub_16sc_ISfs (const Npp16sc ∗pSrc, Npp16sc ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit complex signed short in place signal subtract signal, with scaling, then clamp to saturated value.

- NppStatus nppsSub_32sc_ISfs (const Npp32sc ∗pSrc, Npp32sc ∗pSrcDst, int nLength, int nScaleFactor)

  32-bit complex signed integer in place signal subtract signal, with scaling, then clamp to saturated value.

Div Signal

Sample by sample division of the samples of two signals.

- NppStatus nppsDiv_8u_Sfs (const Npp8u ∗pSrc1, const Npp8u ∗pSrc2, Npp8u ∗pDst, int nLength, int nScaleFactor)

  8-bit unsigned char signal divide signal, scale, then clamp to saturated value.

- NppStatus nppsDiv_16u_Sfs (const Npp16u ∗pSrc1, const Npp16u ∗pSrc2, Npp16u ∗pDst, int nLength, int nScaleFactor)

  16-bit unsigned short signal divide signal, scale, then clamp to saturated value.

- NppStatus nppsDiv_16s_Sfs (const Npp16s ∗pSrc1, const Npp16s ∗pSrc2, Npp16s ∗pDst, int nLength, int nScaleFactor)

  16-bit signed short signal divide signal, scale, then clamp to saturated value.

- NppStatus nppsDiv_32s_Sfs (const Npp32s ∗pSrc1, const Npp32s ∗pSrc2, Npp32s ∗pDst, int nLength, int nScaleFactor)

  32-bit signed integer signal divide signal, scale, then clamp to saturated value.

- NppStatus nppsDiv_16sc_Sfs (const Npp16sc ∗pSrc1, const Npp16sc ∗pSrc2, Npp16sc ∗pDst, int nLength, int nScaleFactor)

  16-bit signed complex short signal divide signal, scale, then clamp to saturated value.

- NppStatus nppsDiv_32s16s_Sfs (const Npp16s ∗pSrc1, const Npp32s ∗pSrc2, Npp16s ∗pDst, int nLength, int nScaleFactor)

  32-bit signed integer signal divided by 16-bit signed short signal, scale, then clamp to 16-bit signed short saturated value.

- NppStatus nppsDiv_32f (const Npp32f ∗pSrc1, const Npp32f ∗pSrc2, Npp32f ∗pDst, int nLength)

  32-bit floating point signal divide signal, then clamp to saturated value.

- NppStatus nppsDiv_64f (const Npp64f ∗pSrc1, const Npp64f ∗pSrc2, Npp64f ∗pDst, int nLength)

  64-bit floating point signal divide signal, then clamp to saturated value.

- NppStatus nppsDiv_32fc (const Npp32fc ∗pSrc1, const Npp32fc ∗pSrc2, Npp32fc ∗pDst, int nLength)

  32-bit complex floating point signal divide signal, then clamp to saturated value.

- NppStatus nppsDiv_64fc (const Npp64fc ∗pSrc1, const Npp64fc ∗pSrc2, Npp64fc ∗pDst, int nLength)

  64-bit complex floating point signal divide signal, then clamp to saturated value.

- NppStatus nppsDiv_8u_ISfs (const Npp8u ∗pSrc, Npp8u ∗pSrcDst, int nLength, int nScaleFactor)

  8-bit unsigned char in place signal divide signal, with scaling, then clamp to saturated value.

- NppStatus nppsDiv_16u_ISfs (const Npp16u ∗pSrc, Npp16u ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit unsigned short in place signal divide signal, with scaling, then clamp to saturated value.

- NppStatus nppsDiv_16s_ISfs (const Npp16s ∗pSrc, Npp16s ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit signed short in place signal divide signal, with scaling, then clamp to saturated value.

- NppStatus nppsDiv_16sc_ISfs (const Npp16sc ∗pSrc, Npp16sc ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit complex signed short in place signal divide signal, with scaling, then clamp to saturated value.

- NppStatus nppsDiv_32s_ISfs (const Npp32s ∗pSrc, Npp32s ∗pSrcDst, int nLength, int nScaleFactor)

  32-bit signed integer in place signal divide signal, with scaling, then clamp to saturated value.

- NppStatus nppsDiv_32f_I (const Npp32f ∗pSrc, Npp32f ∗pSrcDst, int nLength)

32-bit floating point in place signal divide signal, then clamp to saturated value.

- NppStatus nppsDiv_64f_I (const Npp64f ∗pSrc, Npp64f ∗pSrcDst, int nLength)

  64-bit floating point in place signal divide signal, then clamp to saturated value.

- NppStatus nppsDiv_32fc_I (const Npp32fc ∗pSrc, Npp32fc ∗pSrcDst, int nLength)

  32-bit complex floating point in place signal divide signal, then clamp to saturated value.

- NppStatus nppsDiv_64fc_I (const Npp64fc ∗pSrc, Npp64fc ∗pSrcDst, int nLength)

  64-bit complex floating point in place signal divide signal, then clamp to saturated value.

Div_Round Signal

Sample by sample division of the samples of two signals with rounding.

- NppStatus nppsDiv_Round_8u_Sfs (const Npp8u ∗pSrc1, const Npp8u ∗pSrc2, Npp8u ∗pDst, int nLength, NppRoundMode nRndMode, int nScaleFactor)

  8-bit unsigned char signal divide signal, scale, then clamp to saturated value.

- NppStatus nppsDiv_Round_16u_Sfs (const Npp16u ∗pSrc1, const Npp16u ∗pSrc2, Npp16u ∗pDst, int nLength, NppRoundMode nRndMode, int nScaleFactor)

  16-bit unsigned short signal divide signal, scale, round, then clamp to saturated value.

- NppStatus nppsDiv_Round_16s_Sfs (const Npp16s ∗pSrc1, const Npp16s ∗pSrc2, Npp16s ∗pDst, int nLength, NppRoundMode nRndMode, int nScaleFactor)

  16-bit signed short signal divide signal, scale, round, then clamp to saturated value.

- NppStatus nppsDiv_Round_8u_ISfs (const Npp8u ∗pSrc, Npp8u ∗pSrcDst, int nLength, NppRoundMode nRndMode, int nScaleFactor)

  8-bit unsigned char in place signal divide signal, with scaling, rounding then clamp to saturated value.

- NppStatus nppsDiv_Round_16u_ISfs (const Npp16u ∗pSrc, Npp16u ∗pSrcDst, int nLength, NppRoundMode nRndMode, int nScaleFactor)

  16-bit unsigned short in place signal divide signal, with scaling, rounding then clamp to saturated value.

- NppStatus nppsDiv_Round_16s_ISfs (const Npp16s ∗pSrc, Npp16s ∗pSrcDst, int nLength, NppRoundMode nRndMode, int nScaleFactor)

  16-bit signed short in place signal divide signal, with scaling, rounding then clamp to saturated value.

Absolute Value Signal

Absolute value of each sample of a signal.

- NppStatus nppsAbs_16s (const Npp16s ∗pSrc, Npp16s ∗pDst, int nLength)

  16-bit signed short signal absolute value.

- NppStatus nppsAbs_32s (const Npp32s ∗pSrc, Npp32s ∗pDst, int nLength)

32-bit signed integer signal absolute value.

- NppStatus nppsAbs_32f (const Npp32f ∗pSrc, Npp32f ∗pDst, int nLength)
  32-bit floating point signal absolute value.
- NppStatus nppsAbs_64f (const Npp64f ∗pSrc, Npp64f ∗pDst, int nLength)
  64-bit floating point signal absolute value.
- NppStatus nppsAbs_16s_I (Npp16s ∗pSrcDst, int nLength)
  16-bit signed short signal absolute value.
- NppStatus nppsAbs_32s_I (Npp32s ∗pSrcDst, int nLength)
  32-bit signed integer signal absolute value.
- NppStatus nppsAbs_32f_I (Npp32f ∗pSrcDst, int nLength)
  32-bit floating point signal absolute value.
- NppStatus nppsAbs_64f_I (Npp64f ∗pSrcDst, int nLength)
  64-bit floating point signal absolute value.

Square Signal

Squares each sample of a signal.

- NppStatus nppsSqr_32f (const Npp32f ∗pSrc, Npp32f ∗pDst, int nLength)
  32-bit floating point signal squared.
- NppStatus nppsSqr_64f (const Npp64f ∗pSrc, Npp64f ∗pDst, int nLength)
  64-bit floating point signal squared.
- NppStatus nppsSqr_32fc (const Npp32fc ∗pSrc, Npp32fc ∗pDst, int nLength)
  32-bit complex floating point signal squared.
- NppStatus nppsSqr_64fc (const Npp64fc ∗pSrc, Npp64fc ∗pDst, int nLength)
  64-bit complex floating point signal squared.
- NppStatus nppsSqr_32f_I (Npp32f ∗pSrcDst, int nLength)
  32-bit floating point signal squared.
- NppStatus nppsSqr_64f_I (Npp64f ∗pSrcDst, int nLength)
  64-bit floating point signal squared.
- NppStatus nppsSqr_32fc_I (Npp32fc ∗pSrcDst, int nLength)
  32-bit complex floating point signal squared.
- NppStatus nppsSqr_64fc_I (Npp64fc ∗pSrcDst, int nLength)
  64-bit complex floating point signal squared.
- NppStatus nppsSqr_8u_Sfs (const Npp8u ∗pSrc, Npp8u ∗pDst, int nLength, int nScaleFactor)

8-bit unsigned char signal squared, scale, then clamp to saturated value.

- NppStatus nppsSqr_16u_Sfs (const Npp16u ∗pSrc, Npp16u ∗pDst, int nLength, int nScaleFactor)

  16-bit unsigned short signal squared, scale, then clamp to saturated value.

- NppStatus nppsSqr_16s_Sfs (const Npp16s ∗pSrc, Npp16s ∗pDst, int nLength, int nScaleFactor)

  16-bit signed short signal squared, scale, then clamp to saturated value.

- NppStatus nppsSqr_16sc_Sfs (const Npp16sc ∗pSrc, Npp16sc ∗pDst, int nLength, int nScaleFactor)

  16-bit complex signed short signal squared, scale, then clamp to saturated value.

- NppStatus nppsSqr_8u_ISfs (Npp8u ∗pSrcDst, int nLength, int nScaleFactor)

  8-bit unsigned char signal squared, scale, then clamp to saturated value.

- NppStatus nppsSqr_16u_ISfs (Npp16u ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit unsigned short signal squared, scale, then clamp to saturated value.

- NppStatus nppsSqr_16s_ISfs (Npp16s ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit signed short signal squared, scale, then clamp to saturated value.

- NppStatus nppsSqr_16sc_ISfs (Npp16sc ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit complex signed short signal squared, scale, then clamp to saturated value.

Square Root Signal

Square root of each sample of a signal.

- NppStatus nppsSqrt_32f (const Npp32f ∗pSrc, Npp32f ∗pDst, int nLength)

  32-bit floating point signal square root.

- NppStatus nppsSqrt_64f (const Npp64f ∗pSrc, Npp64f ∗pDst, int nLength)

  64-bit floating point signal square root.

- NppStatus nppsSqrt_32fc (const Npp32fc ∗pSrc, Npp32fc ∗pDst, int nLength)

  32-bit complex floating point signal square root.

- NppStatus nppsSqrt_64fc (const Npp64fc ∗pSrc, Npp64fc ∗pDst, int nLength)

  64-bit complex floating point signal square root.

- NppStatus nppsSqrt_32f_I (Npp32f ∗pSrcDst, int nLength)

  32-bit floating point signal square root.

- NppStatus nppsSqrt_64f_I (Npp64f ∗pSrcDst, int nLength)

  64-bit floating point signal square root.

- NppStatus nppsSqrt_32fc_I (Npp32fc ∗pSrcDst, int nLength)

  32-bit complex floating point signal square root.

- NppStatus nppsSqrt_64fc_I (Npp64fc ∗pSrcDst, int nLength)
  64-bit complex floating point signal square root.
- NppStatus nppsSqrt_8u_Sfs (const Npp8u ∗pSrc, Npp8u ∗pDst, int nLength, int nScaleFactor)
  8-bit unsigned char signal square root, scale, then clamp to saturated value.
- NppStatus nppsSqrt_16u_Sfs (const Npp16u ∗pSrc, Npp16u ∗pDst, int nLength, int nScaleFactor)
  16-bit unsigned short signal square root, scale, then clamp to saturated value.
- NppStatus nppsSqrt_16s_Sfs (const Npp16s ∗pSrc, Npp16s ∗pDst, int nLength, int nScaleFactor)
  16-bit signed short signal square root, scale, then clamp to saturated value.
- NppStatus nppsSqrt_16sc_Sfs (const Npp16sc ∗pSrc, Npp16sc ∗pDst, int nLength, int nScaleFactor)
  16-bit complex signed short signal square root, scale, then clamp to saturated value.
- NppStatus nppsSqrt_64s_Sfs (const Npp64s ∗pSrc, Npp64s ∗pDst, int nLength, int nScaleFactor)
  64-bit signed integer signal square root, scale, then clamp to saturated value.
- NppStatus nppsSqrt_32s16s_Sfs (const Npp32s ∗pSrc, Npp16s ∗pDst, int nLength, int nScaleFactor)
  32-bit signed integer signal square root, scale, then clamp to 16-bit signed integer saturated value.
- NppStatus nppsSqrt_64s16s_Sfs (const Npp64s ∗pSrc, Npp16s ∗pDst, int nLength, int nScaleFactor)
  64-bit signed integer signal square root, scale, then clamp to 16-bit signed integer saturated value.
- NppStatus nppsSqrt_8u_ISfs (Npp8u ∗pSrcDst, int nLength, int nScaleFactor)
  8-bit unsigned char signal square root, scale, then clamp to saturated value.
- NppStatus nppsSqrt_16u_ISfs (Npp16u ∗pSrcDst, int nLength, int nScaleFactor)
  16-bit unsigned short signal square root, scale, then clamp to saturated value.
- NppStatus nppsSqrt_16s_ISfs (Npp16s ∗pSrcDst, int nLength, int nScaleFactor)
  16-bit signed short signal square root, scale, then clamp to saturated value.
- NppStatus nppsSqrt_16sc_ISfs (Npp16sc ∗pSrcDst, int nLength, int nScaleFactor)
  16-bit complex signed short signal square root, scale, then clamp to saturated value.
- NppStatus nppsSqrt_64s_ISfs (Npp64s ∗pSrcDst, int nLength, int nScaleFactor)
  64-bit signed integer signal square root, scale, then clamp to saturated value.

Cube Root Signal

Cube root of each sample of a signal.

- NppStatus nppsCubrt_32f (const Npp32f ∗pSrc, Npp32f ∗pDst, int nLength)
  32-bit floating point signal cube root.

- NppStatus nppsCubrt_32s16s_Sfs (const Npp32s ∗pSrc, Npp16s ∗pDst, int nLength, int nScaleFactor)
  32-bit signed integer signal cube root, scale, then clamp to 16-bit signed integer saturated value.

Exponent Signal

E raised to the power of each sample of a signal.

- NppStatus nppsExp_32f (const Npp32f ∗pSrc, Npp32f ∗pDst, int nLength)
  32-bit floating point signal exponent.
- NppStatus nppsExp_64f (const Npp64f ∗pSrc, Npp64f ∗pDst, int nLength)
  64-bit floating point signal exponent.
- NppStatus nppsExp_32f64f (const Npp32f ∗pSrc, Npp64f ∗pDst, int nLength)
  32-bit floating point signal exponent with 64-bit floating point result.
- NppStatus nppsExp_32f_I (Npp32f ∗pSrcDst, int nLength)
  32-bit floating point signal exponent.
- NppStatus nppsExp_64f_I (Npp64f ∗pSrcDst, int nLength)
  64-bit floating point signal exponent.
- NppStatus nppsExp_16s_Sfs (const Npp16s ∗pSrc, Npp16s ∗pDst, int nLength, int nScaleFactor)
  16-bit signed short signal exponent, scale, then clamp to saturated value.
- NppStatus nppsExp_32s_Sfs (const Npp32s ∗pSrc, Npp32s ∗pDst, int nLength, int nScaleFactor)
  32-bit signed integer signal exponent, scale, then clamp to saturated value.
- NppStatus nppsExp_64s_Sfs (const Npp64s ∗pSrc, Npp64s ∗pDst, int nLength, int nScaleFactor)
  64-bit signed integer signal exponent, scale, then clamp to saturated value.
- NppStatus nppsExp_16s_ISfs (Npp16s ∗pSrcDst, int nLength, int nScaleFactor)
  16-bit signed short signal exponent, scale, then clamp to saturated value.
- NppStatus nppsExp_32s_ISfs (Npp32s ∗pSrcDst, int nLength, int nScaleFactor)
  32-bit signed integer signal exponent, scale, then clamp to saturated value.
- NppStatus nppsExp_64s_ISfs (Npp64s ∗pSrcDst, int nLength, int nScaleFactor)
  64-bit signed integer signal exponent, scale, then clamp to saturated value.

Natural Logarithm Signal

Natural logarithm of each sample of a signal.

- NppStatus nppsLn_32f (const Npp32f ∗pSrc, Npp32f ∗pDst, int nLength)
  32-bit floating point signal natural logarithm.

- NppStatus nppsLn_64f (const Npp64f *pSrc, Npp64f *pDst, int nLength)

  64-bit floating point signal natural logarithm.

- NppStatus nppsLn_64f32f (const Npp64f *pSrc, Npp32f *pDst, int nLength)

  64-bit floating point signal natural logarithm with 32-bit floating point result.

- NppStatus nppsLn_32f_I (Npp32f *pSrcDst, int nLength)

  32-bit floating point signal natural logarithm.

- NppStatus nppsLn_64f_I (Npp64f *pSrcDst, int nLength)

  64-bit floating point signal natural logarithm.

- NppStatus nppsLn_16s_Sfs (const Npp16s *pSrc, Npp16s *pDst, int nLength, int nScaleFactor)

  16-bit signed short signal natural logarithm, scale, then clamp to saturated value.

- NppStatus nppsLn_32s_Sfs (const Npp32s *pSrc, Npp32s *pDst, int nLength, int nScaleFactor)

  32-bit signed integer signal natural logarithm, scale, then clamp to saturated value.

- NppStatus nppsLn_32s16s_Sfs (const Npp32s *pSrc, Npp16s *pDst, int nLength, int nScaleFactor)

  32-bit signed integer signal natural logarithm, scale, then clamp to 16-bit signed short saturated value.

- NppStatus nppsLn_16s_ISfs (Npp16s *pSrcDst, int nLength, int nScaleFactor)

  16-bit signed short signal natural logarithm, scale, then clamp to saturated value.

- NppStatus nppsLn_32s_ISfs (Npp32s *pSrcDst, int nLength, int nScaleFactor)

  32-bit signed integer signal natural logarithm, scale, then clamp to saturated value.

Ten Times Base Ten Logarithm Signal

Ten times the decimal logarithm of each sample of a signal.

- NppStatus npps10Log10_32s_Sfs (const Npp32s *pSrc, Npp32s *pDst, int nLength, int nScaleFactor)

  32-bit signed integer signal 10 times base 10 logarithm, scale, then clamp to saturated value.

- NppStatus npps10Log10_32s_ISfs (Npp32s *pSrcDst, int nLength, int nScaleFactor)

  32-bit signed integer signal 10 times base 10 logarithm, scale, then clamp to saturated value.

Inverse Tangent Signal

Inverse tangent of each sample of a signal.

- NppStatus nppsArctan_32f (const Npp32f *pSrc, Npp32f *pDst, int nLength)

  32-bit floating point signal inverse tangent.

- NppStatus nppsArctan_64f (const Npp64f ∗pSrc, Npp64f ∗pDst, int nLength)

  64-bit floating point signal inverse tangent.

- NppStatus nppsArctan_32f_I (Npp32f ∗pSrcDst, int nLength)

  32-bit floating point signal inverse tangent.

- NppStatus nppsArctan_64f_I (Npp64f ∗pSrcDst, int nLength)

  64-bit floating point signal inverse tangent.

Normalize Signal

Normalize each sample of a real or complex signal using offset and division operations.

- NppStatus nppsNormalize_32f (const Npp32f ∗pSrc, Npp32f ∗pDst, int nLength, Npp32f vSub, Npp32f vDiv)

  32-bit floating point signal normalize.

- NppStatus nppsNormalize_32fc (const Npp32fc ∗pSrc, Npp32fc ∗pDst, int nLength, Npp32fc vSub, Npp32f vDiv)

  32-bit complex floating point signal normalize.

- NppStatus nppsNormalize_64f (const Npp64f ∗pSrc, Npp64f ∗pDst, int nLength, Npp64f vSub, Npp64f vDiv)

  64-bit floating point signal normalize.

- NppStatus nppsNormalize_64fc (const Npp64fc ∗pSrc, Npp64fc ∗pDst, int nLength, Npp64fc vSub, Npp64f vDiv)

  64-bit complex floating point signal normalize.

- NppStatus nppsNormalize_16s_Sfs (const Npp16s ∗pSrc, Npp16s ∗pDst, int nLength, Npp16s vSub, int vDiv, int nScaleFactor)

  16-bit signed short signal normalize, scale, then clamp to saturated value.

- NppStatus nppsNormalize_16sc_Sfs (const Npp16sc ∗pSrc, Npp16sc ∗pDst, int nLength, Npp16sc vSub, int vDiv, int nScaleFactor)

  16-bit complex signed short signal normalize, scale, then clamp to saturated value.

Cauchy, CauchyD, and CauchyDD2 Signal

Determine Cauchy robust error function and its first and second derivatives for each sample of a signal.

- NppStatus nppsCauchy_32f_I (Npp32f ∗pSrcDst, int nLength, Npp32f nParam)

  32-bit floating point signal Cauchy error calculation.

- NppStatus nppsCauchyD_32f_I (Npp32f ∗pSrcDst, int nLength, Npp32f nParam)

  32-bit floating point signal Cauchy first derivative.

- NppStatus nppsCauchyDD2_32f_I (Npp32f ∗pSrcDst, Npp32f ∗pD2FVal, int nLength, Npp32f nParam)

  32-bit floating point signal Cauchy first and second derivatives.

AndC

Bitwise AND of a constant and each sample of a signal.

- NppStatus nppsAndC_8u (const Npp8u ∗pSrc, Npp8u nValue, Npp8u ∗pDst, int nLength)

  8-bit unsigned char signal and with constant.

- NppStatus nppsAndC_16u (const Npp16u ∗pSrc, Npp16u nValue, Npp16u ∗pDst, int nLength)

  16-bit unsigned short signal and with constant.

- NppStatus nppsAndC_32u (const Npp32u ∗pSrc, Npp32u nValue, Npp32u ∗pDst, int nLength)

  32-bit unsigned integer signal and with constant.

- NppStatus nppsAndC_8u_I (Npp8u nValue, Npp8u ∗pSrcDst, int nLength)

  8-bit unsigned char in place signal and with constant.

- NppStatus nppsAndC_16u_I (Npp16u nValue, Npp16u ∗pSrcDst, int nLength)

  16-bit unsigned short in place signal and with constant.

- NppStatus nppsAndC_32u_I (Npp32u nValue, Npp32u ∗pSrcDst, int nLength)

  32-bit unsigned signed integer in place signal and with constant.

And

Sample by sample bitwise AND of samples from two signals.

- NppStatus nppsAnd_8u (const Npp8u ∗pSrc1, const Npp8u ∗pSrc2, Npp8u ∗pDst, int nLength)

  8-bit unsigned char signal and with signal.

- NppStatus nppsAnd_16u (const Npp16u ∗pSrc1, const Npp16u ∗pSrc2, Npp16u ∗pDst, int nLength)

  16-bit unsigned short signal and with signal.

- NppStatus nppsAnd_32u (const Npp32u ∗pSrc1, const Npp32u ∗pSrc2, Npp32u ∗pDst, int nLength)

  32-bit unsigned integer signal and with signal.

- NppStatus nppsAnd_8u_I (const Npp8u ∗pSrc, Npp8u ∗pSrcDst, int nLength)

  8-bit unsigned char in place signal and with signal.

- NppStatus nppsAnd_16u_I (const Npp16u ∗pSrc, Npp16u ∗pSrcDst, int nLength)

  16-bit unsigned short in place signal and with signal.

- NppStatus nppsAnd_32u_I (const Npp32u ∗pSrc, Npp32u ∗pSrcDst, int nLength)
  32-bit unsigned integer in place signal and with signal.

OrC

Bitwise OR of a constant and each sample of a signal.

- NppStatus nppsOrC_8u (const Npp8u ∗pSrc, Npp8u nValue, Npp8u ∗pDst, int nLength)
  8-bit unsigned char signal or with constant.
- NppStatus nppsOrC_16u (const Npp16u ∗pSrc, Npp16u nValue, Npp16u ∗pDst, int nLength)
  16-bit unsigned short signal or with constant.
- NppStatus nppsOrC_32u (const Npp32u ∗pSrc, Npp32u nValue, Npp32u ∗pDst, int nLength)
  32-bit unsigned integer signal or with constant.
- NppStatus nppsOrC_8u_I (Npp8u nValue, Npp8u ∗pSrcDst, int nLength)
  8-bit unsigned char in place signal or with constant.
- NppStatus nppsOrC_16u_I (Npp16u nValue, Npp16u ∗pSrcDst, int nLength)
  16-bit unsigned short in place signal or with constant.
- NppStatus nppsOrC_32u_I (Npp32u nValue, Npp32u ∗pSrcDst, int nLength)
  32-bit unsigned signed integer in place signal or with constant.

Or

Sample by sample bitwise OR of the samples from two signals.

- NppStatus nppsOr_8u (const Npp8u ∗pSrc1, const Npp8u ∗pSrc2, Npp8u ∗pDst, int nLength)
  8-bit unsigned char signal or with signal.
- NppStatus nppsOr_16u (const Npp16u ∗pSrc1, const Npp16u ∗pSrc2, Npp16u ∗pDst, int nLength)
  16-bit unsigned short signal or with signal.
- NppStatus nppsOr_32u (const Npp32u ∗pSrc1, const Npp32u ∗pSrc2, Npp32u ∗pDst, int nLength)
  32-bit unsigned integer signal or with signal.
- NppStatus nppsOr_8u_I (const Npp8u ∗pSrc, Npp8u ∗pSrcDst, int nLength)
  8-bit unsigned char in place signal or with signal.
- NppStatus nppsOr_16u_I (const Npp16u ∗pSrc, Npp16u ∗pSrcDst, int nLength)
  16-bit unsigned short in place signal or with signal.
- NppStatus nppsOr_32u_I (const Npp32u ∗pSrc, Npp32u ∗pSrcDst, int nLength)
  32-bit unsigned integer in place signal or with signal.

XorC

Bitwise XOR of a constant and each sample of a signal.

- NppStatus nppsXorC_8u (const Npp8u ∗pSrc, Npp8u nValue, Npp8u ∗pDst, int nLength)

  8-bit unsigned char signal exclusive or with constant.

- NppStatus nppsXorC_16u (const Npp16u ∗pSrc, Npp16u nValue, Npp16u ∗pDst, int nLength)

  16-bit unsigned short signal exclusive or with constant.

- NppStatus nppsXorC_32u (const Npp32u ∗pSrc, Npp32u nValue, Npp32u ∗pDst, int nLength)

  32-bit unsigned integer signal exclusive or with constant.

- NppStatus nppsXorC_8u_I (Npp8u nValue, Npp8u ∗pSrcDst, int nLength)

  8-bit unsigned char in place signal exclusive or with constant.

- NppStatus nppsXorC_16u_I (Npp16u nValue, Npp16u ∗pSrcDst, int nLength)

  16-bit unsigned short in place signal exclusive or with constant.

- NppStatus nppsXorC_32u_I (Npp32u nValue, Npp32u ∗pSrcDst, int nLength)

  32-bit unsigned signed integer in place signal exclusive or with constant.

Xor

Sample by sample bitwise XOR of the samples from two signals.

- NppStatus nppsXor_8u (const Npp8u ∗pSrc1, const Npp8u ∗pSrc2, Npp8u ∗pDst, int nLength)

  8-bit unsigned char signal exclusive or with signal.

- NppStatus nppsXor_16u (const Npp16u ∗pSrc1, const Npp16u ∗pSrc2, Npp16u ∗pDst, int nLength)

  16-bit unsigned short signal exclusive or with signal.

- NppStatus nppsXor_32u (const Npp32u ∗pSrc1, const Npp32u ∗pSrc2, Npp32u ∗pDst, int nLength)

  32-bit unsigned integer signal exclusive or with signal.

- NppStatus nppsXor_8u_I (const Npp8u ∗pSrc, Npp8u ∗pSrcDst, int nLength)

  8-bit unsigned char in place signal exclusive or with signal.

- NppStatus nppsXor_16u_I (const Npp16u ∗pSrc, Npp16u ∗pSrcDst, int nLength)

  16-bit unsigned short in place signal exclusive or with signal.

- NppStatus nppsXor_32u_I (const Npp32u ∗pSrc, Npp32u ∗pSrcDst, int nLength)

  32-bit unsigned integer in place signal exclusive or with signal.

Not

Bitwise NOT of each sample of a signal.

- NppStatus nppsNot_8u (const Npp8u ∗pSrc, Npp8u ∗pDst, int nLength)
  8-bit unsigned char not signal.
- NppStatus nppsNot_16u (const Npp16u ∗pSrc, Npp16u ∗pDst, int nLength)
  16-bit unsigned short not signal.
- NppStatus nppsNot_32u (const Npp32u ∗pSrc, Npp32u ∗pDst, int nLength)
  32-bit unsigned integer not signal.
- NppStatus nppsNot_8u_I (Npp8u ∗pSrcDst, int nLength)
  8-bit unsigned char in place not signal.
- NppStatus nppsNot_16u_I (Npp16u ∗pSrcDst, int nLength)
  16-bit unsigned short in place not signal.
- NppStatus nppsNot_32u_I (Npp32u ∗pSrcDst, int nLength)
  32-bit unsigned signed integer in place not signal.

LShiftC

Left shifts the bits of each sample of a signal by a constant amount.

- NppStatus nppsLShiftC_8u (const Npp8u ∗pSrc, int nValue, Npp8u ∗pDst, int nLength)
  8-bit unsigned char signal left shift with constant.
- NppStatus nppsLShiftC_16u (const Npp16u ∗pSrc, int nValue, Npp16u ∗pDst, int nLength)
  16-bit unsigned short signal left shift with constant.
- NppStatus nppsLShiftC_16s (const Npp16s ∗pSrc, int nValue, Npp16s ∗pDst, int nLength)
  16-bit signed short signal left shift with constant.
- NppStatus nppsLShiftC_32u (const Npp32u ∗pSrc, int nValue, Npp32u ∗pDst, int nLength)
  32-bit unsigned integer signal left shift with constant.
- NppStatus nppsLShiftC_32s (const Npp32s ∗pSrc, int nValue, Npp32s ∗pDst, int nLength)
  32-bit signed integer signal left shift with constant.
- NppStatus nppsLShiftC_8u_I (int nValue, Npp8u ∗pSrcDst, int nLength)
  8-bit unsigned char in place signal left shift with constant.
- NppStatus nppsLShiftC_16u_I (int nValue, Npp16u ∗pSrcDst, int nLength)
  16-bit unsigned short in place signal left shift with constant.
- NppStatus nppsLShiftC_16s_I (int nValue, Npp16s ∗pSrcDst, int nLength)

16-bit signed short in place signal left shift with constant.

- NppStatus nppsLShiftC_32u_I (int nValue, Npp32u ∗pSrcDst, int nLength)
  32-bit unsigned signed integer in place signal left shift with constant.
- NppStatus nppsLShiftC_32s_I (int nValue, Npp32s ∗pSrcDst, int nLength)
  32-bit signed signed integer in place signal left shift with constant.

RShiftC

Right shifts the bits of each sample of a signal by a constant amount.

- NppStatus nppsRShiftC_8u (const Npp8u ∗pSrc, int nValue, Npp8u ∗pDst, int nLength)
  8-bit unsigned char signal right shift with constant.
- NppStatus nppsRShiftC_16u (const Npp16u ∗pSrc, int nValue, Npp16u ∗pDst, int nLength)
  16-bit unsigned short signal right shift with constant.
- NppStatus nppsRShiftC_16s (const Npp16s ∗pSrc, int nValue, Npp16s ∗pDst, int nLength)
  16-bit signed short signal right shift with constant.
- NppStatus nppsRShiftC_32u (const Npp32u ∗pSrc, int nValue, Npp32u ∗pDst, int nLength)
  32-bit unsigned integer signal right shift with constant.
- NppStatus nppsRShiftC_32s (const Npp32s ∗pSrc, int nValue, Npp32s ∗pDst, int nLength)
  32-bit signed integer signal right shift with constant.
- NppStatus nppsRShiftC_8u_I (int nValue, Npp8u ∗pSrcDst, int nLength)
  8-bit unsigned char in place signal right shift with constant.
- NppStatus nppsRShiftC_16u_I (int nValue, Npp16u ∗pSrcDst, int nLength)
  16-bit unsigned short in place signal right shift with constant.
- NppStatus nppsRShiftC_16s_I (int nValue, Npp16s ∗pSrcDst, int nLength)
  16-bit signed short in place signal right shift with constant.
- NppStatus nppsRShiftC_32u_I (int nValue, Npp32u ∗pSrcDst, int nLength)
  32-bit unsigned signed integer in place signal right shift with constant.
- NppStatus nppsRShiftC_32s_I (int nValue, Npp32s ∗pSrcDst, int nLength)
  32-bit signed signed integer in place signal right shift with constant.

Statistical Functions

Functions that provide global signal statistics like: average, standard deviation, minimum, etc.

- NppStatus nppsReductionGetBufferSize_8u (int nLength, int ∗hpBufferSize)

Device-buffer size (in bytes) for 8u reductions.

- NppStatus nppsReductionGetBufferSize_16s (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for 16s reductions.

- NppStatus nppsReductionGetBufferSize_16u (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for 16u reductions.

- NppStatus nppsReductionGetBufferSize_16s_Sfs (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for 16s reductions with integer-results scaling.

- NppStatus nppsReductionGetBufferSize_16sc (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for 16sc reductions.

- NppStatus nppsReductionGetBufferSize_16sc_Sfs (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for 16sc reductions with integer-results scaling.

- NppStatus nppsReductionGetBufferSize_32s (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for 32s reductions.

- NppStatus nppsReductionGetBufferSize_32u (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for 32u reductions.

- NppStatus nppsReductionGetBufferSize_32s_Sfs (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for 32s reductions with integer-results scaling.

- NppStatus nppsReductionGetBufferSize_32sc (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for 32sc reductions.

- NppStatus nppsReductionGetBufferSize_32f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for 32f reductions.

- NppStatus nppsReductionGetBufferSize_32fc (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for 32fc reductions.

- NppStatus nppsReductionGetBufferSize_64s (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for 64s reductions.

- NppStatus nppsReductionGetBufferSize_64f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for 64f reductions.

- NppStatus nppsReductionGetBufferSize_64fc (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for 64fc reductions.

- NppStatus nppsSum_32f (const Npp32f ∗pSrc, int nLength, Npp32f ∗pSum, NppHintAlgorithm eHint, Npp8u ∗pDeviceBuffer)

  32-bit float vector sum method

- NppStatus nppsSum_32fc (const Npp32fc ∗pSrc, int nLength, Npp32fc ∗pSum, NppHintAlgorithm eHint, Npp8u ∗pDeviceBuffer)

32-bit float complex vector sum method

- NppStatus nppsSum_64f (const Npp64f ∗pSrc, int nLength, Npp64f ∗pSum, Npp8u ∗pDeviceBuffer)

  64-bit double vector sum method

- NppStatus nppsSum_64fc (const Npp64fc ∗pSrc, int nLength, Npp64fc ∗pSum, Npp8u ∗pDeviceBuffer)

  64-bit double complex vector sum method

- NppStatus nppsSum_16s_Sfs (const Npp16s ∗pSrc, int nLength, Npp16s ∗pSum, int nScaleFactor, Npp8u ∗pDeviceBuffer)

  16-bit short vector sum with integer scaling method

- NppStatus nppsSum_32s_Sfs (const Npp32s ∗pSrc, int nLength, Npp32s ∗pSum, int nScaleFactor, Npp8u ∗pDeviceBuffer)

  32-bit integer vector sum with integer scaling method

- NppStatus nppsSum_16sc_Sfs (const Npp16sc ∗pSrc, int nLength, Npp16sc ∗pSum, int nScaleFactor, Npp8u ∗pDeviceBuffer)

  16-bit short complex vector sum with integer scaling method

- NppStatus nppsSum_16sc32sc_Sfs (const Npp16sc ∗pSrc, int nLength, Npp32sc ∗pSum, int nScaleFactor, Npp8u ∗pDeviceBuffer)

  16-bit short complex vector sum (32bit int complex) with integer scaling method

- NppStatus nppsSum_16s32s_Sfs (const Npp16s ∗pSrc, int nLength, Npp32s ∗pSum, int nScaleFactor, Npp8u ∗pDeviceBuffer)

  16-bit integer vector sum (32bit) with integer scaling method

- NppStatus nppsMax_16s (const Npp16s ∗pSrc, int nLength, Npp16s ∗pMax, Npp8u ∗pDeviceBuffer)

  16-bit integer vector max method

- NppStatus nppsMax_32s (const Npp32s ∗pSrc, int nLength, Npp32s ∗pMax, Npp8u ∗pDeviceBuffer)

  32-bit integer vector max method

- NppStatus nppsMax_32f (const Npp32f ∗pSrc, int nLength, Npp32f ∗pMax, Npp8u ∗pDeviceBuffer)

  32-bit float vector max method

- NppStatus nppsMax_64f (const Npp64f ∗pSrc, int nLength, Npp64f ∗pMax, Npp8u ∗pDeviceBuffer)

  64-bit float vector max method

- NppStatus nppsMin_16s (const Npp16s ∗pSrc, int nLength, Npp16s ∗pMin, Npp8u ∗pDeviceBuffer)

  16-bit integer vector min method

- NppStatus nppsMin_32s (const Npp32s ∗pSrc, int nLength, Npp32s ∗pMin, Npp8u ∗pDeviceBuffer)

  32-bit integer vector min method

- NppStatus nppsMin_32f (const Npp32f ∗pSrc, int nLength, Npp32f ∗pMin, Npp8u ∗pDeviceBuffer)

  32-bit integer vector min method

- NppStatus nppsMin_64f (const Npp64f ∗pSrc, int nLength, Npp64f ∗pMin, Npp8u ∗pDeviceBuffer)

  64-bit integer vector min method

- NppStatus nppsMinMaxGetBufferSize_8u (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsMinMax_8u.

- NppStatus nppsMinMaxGetBufferSize_16s (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsMinMax_16s.

- NppStatus nppsMinMaxGetBufferSize_16u (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsMinMax_16u.

- NppStatus nppsMinMaxGetBufferSize_32s (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsMinMax_32s.

- NppStatus nppsMinMaxGetBufferSize_32u (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsMinMax_32u.

- NppStatus nppsMinMaxGetBufferSize_32f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsMinMax_32f.

- NppStatus nppsMinMaxGetBufferSize_64f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsMinMax_64f.

- NppStatus nppsMinMax_8u (const Npp8u ∗pSrc, int nLength, Npp8u ∗pMin, Npp8u ∗pMax, Npp8u ∗pDeviceBuffer)

  8-bit char vector min and max method

- NppStatus nppsMinMax_16s (const Npp16s ∗pSrc, int nLength, Npp16s ∗pMin, Npp16s ∗pMax, Npp8u ∗pDeviceBuffer)

  16-bit signed short vector min and max method

- NppStatus nppsMinMax_16u (const Npp16u ∗pSrc, int nLength, Npp16u ∗pMin, Npp16u ∗pMax, Npp8u ∗pDeviceBuffer)

  16-bit unsigned short vector min and max method

- NppStatus nppsMinMax_32u (const Npp32u ∗pSrc, int nLength, Npp32u ∗pMin, Npp32u ∗pMax, Npp8u ∗pDeviceBuffer)

  32-bit unsigned int vector min and max method

- NppStatus nppsMinMax_32s (const Npp32s ∗pSrc, int nLength, Npp32s ∗pMin, Npp32s ∗pMax, Npp8u ∗pDeviceBuffer)

32-bit signed int vector min and max method

- NppStatus nppsMinMax_32f (const Npp32f ∗pSrc, int nLength, Npp32f ∗pMin, Npp32f ∗pMax, Npp8u ∗pDeviceBuffer)

  32-bit float vector min and max method

- NppStatus nppsMinMax_64f (const Npp64f ∗pSrc, int nLength, Npp64f ∗pMin, Npp64f ∗pMax, Npp8u ∗pDeviceBuffer)

  64-bit double vector min and max method

Filtering Functions

Functions that provide functionality of generating output signal based on the input signal like signal integral, etc.

- NppStatus nppsIntegralGetBufferSize_32s (int nLength, int ∗hpBufferSize)
- NppStatus nppsIntegral_32s (const Npp32s ∗pSrc, Npp32s ∗pDst, int nLength, Npp8u ∗pDeviceBuffer)

7.5.1 Function Documentation

7.5.1.1 NppStatus npps10Log10_32s_ISfs (Npp32s ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer signal 10 times base 10 logarithm, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.2 NppStatus npps10Log10_32s_Sfs (const Npp32s ∗ *pSrc*, Npp32s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer signal 10 times base 10 logarithm, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.3 NppStatus nppsAbs_16s (const Npp16s ∗ *pSrc*, Npp16s ∗ *pDst*, int *nLength*)

16-bit signed short signal absolute value.

Parameters:

pSrc Source Signal Pointer.
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.4 NppStatus nppsAbs_16s_I (Npp16s ∗ *pSrcDst*, int *nLength*)

16-bit signed short signal absolute value.

Parameters:

pSrcDst In-Place Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.5 NppStatus nppsAbs_32f (const Npp32f ∗ *pSrc*, Npp32f ∗ *pDst*, int *nLength*)

32-bit floating point signal absolute value.

Parameters:

pSrc Source Signal Pointer.
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.6 NppStatus nppsAbs_32f_I (Npp32f ∗ *pSrcDst*, int *nLength*)

32-bit floating point signal absolute value.

Parameters:

pSrcDst In-Place Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.7 NppStatus nppsAbs_32s (const Npp32s ∗ *pSrc*, Npp32s ∗ *pDst*, int *nLength*)

32-bit signed integer signal absolute value.

Parameters:

pSrc Source Signal Pointer.
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.8 NppStatus nppsAbs_32s_I (Npp32s ∗ *pSrcDst*, int *nLength*)

32-bit signed integer signal absolute value.

Parameters:

pSrcDst In-Place Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.9 NppStatus nppsAbs_64f (const Npp64f ∗ *pSrc*, Npp64f ∗ *pDst*, int *nLength*)

64-bit floating point signal absolute value.

Parameters:

pSrc Source Signal Pointer.
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.10 NppStatus nppsAbs_64f_I (Npp64f ∗ *pSrcDst*, int *nLength*)

64-bit floating point signal absolute value.

Parameters:

pSrcDst In-Place Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.11 NppStatus nppsAdd_16s (const Npp16s ∗ *pSrc1*, const Npp16s ∗ *pSrc2*, Npp16s ∗ *pDst*, int *nLength*)

16-bit signed short signal add signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be added to signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.12 NppStatus nppsAdd_16s32f (const Npp16s ∗ *pSrc1*, const Npp16s ∗ *pSrc2*, Npp32f ∗ *pDst*, int *nLength*)

16-bit signed short signal add signal with 32-bit floating point result, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be added to signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.13 NppStatus nppsAdd_16s32s_I (const Npp16s ∗ *pSrc*, Npp32s ∗ *pSrcDst*, int *nLength*)

16/32-bit signed short in place signal add signal with 32-bit signed integer results, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be added to signal1 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.14 NppStatus nppsAdd_16s_I (const Npp16s ∗ *pSrc*, Npp16s ∗ *pSrcDst*, int *nLength*)

16-bit signed short in place signal add signal, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be added to signal1 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.15 NppStatus nppsAdd_16s_ISfs (const Npp16s ∗ *pSrc*, Npp16s ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short in place signal add signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be added to signal1 elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.16 NppStatus nppsAdd_16s_Sfs (const Npp16s ∗ *pSrc1*, const Npp16s ∗ *pSrc2*, Npp16s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short add signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal2 elements to be added to signal1 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.17 NppStatus nppsAdd_16sc_ISfs (const Npp16sc ∗ *pSrc*, Npp16sc ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit complex signed short in place signal add signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be added to signal1 elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.18 NppStatus nppsAdd_16sc_Sfs (const Npp16sc ∗ *pSrc1*, const Npp16sc ∗ *pSrc2*, Npp16sc ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit signed complex short add signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal2 elements to be added to signal1 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.19 NppStatus nppsAdd_16u (const Npp16u ∗ *pSrc1*, const Npp16u ∗ *pSrc2*, Npp16u ∗ *pDst*, int *nLength*)

16-bit unsigned short signal add signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be added to signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.20 NppStatus nppsAdd_16u_ISfs (const Npp16u ∗ *pSrc*, Npp16u ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short in place signal add signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be added to signal1 elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.21 NppStatus nppsAdd_16u_Sfs (const Npp16u ∗ *pSrc1*, const Npp16u ∗ *pSrc2*, Npp16u ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short add signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal2 elements to be added to signal1 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.22 NppStatus nppsAdd_32f (const Npp32f ∗ *pSrc1*, const Npp32f ∗ *pSrc2*, Npp32f ∗ *pDst*, int *nLength*)

32-bit floating point signal add signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be added to signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.23 NppStatus nppsAdd_32f_I (const Npp32f ∗ *pSrc*, Npp32f ∗ *pSrcDst*, int *nLength*)

32-bit floating point in place signal add signal, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be added to signal1 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.24 NppStatus nppsAdd_32fc (const Npp32fc ∗ *pSrc1*, const Npp32fc ∗ *pSrc2*, Npp32fc ∗ *pDst*, int *nLength*)

32-bit complex floating point signal add signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be added to signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.25 NppStatus nppsAdd_32fc_I (const Npp32fc ∗ *pSrc*, Npp32fc ∗ *pSrcDst*, int *nLength*)

32-bit complex floating point in place signal add signal, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be added to signal1 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.26 NppStatus nppsAdd_32s_ISfs (const Npp32s ∗ *pSrc*, Npp32s ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer in place signal add signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be added to signal1 elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.27 NppStatus nppsAdd_32s_Sfs (const Npp32s ∗ *pSrc1*, const Npp32s ∗ *pSrc2*, Npp32s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer add signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal2 elements to be added to signal1 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.28 NppStatus nppsAdd_32sc_ISfs (const Npp32sc ∗ *pSrc*, Npp32sc ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

32-bit complex signed integer in place signal add signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be added to signal1 elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.29 NppStatus nppsAdd_32sc_Sfs (const Npp32sc ∗ *pSrc1*, const Npp32sc ∗ *pSrc2*, Npp32sc ∗ *pDst*, int *nLength*, int *nScaleFactor*)

32-bit signed complex integer add signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal2 elements to be added to signal1 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.30 NppStatus nppsAdd_32u (const Npp32u ∗ *pSrc1*, const Npp32u ∗ *pSrc2*, Npp32u ∗ *pDst*, int *nLength*)

32-bit unsigned int signal add signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be added to signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.31 NppStatus nppsAdd_64f (const Npp64f ∗ *pSrc1*, const Npp64f ∗ *pSrc2*, Npp64f ∗ *pDst*, int *nLength*)

64-bit floating point signal add signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be added to signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.32 NppStatus nppsAdd_64f_I (const Npp64f ∗ *pSrc*, Npp64f ∗ *pSrcDst*, int *nLength*)

64-bit floating point in place signal add signal, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be added to signal1 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.33 NppStatus nppsAdd_64fc (const Npp64fc ∗ *pSrc1*, const Npp64fc ∗ *pSrc2*, Npp64fc ∗ *pDst*, int *nLength*)

64-bit complex floating point signal add signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be added to signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.34 NppStatus nppsAdd_64fc_I (const Npp64fc ∗ *pSrc*, Npp64fc ∗ *pSrcDst*, int *nLength*)

64-bit complex floating point in place signal add signal, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be added to signal1 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.35 NppStatus nppsAdd_64s_Sfs (const Npp64s ∗ *pSrc1*, const Npp64s ∗ *pSrc2*, Npp64s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

64-bit signed integer add signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal2 elements to be added to signal1 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.36 NppStatus nppsAdd_8u16u (const Npp8u ∗ *pSrc1*, const Npp8u ∗ *pSrc2*, Npp16u ∗ *pDst*, int *nLength*)

8-bit unsigned char signal add signal with 16-bit unsigned result, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be added to signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.37 NppStatus nppsAdd_8u_ISfs (const Npp8u ∗ *pSrc*, Npp8u ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned char in place signal add signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be added to signal1 elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.38 NppStatus nppsAdd_8u_Sfs (const Npp8u ∗ *pSrc1*, const Npp8u ∗ *pSrc2*, Npp8u ∗ *pDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned char add signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal2 elements to be added to signal1 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.39 NppStatus nppsAddC_16s_ISfs (Npp16s *nValue*, Npp16s ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short in place signal add constant, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be added to each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.40 NppStatus nppsAddC_16s_Sfs (const Npp16s ∗ *pSrc*, Npp16s *nValue*, Npp16s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short signal add constant, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be added to each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.41 NppStatus nppsAddC_16sc_ISfs (Npp16sc *nValue*, Npp16sc ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit integer complex number (16 bit real, 16 bit imaginary) signal add constant, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be added to each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.42 NppStatus nppsAddC_16sc_Sfs (const Npp16sc ∗ *pSrc*, Npp16sc *nValue*, Npp16sc ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit integer complex number (16 bit real, 16 bit imaginary) signal add constant, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be added to each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.43 NppStatus nppsAddC_16u_ISfs (Npp16u *nValue*, Npp16u ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short in place signal add constant, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be added to each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.44 NppStatus nppsAddC_16u_Sfs (const Npp16u ∗ *pSrc*, Npp16u *nValue*, Npp16u ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short vector add constant, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be added to each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.45 NppStatus nppsAddC_32f (const Npp32f ∗ *pSrc*, Npp32f *nValue*, Npp32f ∗ *pDst*, int *nLength*)

32-bit floating point signal add constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be added to each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.46 NppStatus nppsAddC_32f_I (Npp32f *nValue*, Npp32f ∗ *pSrcDst*, int *nLength*)

32-bit floating point in place signal add constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be added to each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.47 NppStatus nppsAddC_32fc (const Npp32fc * *pSrc*, Npp32fc *nValue*, Npp32fc * *pDst*, int *nLength*)

32-bit floating point complex number (32 bit real, 32 bit imaginary) signal add constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be added to each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.48 NppStatus nppsAddC_32fc_I (Npp32fc *nValue*, Npp32fc * *pSrcDst*, int *nLength*)

32-bit floating point complex number (32 bit real, 32 bit imaginary) in place signal add constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be added to each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.49 NppStatus nppsAddC_32s_ISfs (Npp32s *nValue*, Npp32s * *pSrcDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer in place signal add constant and scale.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be added to each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.50 NppStatus nppsAddC_32s_Sfs (const Npp32s ∗ *pSrc*, Npp32s *nValue*, Npp32s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integersignal add constant and scale.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be added to each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.51 NppStatus nppsAddC_32sc_ISfs (Npp32sc *nValue*, Npp32sc ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

32-bit integer complex number (32 bit real, 32 bit imaginary) in place signal add constant and scale.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be added to each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.52 NppStatus nppsAddC_32sc_Sfs (const Npp32sc ∗ *pSrc*, Npp32sc *nValue*, Npp32sc ∗ *pDst*, int *nLength*, int *nScaleFactor*)

32-bit integer complex number (32 bit real, 32 bit imaginary) signal add constant and scale.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be added to each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.53 NppStatus nppsAddC_64f (const Npp64f ∗ *pSrc*, Npp64f *nValue*, Npp64f ∗ *pDst*, int *nLength*)

64-bit floating pointsignal add constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be added to each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.54 NppStatus nppsAddC_64f_I (Npp64f *nValue*, Npp64f ∗ *pSrcDst*, int *nLength*)

64-bit floating point, in place signal add constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be added to each vector element

nLength Length of the vectors, number of items.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.55 NppStatus nppsAddC_64fc (const Npp64fc ∗ *pSrc*, Npp64fc *nValue*, Npp64fc ∗ *pDst*, int *nLength*)

64-bit floating point complex number (64 bit real, 64 bit imaginary) signal add constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be added to each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.56 NppStatus nppsAddC_64fc_I (Npp64fc *nValue*, Npp64fc ∗ *pSrcDst*, int *nLength*)

64-bit floating point complex number (64 bit real, 64 bit imaginary) in place signal add constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be added to each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.57 NppStatus nppsAddC_8u_ISfs (Npp8u *nValue*, Npp8u ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned char in place signal add constant, scale, then clamp to saturated value

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be added to each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.58 NppStatus nppsAddC_8u_Sfs (const Npp8u ∗ *pSrc*, Npp8u *nValue*, Npp8u ∗ *pDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned charvector add constant, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be added to each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.59 NppStatus nppsAddProduct_16s32s_Sfs (const Npp16s * *pSrc1*, const Npp16s * *pSrc2*, Npp32s * *pDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short signal add product of source signal1 times source signal2 to 32-bit signed integer destination signal, with scaling, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

pDst Destination Signal Pointer. product of source1 and source2 signal elements to be added to destination elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.60 NppStatus nppsAddProduct_16s_Sfs (const Npp16s * *pSrc1*, const Npp16s * *pSrc2*, Npp16s * *pDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short signal add product of source signal1 times source signal2 to destination signal, with scaling, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

pDst Destination Signal Pointer. product of source1 and source2 signal elements to be added to destination elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.61 NppStatus nppsAddProduct_32f (const Npp32f * *pSrc1*, const Npp32f * *pSrc2*, Npp32f * *pDst*, int *nLength*)

32-bit floating point signal add product of source signal times destination signal to destination signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

pDst Destination Signal Pointer. product of source1 and source2 signal elements to be added to destination elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.62 NppStatus nppsAddProduct_32fc (const Npp32fc ∗ *pSrc1*, const Npp32fc ∗ *pSrc2*, Npp32fc ∗ *pDst*, int *nLength*)

32-bit complex floating point signal add product of source signal times destination signal to destination signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

pDst Destination Signal Pointer. product of source1 and source2 signal elements to be added to destination elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.63 NppStatus nppsAddProduct_32s_Sfs (const Npp32s ∗ *pSrc1*, const Npp32s ∗ *pSrc2*, Npp32s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

32-bit signed short signal add product of source signal1 times source signal2 to destination signal, with scaling, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

pDst Destination Signal Pointer. product of source1 and source2 signal elements to be added to destination elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.64 NppStatus nppsAddProduct_64f (const Npp64f ∗ *pSrc1*, const Npp64f ∗ *pSrc2*, Npp64f ∗ *pDst*, int *nLength*)

64-bit floating point signal add product of source signal times destination signal to destination signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

pDst Destination Signal Pointer. product of source1 and source2 signal elements to be added to destination elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.65 NppStatus nppsAddProduct_64fc (const Npp64fc * *pSrc1*, const Npp64fc * *pSrc2*, Npp64fc * *pDst*, int *nLength*)

64-bit complex floating point signal add product of source signal times destination signal to destination signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

pDst Destination Signal Pointer. product of source1 and source2 signal elements to be added to destination elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.66 NppStatus nppsAddProductC_16s_ISfs (Npp16s *nValue*, Npp16s * *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short in place signal add product of signal times constant to destination signal, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be multiplied by each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.67 NppStatus nppsAddProductC_16s_Sfs (const Npp16s ∗ *pSrc*, Npp16s *nValue*, Npp16s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short signal add product of signal times constant to destination signal, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be multiplied by each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.68 NppStatus nppsAddProductC_16sc_ISfs (Npp16sc *nValue*, Npp16sc ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit integer complex number (16 bit real, 16 bit imaginary)signal add product of signal times constant to destination signal, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be multiplied by each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.69 NppStatus nppsAddProductC_16sc_Sfs (const Npp16sc ∗ *pSrc*, Npp16sc *nValue*, Npp16sc ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit integer complex number (16 bit real, 16 bit imaginary)signal add product of signal times constant to destination signal, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be multiplied by each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.70 NppStatus nppsAddProductC_16u_ISfs (Npp16u *nValue*, Npp16u ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short in place signal add product of signal times constant to destination signal, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be multiplied by each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.71 NppStatus nppsAddProductC_16u_Sfs (const Npp16u ∗ *pSrc*, Npp16u *nValue*, Npp16u ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short add product of signal times constant to destination signal, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be multiplied by each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.72 NppStatus nppsAddProductC_32f (const Npp32f ∗ *pSrc*, Npp32f *nValue*, Npp32f ∗ *pDst*, int *nLength*)

32-bit floating point signal add product of signal times constant to destination signal.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be multiplied by each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.73 NppStatus nppsAddProductC_32f_I (Npp32f *nValue*, Npp32f ∗ *pSrcDst*, int *nLength*)

32-bit floating point in place signal add product of signal times constant to destination signal.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be multiplied by each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.74 NppStatus nppsAddProductC_32s_ISfs (Npp32s *nValue*, Npp32s ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer in place signal add product of signal times constant to destination signal and scale.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be multiplied by each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.75 NppStatus nppsAddProductC_32s_Sfs (const Npp32s ∗ *pSrc*, Npp32s *nValue*, Npp32s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer signal add product of signal times constant to destination signal and scale.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be multiplied by each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.76 NppStatus nppsAddProductC_32sc_ISfs (Npp32sc *nValue*, Npp32sc * *pSrcDst*, int *nLength*, int *nScaleFactor*)

32-bit integer complex number (32 bit real, 32 bit imaginary) in place signal add product of signal times constant to destination signal and scale.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be multiplied by each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.77 NppStatus nppsAddProductC_32sc_Sfs (const Npp32sc * *pSrc*, Npp32sc *nValue*, Npp32sc * *pDst*, int *nLength*, int *nScaleFactor*)

32-bit integer complex number (32 bit real, 32 bit imaginary)signal add product of signal times constant to destination signal and scale.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be multiplied by each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.78 NppStatus nppsAddProductC_8u_ISfs (Npp8u *nValue*, Npp8u * *pSrcDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned char in place signal add product of signal times constant to destination signal, scale, then clamp to saturated value

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be multiplied by each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.79 NppStatus nppsAddProductC_8u_Sfs (const Npp8u ∗ *pSrc*, Npp8u *nValue*, Npp8u ∗ *pDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned char add product of signal times constant to destination signal, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be multiplied by each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.80 NppStatus nppsAnd_16u (const Npp16u ∗ *pSrc1*, const Npp16u ∗ *pSrc2*, Npp16u ∗ *pDst*, int *nLength*)

16-bit unsigned short signal and with signal.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be anded with signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.81 NppStatus nppsAnd_16u_I (const Npp16u ∗ *pSrc*, Npp16u ∗ *pSrcDst*, int *nLength*)

16-bit unsigned short in place signal and with signal.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be anded with signal1 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.82 NppStatus nppsAnd_32u (const Npp32u ∗ *pSrc1*, const Npp32u ∗ *pSrc2*, Npp32u ∗ *pDst*, int *nLength*)

32-bit unsigned integer signal and with signal.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be anded with signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.83 NppStatus nppsAnd_32u_I (const Npp32u ∗ *pSrc*, Npp32u ∗ *pSrcDst*, int *nLength*)

32-bit unsigned integer in place signal and with signal.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be anded with signal1 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.84 NppStatus nppsAnd_8u (const Npp8u ∗ *pSrc1*, const Npp8u ∗ *pSrc2*, Npp8u ∗ *pDst*, int *nLength*)

8-bit unsigned char signal and with signal.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be anded with signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.85 NppStatus nppsAnd_8u_I (const Npp8u ∗ *pSrc*, Npp8u ∗ *pSrcDst*, int *nLength*)

8-bit unsigned char in place signal and with signal.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be anded with signal1 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.86 NppStatus nppsAndC_16u (const Npp16u ∗ *pSrc*, Npp16u *nValue*, Npp16u ∗ *pDst*, int *nLength*)

16-bit unsigned short signal and with constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be anded with each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.87 NppStatus nppsAndC_16u_I (Npp16u *nValue*, Npp16u ∗ *pSrcDst*, int *nLength*)

16-bit unsigned short in place signal and with constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be anded with each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.88 NppStatus nppsAndC_32u (const Npp32u ∗ *pSrc*, Npp32u *nValue*, Npp32u ∗ *pDst*, int *nLength*)

32-bit unsigned integer signal and with constant.

Parameters:

pSrc Source Signal Pointer.
nValue Constant value to be anded with each vector element
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.89 NppStatus nppsAndC_32u_I (Npp32u *nValue*, Npp32u * *pSrcDst*, int *nLength*)

32-bit unsigned signed integer in place signal and with constant.

Parameters:

pSrcDst In-Place Signal Pointer.
nValue Constant value to be anded with each vector element
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.90 NppStatus nppsAndC_8u (const Npp8u * *pSrc*, Npp8u *nValue*, Npp8u * *pDst*, int *nLength*)

8-bit unsigned char signal and with constant.

Parameters:

pSrc Source Signal Pointer.
nValue Constant value to be anded with each vector element
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.91 NppStatus nppsAndC_8u_I (Npp8u *nValue*, Npp8u * *pSrcDst*, int *nLength*)

8-bit unsigned char in place signal and with constant.

Parameters:

pSrcDst In-Place Signal Pointer.
nValue Constant value to be anded with each vector element
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.92 NppStatus nppsArctan_32f (const Npp32f $*$ *pSrc*, Npp32f $*$ *pDst*, int *nLength*)

32-bit floating point signal inverse tangent.

Parameters:

pSrc Source Signal Pointer.
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.93 NppStatus nppsArctan_32f_I (Npp32f $*$ *pSrcDst*, int *nLength*)

32-bit floating point signal inverse tangent.

Parameters:

pSrcDst In-Place Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.94 NppStatus nppsArctan_64f (const Npp64f $*$ *pSrc*, Npp64f $*$ *pDst*, int *nLength*)

64-bit floating point signal inverse tangent.

Parameters:

pSrc Source Signal Pointer.
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.95 NppStatus nppsArctan_64f_I (Npp64f $*$ *pSrcDst*, int *nLength*)

64-bit floating point signal inverse tangent.

Parameters:

pSrcDst In-Place Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.96 NppStatus nppsCauchy_32f_I (Npp32f $*$ *pSrcDst*, int *nLength*, Npp32f *nParam*)

32-bit floating point signal Cauchy error calculation.

Parameters:

pSrcDst In-Place Signal Pointer.
nLength Signal Length.
nParam constant used in Cauchy formula

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.97 NppStatus nppsCauchyD_32f_I (Npp32f $*$ *pSrcDst*, int *nLength*, Npp32f *nParam*)

32-bit floating point signal Cauchy first derivative.

Parameters:

pSrcDst In-Place Signal Pointer.
nLength Signal Length.
nParam constant used in Cauchy formula

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.98 NppStatus nppsCauchyDD2_32f_I (Npp32f $*$ *pSrcDst*, Npp32f $*$ *pD2FVal*, int *nLength*, Npp32f *nParam*)

32-bit floating point signal Cauchy first and second derivatives.

Parameters:

pSrcDst In-Place Signal Pointer.
pD2FVal Source Signal Pointer. This signal contains the second derivative of the source signal.
nLength Signal Length.
nParam constant used in Cauchy formula

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.99 NppStatus nppsCopy_16s (const Npp16s $*$ *pSrc*, Npp16s $*$ *pDst*, int *len*)

16-bit signed short, vector copy method.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

len Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.100 NppStatus nppsCopy_16sc (const Npp16sc * *pSrc*, Npp16sc * *pDst*, int *len*)

16-bit complex short, vector copy method.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

len Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.101 NppStatus nppsCopy_32f (const Npp32f * *pSrc*, Npp32f * *pDst*, int *len*)

32-bit float, vector copy method.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

len Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.102 NppStatus nppsCopy_32fc (const Npp32fc * *pSrc*, Npp32fc * *pDst*, int *len*)

32-bit complex float, vector copy method.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

len Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.103 NppStatus nppsCopy_32s (const Npp32s $*$ *pSrc*, Npp32s $*$ *pDst*, int *nLength*)

32-bit signed integer, vector copy method.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.104 NppStatus nppsCopy_32sc (const Npp32sc $*$ *pSrc*, Npp32sc $*$ *pDst*, int *len*)

32-bit complex signed integer, vector copy method.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

len Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.105 NppStatus nppsCopy_64fc (const Npp64fc $*$ *pSrc*, Npp64fc $*$ *pDst*, int *len*)

64-bit complex double, vector copy method.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

len Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.106 NppStatus nppsCopy_64s (const Npp64s $*$ *pSrc*, Npp64s $*$ *pDst*, int *len*)

64-bit signed integer, vector copy method.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

len Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.107 NppStatus nppsCopy_64sc (const Npp64sc ∗ *pSrc*, Npp64sc ∗ *pDst*, int *len*)

64-bit complex signed integer, vector copy method.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

len Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.108 NppStatus nppsCopy_8u (const Npp8u ∗ *pSrc*, Npp8u ∗ *pDst*, int *len*)

8-bit unsigned char, vector copy method

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

len Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.109 NppStatus nppsCubrt_32f (const Npp32f ∗ *pSrc*, Npp32f ∗ *pDst*, int *nLength*)

32-bit floating point signal cube root.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.110 NppStatus nppsCubrt_32s16s_Sfs (const Npp32s * *pSrc*, Npp16s * *pDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer signal cube root, scale, then clamp to 16-bit signed integer saturated value.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.111 NppStatus nppsDiv_16s_ISfs (const Npp16s * *pSrc*, Npp16s * *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short in place signal divide signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 divisor elements to be divided into signal2 dividend elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.112 NppStatus nppsDiv_16s_Sfs (const Npp16s * *pSrc1*, const Npp16s * *pSrc2*, Npp16s * *pDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short signal divide signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal1 divisor elements to be divided into signal2 dividend elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.113 NppStatus nppsDiv_16sc_ISfs (const Npp16sc ∗ *pSrc*, Npp16sc ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit complex signed short in place signal divide signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 divisor elements to be divided into signal2 dividend elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.114 NppStatus nppsDiv_16sc_Sfs (const Npp16sc ∗ *pSrc1*, const Npp16sc ∗ *pSrc2*, Npp16sc ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit signed complex short signal divide signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal1 divisor elements to be divided into signal2 dividend elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.115 NppStatus nppsDiv_16u_ISfs (const Npp16u ∗ *pSrc*, Npp16u ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short in place signal divide signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 divisor elements to be divided into signal2 dividend elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.116 NppStatus nppsDiv_16u_Sfs (const Npp16u ∗ *pSrc1*, const Npp16u ∗ *pSrc2*, Npp16u ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short signal divide signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal1 divisor elements to be divided into signal2 dividend elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.117 NppStatus nppsDiv_32f (const Npp32f ∗ *pSrc1*, const Npp32f ∗ *pSrc2*, Npp32f ∗ *pDst*, int *nLength*)

32-bit floating point signal divide signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal1 divisor elements to be divided into signal2 dividend elements.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.118 NppStatus nppsDiv_32f_I (const Npp32f ∗ *pSrc*, Npp32f ∗ *pSrcDst*, int *nLength*)

32-bit floating point in place signal divide signal, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 divisor elements to be divided into signal2 dividend elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.119 NppStatus nppsDiv_32fc (const Npp32fc ∗ *pSrc1*, const Npp32fc ∗ *pSrc2*, Npp32fc ∗ *pDst*, int *nLength*)

32-bit complex floating point signal divide signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal1 divisor elements to be divided into signal2 dividend elements.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.120 NppStatus nppsDiv_32fc_I (const Npp32fc ∗ *pSrc*, Npp32fc ∗ *pSrcDst*, int *nLength*)

32-bit complex floating point in place signal divide signal, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 divisor elements to be divided into signal2 dividend elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.121 NppStatus nppsDiv_32s16s_Sfs (const Npp16s ∗ *pSrc1*, const Npp32s ∗ *pSrc2*, Npp16s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer signal divided by 16-bit signed short signal, scale, then clamp to 16-bit signed short saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal1 divisor elements to be divided into signal2 dividend elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.122 NppStatus nppsDiv_32s_ISfs (const Npp32s * *pSrc*, Npp32s * *pSrcDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer in place signal divide signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 divisor elements to be divided into signal2 dividend elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.123 NppStatus nppsDiv_32s_Sfs (const Npp32s * *pSrc1*, const Npp32s * *pSrc2*, Npp32s * *pDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer signal divide signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal1 divisor elements to be divided into signal2 dividend elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.124 NppStatus nppsDiv_64f (const Npp64f * *pSrc1*, const Npp64f * *pSrc2*, Npp64f * *pDst*, int *nLength*)

64-bit floating point signal divide signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal1 divisor elements to be divided into signal2 dividend elements.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.125 NppStatus nppsDiv_64f_I (const Npp64f ∗ *pSrc*, Npp64f ∗ *pSrcDst*, int *nLength*)

64-bit floating point in place signal divide signal, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 divisor elements to be divided into signal2 dividend elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.126 NppStatus nppsDiv_64fc (const Npp64fc ∗ *pSrc1*, const Npp64fc ∗ *pSrc2*, Npp64fc ∗ *pDst*, int *nLength*)

64-bit complex floating point signal divide signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal1 divisor elements to be divided into signal2 dividend elements.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.127 NppStatus nppsDiv_64fc_I (const Npp64fc ∗ *pSrc*, Npp64fc ∗ *pSrcDst*, int *nLength*)

64-bit complex floating point in place signal divide signal, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 divisor elements to be divided into signal2 dividend elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.128 NppStatus nppsDiv_8u_ISfs (const Npp8u ∗ *pSrc*, Npp8u ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned char in place signal divide signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 divisor elements to be divided into signal2 dividend elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.129 NppStatus nppsDiv_8u_Sfs (const Npp8u ∗ *pSrc1*, const Npp8u ∗ *pSrc2*, Npp8u ∗ *pDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned char signal divide signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal1 divisor elements to be divided into signal2 dividend elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.130 NppStatus nppsDiv_Round_16s_ISfs (const Npp16s ∗ *pSrc*, Npp16s ∗ *pSrcDst*, int *nLength*, NppRoundMode *nRndMode*, int *nScaleFactor*)

16-bit signed short in place signal divide signal, with scaling, rounding then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 divisor elements to be divided into signal2 dividend elements

nLength Signal Length.

nRndMode various rounding modes.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.131 NppStatus nppsDiv_Round_16s_Sfs (const Npp16s ∗ *pSrc1*, const Npp16s ∗ *pSrc2*, Npp16s ∗ *pDst*, int *nLength*, NppRoundMode *nRndMode*, int *nScaleFactor*)

16-bit signed short signal divide signal, scale, round, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal1 divisor elements to be divided into signal2 dividend elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nRndMode various rounding modes.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.132 NppStatus nppsDiv_Round_16u_ISfs (const Npp16u ∗ *pSrc*, Npp16u ∗ *pSrcDst*, int *nLength*, NppRoundMode *nRndMode*, int *nScaleFactor*)

16-bit unsigned short in place signal divide signal, with scaling, rounding then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 divisor elements to be divided into signal2 dividend elements

nLength Signal Length.

nRndMode various rounding modes.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.133 NppStatus nppsDiv_Round_16u_Sfs (const Npp16u ∗ *pSrc1*, const Npp16u ∗ *pSrc2*, Npp16u ∗ *pDst*, int *nLength*, NppRoundMode *nRndMode*, int *nScaleFactor*)

16-bit unsigned short signal divide signal, scale, round, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal1 divisor elements to be divided into signal2 dividend elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nRndMode various rounding modes.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.134 NppStatus nppsDiv_Round_8u_ISfs (const Npp8u ∗ *pSrc*, Npp8u ∗ *pSrcDst*, int *nLength*, NppRoundMode *nRndMode*, int *nScaleFactor*)

8-bit unsigned char in place signal divide signal, with scaling, rounding then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 divisor elements to be divided into signal2 dividend elements

nLength Signal Length.

nRndMode various rounding modes.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.135 NppStatus nppsDiv_Round_8u_Sfs (const Npp8u ∗ *pSrc1*, const Npp8u ∗ *pSrc2*, Npp8u ∗ *pDst*, int *nLength*, NppRoundMode *nRndMode*, int *nScaleFactor*)

8-bit unsigned char signal divide signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal1 divisor elements to be divided into signal2 dividend elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nRndMode various rounding modes.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.136 NppStatus nppsDivC_16s_ISfs (Npp16s *nValue*, Npp16s ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short in place signal divided by constant, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be divided into each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.137 NppStatus nppsDivC_16s_Sfs (const Npp16s * *pSrc*, Npp16s *nValue*, Npp16s * *pDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short signal divided by constant, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be divided into each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.138 NppStatus nppsDivC_16sc_ISfs (Npp16sc *nValue*, Npp16sc * *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit integer complex number (16 bit real, 16 bit imaginary) signal divided by constant, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be divided into each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.139 NppStatus nppsDivC_16sc_Sfs (const Npp16sc * *pSrc*, Npp16sc *nValue*, Npp16sc * *pDst*, int *nLength*, int *nScaleFactor*)

16-bit integer complex number (16 bit real, 16 bit imaginary) signal divided by constant, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be divided into each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.140 NppStatus nppsDivC_16u_ISfs (Npp16u *nValue*, Npp16u ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short in place signal divided by constant, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be divided into each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.141 NppStatus nppsDivC_16u_Sfs (const Npp16u ∗ *pSrc*, Npp16u *nValue*, Npp16u ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short signal divided by constant, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be divided into each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.142 NppStatus nppsDivC_32f (const Npp32f ∗ *pSrc*, Npp32f *nValue*, Npp32f ∗ *pDst*, int *nLength*)

32-bit floating point signal divided by constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be divided into each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.143 NppStatus nppsDivC_32f_I (Npp32f *nValue*, Npp32f ∗ *pSrcDst*, int *nLength*)

32-bit floating point in place signal divided by constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be divided into each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.144 NppStatus nppsDivC_32fc (const Npp32fc ∗ *pSrc*, Npp32fc *nValue*, Npp32fc ∗ *pDst*, int *nLength*)

32-bit floating point complex number (32 bit real, 32 bit imaginary) signal divided by constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be divided into each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.145 NppStatus nppsDivC_32fc_I (Npp32fc *nValue*, Npp32fc ∗ *pSrcDst*, int *nLength*)

32-bit floating point complex number (32 bit real, 32 bit imaginary) in place signal divided by constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be divided into each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.146 NppStatus nppsDivC_32s_ISfs (Npp32s *nValue*, Npp32s ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer in place signal divided by constant and scale.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be divided into each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.147 NppStatus nppsDivC_32s_Sfs (const Npp32s ∗ *pSrc*, Npp32s *nValue*, Npp32s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer signal divided by constant and scale.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be divided into each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.148 NppStatus nppsDivC_32sc_ISfs (Npp32sc *nValue*, Npp32sc ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

32-bit integer complex number (32 bit real, 32 bit imaginary) in place signal divided by constant and scale.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be divided into each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.149 NppStatus nppsDivC_32sc_Sfs (const Npp32sc ∗ *pSrc*, Npp32sc *nValue*, Npp32sc ∗ *pDst*, int *nLength*, int *nScaleFactor*)

32-bit integer complex number (32 bit real, 32 bit imaginary) signal divided by constant and scale.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be divided into each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.150 NppStatus nppsDivC_64f (const Npp64f ∗ *pSrc*, Npp64f *nValue*, Npp64f ∗ *pDst*, int *nLength*)

64-bit floating point signal divided by constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be divided into each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.151 NppStatus nppsDivC_64f_I (Npp64f *nValue*, Npp64f ∗ *pSrcDst*, int *nLength*)

64-bit floating point in place signal divided by constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be divided into each vector element

nLength Length of the vectors, number of items.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.152 NppStatus nppsDivC_64fc (const Npp64fc * *pSrc*, Npp64fc *nValue*, Npp64fc * *pDst*, int *nLength*)

64-bit floating point complex number (64 bit real, 64 bit imaginary) signal divided by constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be divided into each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.153 NppStatus nppsDivC_64fc_I (Npp64fc *nValue*, Npp64fc * *pSrcDst*, int *nLength*)

64-bit floating point complex number (64 bit real, 64 bit imaginary) in place signal divided by constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be divided into each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.154 NppStatus nppsDivC_8u_ISfs (Npp8u *nValue*, Npp8u * *pSrcDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned char in place signal divided by constant, scale, then clamp to saturated value

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be divided into each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.155 NppStatus nppsDivC_8u_Sfs (const Npp8u ∗ *pSrc*, Npp8u *nValue*, Npp8u ∗ *pDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned char signal divided by constant, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be divided into each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.156 NppStatus nppsDivCRev_16s (const Npp16s ∗ *pSrc*, Npp16s *nValue*, Npp16s ∗ *pDst*, int *nLength*)

16-bit signed short constant divided by signal, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be divided by each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.157 NppStatus nppsDivCRev_16s_I (Npp16s *nValue*, Npp16s ∗ *pSrcDst*, int *nLength*)

16-bit signed short in place constant divided by signal, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be divided by each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.158 NppStatus nppsDivCRev_16u (const Npp16u * *pSrc*, Npp16u *nValue*, Npp16u * *pDst*, int *nLength*)

16-bit unsigned short vector divided by constant, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be divided by each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.159 NppStatus nppsDivCRev_16u_I (Npp16u *nValue*, Npp16u * *pSrcDst*, int *nLength*)

16-bit unsigned short in place constant divided by signal, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be divided by each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.160 NppStatus nppsDivCRev_32f (const Npp32f * *pSrc*, Npp32f *nValue*, Npp32f * *pDst*, int *nLength*)

32-bit floating point constant divided by signal.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be divided by each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.161 NppStatus nppsDivCRev_32f_I (Npp32f *nValue*, Npp32f ∗ *pSrcDst*, int *nLength*)

32-bit floating point in place constant divided by signal.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be divided by each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.162 NppStatus nppsDivCRev_32s (const Npp32s ∗ *pSrc*, Npp32s *nValue*, Npp32s ∗ *pDst*, int *nLength*)

32-bit signed integer constant divided by signal.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be divided by each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.163 NppStatus nppsDivCRev_32s_I (Npp32s *nValue*, Npp32s ∗ *pSrcDst*, int *nLength*)

32-bit signed integer in place constant divided by signal.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be divided by each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.164 NppStatus nppsDivCRev_64f (const Npp64f ∗ *pSrc*, Npp64f *nValue*, Npp64f ∗ *pDst*, int *nLength*)

64-bit floating point constant divided by signal.

Parameters:

pSrc Source Signal Pointer.
nValue Constant value to be divided by each vector element
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.165 NppStatus nppsDivCRev_64f_I (Npp64f *nValue*, Npp64f ∗ *pSrcDst*, int *nLength*)

64-bit floating point in place constant divided by signal.

Parameters:

pSrcDst In-Place Signal Pointer.
nValue Constant value to be divided by each vector element
nLength Length of the vectors, number of items.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.166 NppStatus nppsDivCRev_8u (const Npp8u ∗ *pSrc*, Npp8u *nValue*, Npp8u ∗ *pDst*, int *nLength*)

8-bit unsigned char signal divided by constant, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.
nValue Constant value to be divided by each vector element
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.167 NppStatus nppsDivCRev_8u_I (Npp8u *nValue*, Npp8u ∗ *pSrcDst*, int *nLength*)

8-bit unsigned char signal in place constant divided by signal, scale, then clamp to saturated value

Parameters:

pSrcDst In-Place Signal Pointer.
nValue Constant value to be divided by each vector element
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.168 NppStatus nppsExp_16s_ISfs (Npp16s ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short signal exponent, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.169 NppStatus nppsExp_16s_Sfs (const Npp16s ∗ *pSrc*, Npp16s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short signal exponent, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.170 NppStatus nppsExp_32f (const Npp32f ∗ *pSrc*, Npp32f ∗ *pDst*, int *nLength*)

32-bit floating point signal exponent.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.171 NppStatus nppsExp_32f64f (const Npp32f ∗ *pSrc*, Npp64f ∗ *pDst*, int *nLength*)

32-bit floating point signal exponent with 64-bit floating point result.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.172 NppStatus nppsExp_32f_I (Npp32f ∗ *pSrcDst*, int *nLength*)

32-bit floating point signal exponent.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.173 NppStatus nppsExp_32s_ISfs (Npp32s ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer signal exponent, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.174 NppStatus nppsExp_32s_Sfs (const Npp32s ∗ *pSrc*, Npp32s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer signal exponent, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.175 NppStatus nppsExp_64f (const Npp64f ∗ *pSrc*, Npp64f ∗ *pDst*, int *nLength*)

64-bit floating point signal exponent.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.176 NppStatus nppsExp_64f_I (Npp64f ∗ *pSrcDst*, int *nLength*)

64-bit floating point signal exponent.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.177 NppStatus nppsExp_64s_ISfs (Npp64s ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

64-bit signed integer signal exponent, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.178 NppStatus nppsExp_64s_Sfs (const Npp64s ∗ *pSrc*, Npp64s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

64-bit signed integer signal exponent, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.179 void nppsFree (void ∗ *pValues*)

Free method for any 2D allocated memory.

This method should be used to free memory allocated with any of the nppiMalloc_<modifier> methods.

Parameters:

pValues A pointer to memory allocated using nppiMalloc_<modifier>.

7.5.1.180 NppStatus nppsIntegral_32s (const Npp32s ∗ *pSrc*, Npp32s ∗ *pDst*, int *nLength*, Npp8u ∗ *pDeviceBuffer*)

7.5.1.181 NppStatus nppsIntegralGetBufferSize_32s (int *nLength*, int ∗ *hpBufferSize*)

7.5.1.182 NppStatus nppsLn_16s_ISfs (Npp16s ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short signal natural logarithm, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.183 NppStatus nppsLn_16s_Sfs (const Npp16s ∗ *pSrc*, Npp16s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short signal natural logarithm, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.184 NppStatus nppsLn_32f (const Npp32f ∗ *pSrc*, Npp32f ∗ *pDst*, int *nLength*)

32-bit floating point signal natural logarithm.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.185 NppStatus nppsLn_32f_I (Npp32f ∗ *pSrcDst*, int *nLength*)

32-bit floating point signal natural logarithm.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.186 NppStatus nppsLn_32s16s_Sfs (const Npp32s ∗ *pSrc*, Npp16s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer signal natural logarithm, scale, then clamp to 16-bit signed short saturated value.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.187 NppStatus nppsLn_32s_ISfs (Npp32s ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer signal natural logarithm, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.188 NppStatus nppsLn_32s_Sfs (const Npp32s ∗ *pSrc*, Npp32s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer signal natural logarithm, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.189 NppStatus nppsLn_64f (const Npp64f ∗ *pSrc*, Npp64f ∗ *pDst*, int *nLength*)

64-bit floating point signal natural logarithm.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.190 NppStatus nppsLn_64f32f (const Npp64f ∗ *pSrc*, Npp32f ∗ *pDst*, int *nLength*)

64-bit floating point signal natural logarithm with 32-bit floating point result.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.191 NppStatus nppsLn_64f_I (Npp64f ∗ *pSrcDst*, int *nLength*)

64-bit floating point signal natural logarithm.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.192 NppStatus nppsLShiftC_16s (const Npp16s ∗ *pSrc*, int *nValue*, Npp16s ∗ *pDst*, int *nLength*)

16-bit signed short signal left shift with constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be used to left shift each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.193 NppStatus nppsLShiftC_16s_I (int *nValue*, Npp16s ∗ *pSrcDst*, int *nLength*)

16-bit signed short in place signal left shift with constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be used to left shift each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.194 NppStatus nppsLShiftC_16u (const Npp16u ∗ *pSrc*, int *nValue*, Npp16u ∗ *pDst*, int *nLength*)

16-bit unsigned short signal left shift with constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be used to left shift each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.195 NppStatus nppsLShiftC_16u_I (int *nValue*, Npp16u * *pSrcDst*, int *nLength*)

16-bit unsigned short in place signal left shift with constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be used to left shift each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.196 NppStatus nppsLShiftC_32s (const Npp32s * *pSrc*, int *nValue*, Npp32s * *pDst*, int *nLength*)

32-bit signed integer signal left shift with constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be used to left shift each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.197 NppStatus nppsLShiftC_32s_I (int *nValue*, Npp32s * *pSrcDst*, int *nLength*)

32-bit signed signed integer in place signal left shift with constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be used to left shift each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.198 NppStatus nppsLShiftC_32u (const Npp32u ∗ *pSrc*, int *nValue*, Npp32u ∗ *pDst*, int *nLength*)

32-bit unsigned integer signal left shift with constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be used to left shift each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.199 NppStatus nppsLShiftC_32u_I (int *nValue*, Npp32u ∗ *pSrcDst*, int *nLength*)

32-bit unsigned signed integer in place signal left shift with constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be used to left shift each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.200 NppStatus nppsLShiftC_8u (const Npp8u ∗ *pSrc*, int *nValue*, Npp8u ∗ *pDst*, int *nLength*)

8-bit unsigned char signal left shift with constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be used to left shift each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.201 NppStatus nppsLShiftC_8u_I (int *nValue*, Npp8u ∗ *pSrcDst*, int *nLength*)

8-bit unsigned char in place signal left shift with constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be used to left shift each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.202 Npp16s∗ nppsMalloc_16s (int *nSize*)

16-bit signal allocator.

Parameters:

nSize Number of shorts in the new signal.

Returns:

A pointer to the new signal. 0 (NULL-pointer) indicates that an error occurred during allocation.

7.5.1.203 Npp16sc∗ nppsMalloc_16sc (int *nSize*)

16-bit complex-value signal allocator.

Parameters:

nSize Number of 16-bit complex numbers in the new signal.

Returns:

A pointer to the new signal. 0 (NULL-pointer) indicates that an error occurred during allocation.

7.5.1.204 Npp16u∗ nppsMalloc_16u (int *nSize*)

16-bit unsigned signal allocator.

Parameters:

nSize Number of unsigned shorts in the new signal.

Returns:

A pointer to the new signal. 0 (NULL-pointer) indicates that an error occurred during allocation.

7.5.1.205 Npp32f∗ nppsMalloc_32f (int *nSize*)

32-bit float signal allocator.

Parameters:

nSize Number of floats in the new signal.

Returns:

A pointer to the new signal. 0 (NULL-pointer) indicates that an error occurred during allocation.

7.5.1.206 Npp32fc∗ nppsMalloc_32fc (int *nSize*)

32-bit complex float signal allocator.

Parameters:

nSize Number of complex float values in the new signal.

Returns:

A pointer to the new signal. 0 (NULL-pointer) indicates that an error occurred during allocation.

7.5.1.207 Npp32s∗ nppsMalloc_32s (int *nSize*)

32-bit integer signal allocator.

Parameters:

nSize Number of ints in the new signal.

Returns:

A pointer to the new signal. 0 (NULL-pointer) indicates that an error occurred during allocation.

7.5.1.208 Npp32sc∗ nppsMalloc_32sc (int *nSize*)

32-bit complex integer signal allocator.

Parameters:

nSize Number of complex integner values in the new signal.

Returns:

A pointer to the new signal. 0 (NULL-pointer) indicates that an error occurred during allocation.

7.5.1.209 Npp32u∗ nppsMalloc_32u (int *nSize*)

32-bit unsigned signal allocator.

Parameters:

nSize Number of unsigned ints in the new signal.

Returns:

A pointer to the new signal. 0 (NULL-pointer) indicates that an error occurred during allocation.

7.5.1.210 Npp64f∗ nppsMalloc_64f (int *nSize*)

64-bit float (double) signal allocator.

Parameters:

nSize Number of doubles in the new signal.

Returns:

A pointer to the new signal. 0 (NULL-pointer) indicates that an error occurred during allocation.

7.5.1.211 Npp64fc∗ nppsMalloc_64fc (int *nSize*)

64-bit complex complex signal allocator.

Parameters:

nSize Number of complex double valuess in the new signal.

Returns:

A pointer to the new signal. 0 (NULL-pointer) indicates that an error occurred during allocation.

7.5.1.212 Npp64s∗ nppsMalloc_64s (int *nSize*)

64-bit long integer signal allocator.

Parameters:

nSize Number of long ints in the new signal.

Returns:

A pointer to the new signal. 0 (NULL-pointer) indicates that an error occurred during allocation.

7.5.1.213 Npp64sc∗ nppsMalloc_64sc (int *nSize*)

64-bit complex long integer signal allocator.

Parameters:

nSize Number of complex long int values in the new signal.

Returns:

A pointer to the new signal. 0 (NULL-pointer) indicates that an error occurred during allocation.

7.5.1.214 Npp8u∗ nppsMalloc_8u (int *nSize*)

8-bit unsigned signal allocator.

Parameters:

nSize Number of unsigned chars in the new signal.

Returns:

A pointer to the new signal. 0 (NULL-pointer) indicates that an error occurred during allocation.

7.5.1.215 NppStatus nppsMax_16s (const Npp16s ∗ *pSrc*, int *nLength*, Npp16s ∗ *pMax*, Npp8u ∗ *pDeviceBuffer*)

16-bit integer vector max method

Parameters:

pSrc Source Signal Pointer.
nLength Signal Length.
pMax Pointer to the output result.
pDeviceBuffer Pointer to the required device memory allocation.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.216 NppStatus nppsMax_32f (const Npp32f ∗ *pSrc*, int *nLength*, Npp32f ∗ *pMax*, Npp8u ∗ *pDeviceBuffer*)

32-bit float vector max method

Parameters:

pSrc Source Signal Pointer.
nLength Signal Length.
pMax Pointer to the output result.
pDeviceBuffer Pointer to the required device memory allocation.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.217 NppStatus nppsMax_32s (const Npp32s ∗ *pSrc*, int *nLength*, Npp32s ∗ *pMax*, Npp8u ∗ *pDeviceBuffer*)

32-bit integer vector max method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMax Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.218 NppStatus nppsMax_64f (const Npp64f ∗ *pSrc*, int *nLength*, Npp64f ∗ *pMax*, Npp8u ∗ *pDeviceBuffer*)

64-bit float vector max method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMax Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.219 NppStatus nppsMin_16s (const Npp16s ∗ *pSrc*, int *nLength*, Npp16s ∗ *pMin*, Npp8u ∗ *pDeviceBuffer*)

16-bit integer vector min method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMin Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.220 NppStatus nppsMin_32f (const Npp32f $*$ *pSrc*, int *nLength*, Npp32f $*$ *pMin*, Npp8u $*$ *pDeviceBuffer*)

32-bit integer vector min method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMin Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.221 NppStatus nppsMin_32s (const Npp32s $*$ *pSrc*, int *nLength*, Npp32s $*$ *pMin*, Npp8u $*$ *pDeviceBuffer*)

32-bit integer vector min method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMin Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.222 NppStatus nppsMin_64f (const Npp64f $*$ *pSrc*, int *nLength*, Npp64f $*$ *pMin*, Npp8u $*$ *pDeviceBuffer*)

64-bit integer vector min method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMin Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.223 NppStatus nppsMinMax_16s (const Npp16s ∗ *pSrc*, int *nLength*, Npp16s ∗ *pMin*, Npp16s ∗ *pMax*, Npp8u ∗ *pDeviceBuffer*)

16-bit signed short vector min and max method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMin Pointer to the min output result.

pMax Pointer to the max output result.

pDeviceBuffer Pointer to the required device memory allocation.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.224 NppStatus nppsMinMax_16u (const Npp16u ∗ *pSrc*, int *nLength*, Npp16u ∗ *pMin*, Npp16u ∗ *pMax*, Npp8u ∗ *pDeviceBuffer*)

16-bit unsigned short vector min and max method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMin Pointer to the min output result.

pMax Pointer to the max output result.

pDeviceBuffer Pointer to the required device memory allocation.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.225 NppStatus nppsMinMax_32f (const Npp32f ∗ *pSrc*, int *nLength*, Npp32f ∗ *pMin*, Npp32f ∗ *pMax*, Npp8u ∗ *pDeviceBuffer*)

32-bit float vector min and max method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMin Pointer to the min output result.

pMax Pointer to the max output result.

pDeviceBuffer Pointer to the required device memory allocation.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.226 NppStatus nppsMinMax_32s (const Npp32s * *pSrc*, int *nLength*, Npp32s * *pMin*, Npp32s * *pMax*, Npp8u * *pDeviceBuffer*)

32-bit signed int vector min and max method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMin Pointer to the min output result.

pMax Pointer to the max output result.

pDeviceBuffer Pointer to the required device memory allocation.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.227 NppStatus nppsMinMax_32u (const Npp32u * *pSrc*, int *nLength*, Npp32u * *pMin*, Npp32u * *pMax*, Npp8u * *pDeviceBuffer*)

32-bit unsigned int vector min and max method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMin Pointer to the min output result.

pMax Pointer to the max output result.

pDeviceBuffer Pointer to the required device memory allocation.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.228 NppStatus nppsMinMax_64f (const Npp64f * *pSrc*, int *nLength*, Npp64f * *pMin*, Npp64f * *pMax*, Npp8u * *pDeviceBuffer*)

64-bit double vector min and max method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMin Pointer to the min output result.

pMax Pointer to the max output result.

pDeviceBuffer Pointer to the required device memory allocation.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.229 NppStatus nppsMinMax_8u (const Npp8u ∗ *pSrc*, int *nLength*, Npp8u ∗ *pMin*, Npp8u ∗ *pMax*, Npp8u ∗ *pDeviceBuffer*)

8-bit char vector min and max method

Parameters:

pSrc Source Signal Pointer.
nLength Signal Length.
pMin Pointer to the min output result.
pMax Pointer to the max output result.
pDeviceBuffer Pointer to the required device memory allocation.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.230 NppStatus nppsMinMaxGetBufferSize_16s (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsMinMax_16s.

Parameters:

nLength Signal Length.
hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.5.1.231 NppStatus nppsMinMaxGetBufferSize_16u (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsMinMax_16u.

Parameters:

nLength Signal Length.
hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.5.1.232 NppStatus nppsMinMaxGetBufferSize_32f (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsMinMax_32f.

Parameters:

nLength Signal Length.
hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.5.1.233 NppStatus nppsMinMaxGetBufferSize_32s (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsMinMax_32s.

Parameters:

nLength Signal Length.
hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.5.1.234 NppStatus nppsMinMaxGetBufferSize_32u (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsMinMax_32u.

Parameters:

nLength Signal Length.
hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.5.1.235 NppStatus nppsMinMaxGetBufferSize_64f (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsMinMax_64f.

Parameters:

nLength Signal Length.
hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.5.1.236 NppStatus nppsMinMaxGetBufferSize_8u (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsMinMax_8u.

Parameters:

nLength Signal Length.
hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.5.1.237 NppStatus nppsMul_16s (const Npp16s $*$ *pSrc1*, const Npp16s $*$ *pSrc2*, Npp16s $*$ *pDst*, int *nLength*)

16-bit signed short signal times signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be multiplied by signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.238 NppStatus nppsMul_16s32f (const Npp16s $*$ *pSrc1*, const Npp16s $*$ *pSrc2*, Npp32f $*$ *pDst*, int *nLength*)

16-bit signed short signal times signal with 32-bit floating point result, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be multiplied by signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.239 NppStatus nppsMul_16s32s_Sfs (const Npp16s $*$ *pSrc1*, const Npp16s $*$ *pSrc2*, Npp32s $*$ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short signal times signal, scale, then clamp to 32-bit signed saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal2 elements to be multiplied by signal1 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.240 NppStatus nppsMul_16s_I (const Npp16s ∗ *pSrc*, Npp16s ∗ *pSrcDst*, int *nLength*)

16-bit signed short in place signal times signal, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be multiplied by signal1 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.241 NppStatus nppsMul_16s_ISfs (const Npp16s ∗ *pSrc*, Npp16s ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short in place signal times signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be multiplied by signal1 elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.242 NppStatus nppsMul_16s_Sfs (const Npp16s ∗ *pSrc1*, const Npp16s ∗ *pSrc2*, Npp16s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short signal times signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal2 elements to be multiplied by signal1 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.243 NppStatus nppsMul_16sc_ISfs (const Npp16sc ∗ *pSrc*, Npp16sc ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit complex signed short in place signal times signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be multiplied by signal1 elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.244 NppStatus nppsMul_16sc_Sfs (const Npp16sc ∗ *pSrc1*, const Npp16sc ∗ *pSrc2*, Npp16sc ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit signed complex short signal times signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal2 elements to be multiplied by signal1 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.245 NppStatus nppsMul_16u16s_Sfs (const Npp16u ∗ *pSrc1*, const Npp16s ∗ *pSrc2*, Npp16s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short signal times 16-bit signed short signal, scale, then clamp to 16-bit signed saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal2 elements to be multiplied by signal1 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.246 NppStatus nppsMul_16u_ISfs (const Npp16u ∗ *pSrc*, Npp16u ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short in place signal times signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be multiplied by signal1 elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.247 NppStatus nppsMul_16u_Sfs (const Npp16u ∗ *pSrc1*, const Npp16u ∗ *pSrc2*, Npp16u ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short signal time signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal2 elements to be multiplied by signal1 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.248 NppStatus nppsMul_32f (const Npp32f ∗ *pSrc1*, const Npp32f ∗ *pSrc2*, Npp32f ∗ *pDst*, int *nLength*)

32-bit floating point signal times signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be multiplied by signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.249 NppStatus nppsMul_32f32fc (const Npp32f ∗ *pSrc1*, const Npp32fc ∗ *pSrc2*, Npp32fc ∗ *pDst*, int *nLength*)

32-bit floating point signal times 32-bit complex floating point signal with complex 32-bit floating point result, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be multiplied by signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.250 NppStatus nppsMul_32f32fc_I (const Npp32f ∗ *pSrc*, Npp32fc ∗ *pSrcDst*, int *nLength*)

32-bit complex floating point in place signal times 32-bit floating point signal, then clamp to 32-bit complex floating point saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be multiplied by signal1 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.251 NppStatus nppsMul_32f_I (const Npp32f ∗ *pSrc*, Npp32f ∗ *pSrcDst*, int *nLength*)

32-bit floating point in place signal times signal, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be multiplied by signal1 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.252 NppStatus nppsMul_32fc (const Npp32fc ∗ *pSrc1*, const Npp32fc ∗ *pSrc2*, Npp32fc ∗ *pDst*, int *nLength*)

32-bit complex floating point signal times signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be multiplied by signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.253 NppStatus nppsMul_32fc_I (const Npp32fc ∗ *pSrc*, Npp32fc ∗ *pSrcDst*, int *nLength*)

32-bit complex floating point in place signal times signal, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be multiplied by signal1 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.254 NppStatus nppsMul_32s32sc_ISfs (const Npp32s ∗ *pSrc*, Npp32sc ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

32-bit complex signed integer in place signal times 32-bit signed integer signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be multiplied by signal1 elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.255 NppStatus nppsMul_32s32sc_Sfs (const Npp32s * *pSrc1*, const Npp32sc * *pSrc2*, Npp32sc * *pDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer signal times 32-bit complex signed integer signal, scale, then clamp to 32-bit complex integer saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal2 elements to be multiplied by signal1 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.256 NppStatus nppsMul_32s_ISfs (const Npp32s * *pSrc*, Npp32s * *pSrcDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer in place signal times signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be multiplied by signal1 elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.257 NppStatus nppsMul_32s_Sfs (const Npp32s * *pSrc1*, const Npp32s * *pSrc2*, Npp32s * *pDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer signal times signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal2 elements to be multiplied by signal1 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.258 NppStatus nppsMul_32sc_ISfs (const Npp32sc ∗ *pSrc*, Npp32sc ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

32-bit complex signed integer in place signal times signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be multiplied by signal1 elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.259 NppStatus nppsMul_32sc_Sfs (const Npp32sc ∗ *pSrc1*, const Npp32sc ∗ *pSrc2*, Npp32sc ∗ *pDst*, int *nLength*, int *nScaleFactor*)

32-bit signed complex integer signal times signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal2 elements to be multiplied by signal1 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.260 NppStatus nppsMul_64f (const Npp64f ∗ *pSrc1*, const Npp64f ∗ *pSrc2*, Npp64f ∗ *pDst*, int *nLength*)

64-bit floating point signal times signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be multiplied by signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.261 NppStatus nppsMul_64f_I (const Npp64f * *pSrc*, Npp64f * *pSrcDst*, int *nLength*)

64-bit floating point in place signal times signal, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be multiplied by signal1 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.262 NppStatus nppsMul_64fc (const Npp64fc * *pSrc1*, const Npp64fc * *pSrc2*, Npp64fc * *pDst*, int *nLength*)

64-bit complex floating point signal times signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be multiplied by signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.263 NppStatus nppsMul_64fc_I (const Npp64fc * *pSrc*, Npp64fc * *pSrcDst*, int *nLength*)

64-bit complex floating point in place signal times signal, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be multiplied by signal1 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.264 NppStatus nppsMul_8u16u (const Npp8u * *pSrc1*, const Npp8u * *pSrc2*, Npp16u * *pDst*, int *nLength*)

8-bit unsigned char signal times signal with 16-bit unsigned result, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be multiplied by signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.265 NppStatus nppsMul_8u_ISfs (const Npp8u ∗ *pSrc*, Npp8u ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned char in place signal times signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be multiplied by signal1 elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.266 NppStatus nppsMul_8u_Sfs (const Npp8u ∗ *pSrc1*, const Npp8u ∗ *pSrc2*, Npp8u ∗ *pDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned char signal times signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal2 elements to be multiplied by signal1 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.267 NppStatus nppsMul_Low_32s_Sfs (const Npp32s * *pSrc1*, const Npp32s * *pSrc2*, Npp32s * *pDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer signal times signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal2 elements to be multiplied by signal1 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.268 NppStatus nppsMulC_16s_ISfs (Npp16s *nValue*, Npp16s * *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short in place signal times constant, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be multiplied by each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.269 NppStatus nppsMulC_16s_Sfs (const Npp16s * *pSrc*, Npp16s *nValue*, Npp16s * *pDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short signal times constant, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be multiplied by each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.270 NppStatus nppsMulC_16sc_ISfs (Npp16sc *nValue*, Npp16sc ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit integer complex number (16 bit real, 16 bit imaginary)signal times constant, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be multiplied by each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.271 NppStatus nppsMulC_16sc_Sfs (const Npp16sc ∗ *pSrc*, Npp16sc *nValue*, Npp16sc ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit integer complex number (16 bit real, 16 bit imaginary)signal times constant, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be multiplied by each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.272 NppStatus nppsMulC_16u_ISfs (Npp16u *nValue*, Npp16u ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short in place signal times constant, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be multiplied by each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.273 NppStatus nppsMulC_16u_Sfs (const Npp16u * *pSrc*, Npp16u *nValue*, Npp16u * *pDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short signal times constant, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be multiplied by each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.274 NppStatus nppsMulC_32f (const Npp32f * *pSrc*, Npp32f *nValue*, Npp32f * *pDst*, int *nLength*)

32-bit floating point signal times constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be multiplied by each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.275 NppStatus nppsMulC_32f16s_Sfs (const Npp32f * *pSrc*, Npp32f *nValue*, Npp16s * *pDst*, int *nLength*, int *nScaleFactor*)

32-bit floating point signal times constant with output converted to 16-bit signed integer with scaling and saturation of output result.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be multiplied by each vector element

pDst Destination Signal Pointer.

nScaleFactor Integer Result Scaling.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.276 NppStatus nppsMulC_32f_I (Npp32f *nValue*, Npp32f ∗ *pSrcDst*, int *nLength*)

32-bit floating point in place signal times constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be multiplied by each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.277 NppStatus nppsMulC_32fc (const Npp32fc ∗ *pSrc*, Npp32fc *nValue*, Npp32fc ∗ *pDst*, int *nLength*)

32-bit floating point complex number (32 bit real, 32 bit imaginary) signal times constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be multiplied by each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.278 NppStatus nppsMulC_32fc_I (Npp32fc *nValue*, Npp32fc ∗ *pSrcDst*, int *nLength*)

32-bit floating point complex number (32 bit real, 32 bit imaginary) in place signal times constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be multiplied by each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.279 NppStatus nppsMulC_32s_ISfs (Npp32s *nValue*, Npp32s ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer in place signal times constant and scale.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be multiplied by each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.280 NppStatus nppsMulC_32s_Sfs (const Npp32s * *pSrc*, Npp32s *nValue*, Npp32s * *pDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer signal times constant and scale.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be multiplied by each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.281 NppStatus nppsMulC_32sc_ISfs (Npp32sc *nValue*, Npp32sc * *pSrcDst*, int *nLength*, int *nScaleFactor*)

32-bit integer complex number (32 bit real, 32 bit imaginary) in place signal times constant and scale.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be multiplied by each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.282 NppStatus nppsMulC_32sc_Sfs (const Npp32sc * *pSrc*, Npp32sc *nValue*, Npp32sc * *pDst*, int *nLength*, int *nScaleFactor*)

32-bit integer complex number (32 bit real, 32 bit imaginary) signal times constant and scale.

Parameters:

pSrc	Source Signal Pointer.
nValue	Constant value to be multiplied by each vector element
pDst	Destination Signal Pointer.
nLength	Signal Length.
nScaleFactor	Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.283 NppStatus nppsMulC_64f (const Npp64f * *pSrc*, Npp64f *nValue*, Npp64f * *pDst*, int *nLength*)

64-bit floating point signal times constant.

Parameters:

pSrc	Source Signal Pointer.
nValue	Constant value to be multiplied by each vector element
pDst	Destination Signal Pointer.
nLength	Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.284 NppStatus nppsMulC_64f64s_ISfs (Npp64f *nValue*, Npp64s * *pDst*, int *nLength*, int *nScaleFactor*)

64-bit floating point signal times constant with in place conversion to 64-bit signed integer and with scaling and saturation of output result.

Parameters:

nValue	Constant value to be multiplied by each vector element
pDst	Destination Signal Pointer.
nLength	Signal Length.
nScaleFactor	Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.285 NppStatus nppsMulC_64f_I (Npp64f *nValue*, Npp64f * *pSrcDst*, int *nLength*)

64-bit floating point, in place signal times constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be multiplied by each vector element

nLength Length of the vectors, number of items.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.286 NppStatus nppsMulC_64fc (const Npp64fc * *pSrc*, Npp64fc *nValue*, Npp64fc * *pDst*, int *nLength*)

64-bit floating point complex number (64 bit real, 64 bit imaginary) signal times constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be multiplied by each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.287 NppStatus nppsMulC_64fc_I (Npp64fc *nValue*, Npp64fc * *pSrcDst*, int *nLength*)

64-bit floating point complex number (64 bit real, 64 bit imaginary) in place signal times constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be multiplied by each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.288 NppStatus nppsMulC_8u_ISfs (Npp8u *nValue*, Npp8u * *pSrcDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned char in place signal times constant, scale, then clamp to saturated value

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be multiplied by each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.289 NppStatus nppsMulC_8u_Sfs (const Npp8u ∗ *pSrc*, Npp8u *nValue*, Npp8u ∗ *pDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned char signal times constant, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be multiplied by each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.290 NppStatus nppsMulC_Low_32f16s (const Npp32f ∗ *pSrc*, Npp32f *nValue*, Npp16s ∗ *pDst*, int *nLength*)

32-bit floating point signal times constant with output converted to 16-bit signed integer.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be multiplied by each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.291 NppStatus nppsNormalize_16s_Sfs (const Npp16s * *pSrc*, Npp16s * *pDst*, int *nLength*, Npp16s *vSub*, int *vDiv*, int *nScaleFactor*)

16-bit signed short signal normalize, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.
pDst Destination Signal Pointer.
nLength Signal Length.
vSub value subtracted from each signal element before division
vDiv divisor of post-subtracted signal element dividend
nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.292 NppStatus nppsNormalize_16sc_Sfs (const Npp16sc * *pSrc*, Npp16sc * *pDst*, int *nLength*, Npp16sc *vSub*, int *vDiv*, int *nScaleFactor*)

16-bit complex signed short signal normalize, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.
pDst Destination Signal Pointer.
nLength Signal Length.
vSub value subtracted from each signal element before division
vDiv divisor of post-subtracted signal element dividend
nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.293 NppStatus nppsNormalize_32f (const Npp32f * *pSrc*, Npp32f * *pDst*, int *nLength*, Npp32f *vSub*, Npp32f *vDiv*)

32-bit floating point signal normalize.

Parameters:

pSrc Source Signal Pointer.
pDst Destination Signal Pointer.
nLength Signal Length.
vSub value subtracted from each signal element before division
vDiv divisor of post-subtracted signal element dividend

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.294 NppStatus nppsNormalize_32fc (const Npp32fc ∗ *pSrc*, Npp32fc ∗ *pDst*, int *nLength*, Npp32fc *vSub*, Npp32f *vDiv*)

32-bit complex floating point signal normalize.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

vSub value subtracted from each signal element before division

vDiv divisor of post-subtracted signal element dividend

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.295 NppStatus nppsNormalize_64f (const Npp64f ∗ *pSrc*, Npp64f ∗ *pDst*, int *nLength*, Npp64f *vSub*, Npp64f *vDiv*)

64-bit floating point signal normalize.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

vSub value subtracted from each signal element before division

vDiv divisor of post-subtracted signal element dividend

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.296 NppStatus nppsNormalize_64fc (const Npp64fc ∗ *pSrc*, Npp64fc ∗ *pDst*, int *nLength*, Npp64fc *vSub*, Npp64f *vDiv*)

64-bit complex floating point signal normalize.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

vSub value subtracted from each signal element before division

vDiv divisor of post-subtracted signal element dividend

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.297 NppStatus nppsNot_16u (const Npp16u $*$ *pSrc*, Npp16u $*$ *pDst*, int *nLength*)

16-bit unsigned short not signal.

Parameters:

pSrc Source Signal Pointer.
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.298 NppStatus nppsNot_16u_I (Npp16u $*$ *pSrcDst*, int *nLength*)

16-bit unsigned short in place not signal.

Parameters:

pSrcDst In-Place Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.299 NppStatus nppsNot_32u (const Npp32u $*$ *pSrc*, Npp32u $*$ *pDst*, int *nLength*)

32-bit unsigned integer not signal.

Parameters:

pSrc Source Signal Pointer.
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.300 NppStatus nppsNot_32u_I (Npp32u $*$ *pSrcDst*, int *nLength*)

32-bit unsigned signed integer in place not signal.

Parameters:

pSrcDst In-Place Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.301 NppStatus nppsNot_8u (const Npp8u ∗ *pSrc*, Npp8u ∗ *pDst*, int *nLength*)

8-bit unsigned char not signal.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.302 NppStatus nppsNot_8u_I (Npp8u ∗ *pSrcDst*, int *nLength*)

8-bit unsigned char in place not signal.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.303 NppStatus nppsOr_16u (const Npp16u ∗ *pSrc1*, const Npp16u ∗ *pSrc2*, Npp16u ∗ *pDst*, int *nLength*)

16-bit unsigned short signal or with signal.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be ored with signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.304 NppStatus nppsOr_16u_I (const Npp16u ∗ *pSrc*, Npp16u ∗ *pSrcDst*, int *nLength*)

16-bit unsigned short in place signal or with signal.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be ored with signal1 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.305 NppStatus nppsOr_32u (const Npp32u ∗ *pSrc1*, const Npp32u ∗ *pSrc2*, Npp32u ∗ *pDst*, int *nLength*)

32-bit unsigned integer signal or with signal.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be ored with signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.306 NppStatus nppsOr_32u_I (const Npp32u ∗ *pSrc*, Npp32u ∗ *pSrcDst*, int *nLength*)

32-bit unsigned integer in place signal or with signal.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be ored with signal1 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.307 NppStatus nppsOr_8u (const Npp8u ∗ *pSrc1*, const Npp8u ∗ *pSrc2*, Npp8u ∗ *pDst*, int *nLength*)

8-bit unsigned char signal or with signal.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be ored with signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.308 NppStatus nppsOr_8u_I (const Npp8u ∗ *pSrc*, Npp8u ∗ *pSrcDst*, int *nLength*)

8-bit unsigned char in place signal or with signal.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be ored with signal1 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.309 NppStatus nppsOrC_16u (const Npp16u ∗ *pSrc*, Npp16u *nValue*, Npp16u ∗ *pDst*, int *nLength*)

16-bit unsigned short signal or with constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be ored with each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.310 NppStatus nppsOrC_16u_I (Npp16u *nValue*, Npp16u ∗ *pSrcDst*, int *nLength*)

16-bit unsigned short in place signal or with constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be ored with each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.311 NppStatus nppsOrC_32u (const Npp32u ∗ *pSrc*, Npp32u *nValue*, Npp32u ∗ *pDst*, int *nLength*)

32-bit unsigned integer signal or with constant.

Parameters:

pSrc Source Signal Pointer.
nValue Constant value to be ored with each vector element
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.312 NppStatus nppsOrC_32u_I (Npp32u *nValue*, Npp32u ∗ *pSrcDst*, int *nLength*)

32-bit unsigned signed integer in place signal or with constant.

Parameters:

pSrcDst In-Place Signal Pointer.
nValue Constant value to be ored with each vector element
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.313 NppStatus nppsOrC_8u (const Npp8u ∗ *pSrc*, Npp8u *nValue*, Npp8u ∗ *pDst*, int *nLength*)

8-bit unsigned char signal or with constant.

Parameters:

pSrc Source Signal Pointer.
nValue Constant value to be ored with each vector element
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.314 NppStatus nppsOrC_8u_I (Npp8u *nValue*, Npp8u ∗ *pSrcDst*, int *nLength*)

8-bit unsigned char in place signal or with constant.

Parameters:

pSrcDst In-Place Signal Pointer.
nValue Constant value to be ored with each vector element
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.315 NppStatus nppsReductionGetBufferSize_16s (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for 16s reductions.

This primitive provides the correct buffer size for nppsSum_16s, nppsMin_16s, nppsMax_16s.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.5.1.316 NppStatus nppsReductionGetBufferSize_16s_Sfs (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for 16s reductions with integer-results scaling.

This primitive provides the correct buffer size for nppsSum_16s_Sfs.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.5.1.317 NppStatus nppsReductionGetBufferSize_16sc (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for 16sc reductions.

This primitive provides the correct buffer size for nppsSum_16sc, nppsMin_16sc, nppsMax_16sc.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.5.1.318 NppStatus nppsReductionGetBufferSize_16sc_Sfs (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for 16sc reductions with integer-results scaling.

This primitive provides the correct buffer size for nppsSum_16sc_Sfs.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.5.1.319 NppStatus nppsReductionGetBufferSize_16u (int *nLength*, int * *hpBufferSize*) [inline]

Device-buffer size (in bytes) for 16u reductions.

This primitive provides the correct buffer size for nppsSum_16u, nppsMin_16u, nppsMax_16u.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.5.1.320 NppStatus nppsReductionGetBufferSize_32f (int *nLength*, int * *hpBufferSize*)

Device-buffer size (in bytes) for 32f reductions.

This primitive provides the correct buffer size for nppsSum_32f, nppsMin_32f, nppsMax_32f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.5.1.321 NppStatus nppsReductionGetBufferSize_32fc (int *nLength*, int * *hpBufferSize*)

Device-buffer size (in bytes) for 32fc reductions.

This primitive provides the correct buffer size for nppsSum_32fc, nppsMin_32fc, nppsMax_32fc.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.5.1.322 NppStatus nppsReductionGetBufferSize_32s (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for 32s reductions.

This primitive provides the correct buffer size for nppsSum_32sc, nppsMin_32sc, nppsMax_32sc.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.5.1.323 NppStatus nppsReductionGetBufferSize_32s_Sfs (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for 32s reductions with integer-results scaling.

This primitive provides the correct buffer size for nppsSum_32s_Sfs.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.5.1.324 NppStatus nppsReductionGetBufferSize_32sc (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for 32sc reductions.

This primitive provides the correct buffer size for nppsSum_32sc, nppsMin_32sc, nppsMax_32sc.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.5.1.325 NppStatus nppsReductionGetBufferSize_32u (int *nLength*, int ∗ *hpBufferSize*) `[inline]`

Device-buffer size (in bytes) for 32u reductions.

This primitive provides the correct buffer size for nppsSum_32u, nppsMin_32u, nppsMax_32u.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer*.

Returns:

NPP_SUCCESS

7.5.1.326 NppStatus nppsReductionGetBufferSize_64f (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for 64f reductions.

This primitive provides the correct buffer size for nppsSum_64f, nppsMin_64f, nppsMax_64f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer*.

Returns:

NPP_SUCCESS

7.5.1.327 NppStatus nppsReductionGetBufferSize_64fc (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for 64fc reductions.

This primitive provides the correct buffer size for nppsSum_64fc, nppsMin_64fc, nppsMax_64fc.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer*.

Returns:

NPP_SUCCESS

7.5.1.328 NppStatus nppsReductionGetBufferSize_64s (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for 64s reductions.

This primitive provides the correct buffer size for nppsSum_64s, nppsMin_64s, nppsMax_64s.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer*.

Returns:

NPP_SUCCESS

7.5.1.329 NppStatus nppsReductionGetBufferSize_8u (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for 8u reductions.

This primitive provides the correct buffer size for nppsSum_8u, nppsMin_8u, nppsMax_8u.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.5.1.330 NppStatus nppsRShiftC_16s (const Npp16s ∗ *pSrc*, int *nValue*, Npp16s ∗ *pDst*, int *nLength*)

16-bit signed short signal right shift with constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be used to right shift each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.331 NppStatus nppsRShiftC_16s_I (int *nValue*, Npp16s ∗ *pSrcDst*, int *nLength*)

16-bit signed short in place signal right shift with constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be used to right shift each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.332 NppStatus nppsRShiftC_16u (const Npp16u ∗ *pSrc*, int *nValue*, Npp16u ∗ *pDst*, int *nLength*)

16-bit unsigned short signal right shift with constant.

Parameters:

pSrc Source Signal Pointer.
nValue Constant value to be used to right shift each vector element
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.333 NppStatus nppsRShiftC_16u_I (int *nValue*, Npp16u ∗ *pSrcDst*, int *nLength*)

16-bit unsigned short in place signal right shift with constant.

Parameters:

pSrcDst In-Place Signal Pointer.
nValue Constant value to be used to right shift each vector element
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.334 NppStatus nppsRShiftC_32s (const Npp32s ∗ *pSrc*, int *nValue*, Npp32s ∗ *pDst*, int *nLength*)

32-bit signed integer signal right shift with constant.

Parameters:

pSrc Source Signal Pointer.
nValue Constant value to be used to right shift each vector element
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.335 NppStatus nppsRShiftC_32s_I (int *nValue*, Npp32s ∗ *pSrcDst*, int *nLength*)

32-bit signed signed integer in place signal right shift with constant.

Parameters:

pSrcDst In-Place Signal Pointer.
nValue Constant value to be used to right shift each vector element
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.336 NppStatus nppsRShiftC_32u (const Npp32u ∗ *pSrc*, int *nValue*, Npp32u ∗ *pDst*, int *nLength*)

32-bit unsigned integer signal right shift with constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be used to right shift each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.337 NppStatus nppsRShiftC_32u_I (int *nValue*, Npp32u ∗ *pSrcDst*, int *nLength*)

32-bit unsigned signed integer in place signal right shift with constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be used to right shift each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.338 NppStatus nppsRShiftC_8u (const Npp8u ∗ *pSrc*, int *nValue*, Npp8u ∗ *pDst*, int *nLength*)

8-bit unsigned char signal right shift with constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be used to right shift each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.339 NppStatus nppsRShiftC_8u_I (int *nValue*, Npp8u * *pSrcDst*, int *nLength*)

8-bit unsigned char in place signal right shift with constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be used to right shift each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.340 NppStatus nppsSet_16s (Npp16s *nValue*, Npp16s * *pDst*, int *nLength*)

16-bit integer, vector set method.

Parameters:

nValue Value used to initialize the vector pDst.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.341 NppStatus nppsSet_16sc (Npp16sc *nValue*, Npp16sc * *pDst*, int *nLength*)

16-bit integer complex, vector set method.

Parameters:

nValue Value used to initialize the vector pDst.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.342 NppStatus nppsSet_32f (Npp32f *nValue*, Npp32f * *pDst*, int *nLength*)

32-bit float, vector set method.

Parameters:

nValue Value used to initialize the vector pDst.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.343 NppStatus nppsSet_32fc (Npp32fc *nValue*, Npp32fc ∗ *pDst*, int *nLength*)

32-bit float complex, vector set method.

Parameters:

nValue Value used to initialize the vector pDst.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.344 NppStatus nppsSet_32s (Npp32s *nValue*, Npp32s ∗ *pDst*, int *nLength*)

32-bit integer, vector set method.

Parameters:

nValue Value used to initialize the vector pDst.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.345 NppStatus nppsSet_32sc (Npp32sc *nValue*, Npp32sc ∗ *pDst*, int *nLength*)

32-bit integer complex, vector set method.

Parameters:

nValue Value used to initialize the vector pDst.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.346 NppStatus nppsSet_64f (Npp64f *nValue*, Npp64f ∗ *pDst*, int *nLength*)

64-bit double, vector set method.

Parameters:

nValue Value used to initialize the vector pDst.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.347 NppStatus nppsSet_64fc (Npp64fc *nValue*, Npp64fc ∗ *pDst*, int *nLength*)

64-bit double complex, vector set method.

Parameters:

nValue Value used to initialize the vector pDst.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.348 NppStatus nppsSet_64s (Npp64s *nValue*, Npp64s ∗ *pDst*, int *nLength*)

64-bit long long integer, vector set method.

Parameters:

nValue Value used to initialize the vector pDst.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.349 NppStatus nppsSet_64sc (Npp64sc *nValue*, Npp64sc ∗ *pDst*, int *nLength*)

64-bit long long integer complex, vector set method.

Parameters:

nValue Value used to initialize the vector pDst.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.350 NppStatus nppsSet_8u (Npp8u *nValue*, Npp8u ∗ *pDst*, int *nLength*)

8-bit unsigned char, vector set method.

Parameters:

nValue Value used to initialize the vector pDst.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.351 NppStatus nppsSqr_16s_ISfs (Npp16s ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short signal squared, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.352 NppStatus nppsSqr_16s_Sfs (const Npp16s ∗ *pSrc*, Npp16s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short signal squared, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.353 NppStatus nppsSqr_16sc_ISfs (Npp16sc ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit complex signed short signal squared, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.354 NppStatus nppsSqr_16sc_Sfs (const Npp16sc ∗ *pSrc*, Npp16sc ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit complex signed short signal squared, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.355 NppStatus nppsSqr_16u_ISfs (Npp16u ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short signal squared, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.356 NppStatus nppsSqr_16u_Sfs (const Npp16u ∗ *pSrc*, Npp16u ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short signal squared, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.
pDst Destination Signal Pointer.
nLength Signal Length.
nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.357 NppStatus nppsSqr_32f (const Npp32f ∗ *pSrc*, Npp32f ∗ *pDst*, int *nLength*)

32-bit floating point signal squared.

Parameters:

pSrc Source Signal Pointer.
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.358 NppStatus nppsSqr_32f_I (Npp32f ∗ *pSrcDst*, int *nLength*)

32-bit floating point signal squared.

Parameters:

pSrcDst In-Place Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.359 NppStatus nppsSqr_32fc (const Npp32fc ∗ *pSrc*, Npp32fc ∗ *pDst*, int *nLength*)

32-bit complex floating point signal squared.

Parameters:

pSrc Source Signal Pointer.
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.360 NppStatus nppsSqr_32fc_I (Npp32fc * *pSrcDst*, int *nLength*)

32-bit complex floating point signal squared.

Parameters:

pSrcDst In-Place Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.361 NppStatus nppsSqr_64f (const Npp64f * *pSrc*, Npp64f * *pDst*, int *nLength*)

64-bit floating point signal squared.

Parameters:

pSrc Source Signal Pointer.
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.362 NppStatus nppsSqr_64f_I (Npp64f * *pSrcDst*, int *nLength*)

64-bit floating point signal squared.

Parameters:

pSrcDst In-Place Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.363 NppStatus nppsSqr_64fc (const Npp64fc * *pSrc*, Npp64fc * *pDst*, int *nLength*)

64-bit complex floating point signal squared.

Parameters:

pSrc Source Signal Pointer.
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.364 NppStatus nppsSqr_64fc_I (Npp64fc ∗ *pSrcDst*, int *nLength*)

64-bit complex floating point signal squared.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.365 NppStatus nppsSqr_8u_ISfs (Npp8u ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned char signal squared, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.366 NppStatus nppsSqr_8u_Sfs (const Npp8u ∗ *pSrc*, Npp8u ∗ *pDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned char signal squared, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.367 NppStatus nppsSqrt_16s_ISfs (Npp16s ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short signal square root, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.368 NppStatus nppsSqrt_16s_Sfs (const Npp16s * *pSrc*, Npp16s * *pDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short signal square root, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.369 NppStatus nppsSqrt_16sc_ISfs (Npp16sc * *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit complex signed short signal square root, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.370 NppStatus nppsSqrt_16sc_Sfs (const Npp16sc * *pSrc*, Npp16sc * *pDst*, int *nLength*, int *nScaleFactor*)

16-bit complex signed short signal square root, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.371 NppStatus nppsSqrt_16u_ISfs (Npp16u ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short signal square root, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.
nLength Signal Length.
nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.372 NppStatus nppsSqrt_16u_Sfs (const Npp16u ∗ *pSrc*, Npp16u ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short signal square root, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.
pDst Destination Signal Pointer.
nLength Signal Length.
nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.373 NppStatus nppsSqrt_32f (const Npp32f ∗ *pSrc*, Npp32f ∗ *pDst*, int *nLength*)

32-bit floating point signal square root.

Parameters:

pSrc Source Signal Pointer.
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.374 NppStatus nppsSqrt_32f_I (Npp32f ∗ *pSrcDst*, int *nLength*)

32-bit floating point signal square root.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.375 NppStatus nppsSqrt_32fc (const Npp32fc ∗ *pSrc*, Npp32fc ∗ *pDst*, int *nLength*)

32-bit complex floating point signal square root.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.376 NppStatus nppsSqrt_32fc_I (Npp32fc ∗ *pSrcDst*, int *nLength*)

32-bit complex floating point signal square root.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.377 NppStatus nppsSqrt_32s16s_Sfs (const Npp32s ∗ *pSrc*, Npp16s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer signal square root, scale, then clamp to 16-bit signed integer saturated value.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.378 NppStatus nppsSqrt_64f (const Npp64f ∗ *pSrc*, Npp64f ∗ *pDst*, int *nLength*)

64-bit floating point signal square root.

Parameters:

pSrc Source Signal Pointer.
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.379 NppStatus nppsSqrt_64f_I (Npp64f ∗ *pSrcDst*, int *nLength*)

64-bit floating point signal square root.

Parameters:

pSrcDst In-Place Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.380 NppStatus nppsSqrt_64fc (const Npp64fc ∗ *pSrc*, Npp64fc ∗ *pDst*, int *nLength*)

64-bit complex floating point signal square root.

Parameters:

pSrc Source Signal Pointer.
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.381 NppStatus nppsSqrt_64fc_I (Npp64fc ∗ *pSrcDst*, int *nLength*)

64-bit complex floating point signal square root.

Parameters:

pSrcDst In-Place Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.382 NppStatus nppsSqrt_64s16s_Sfs (const Npp64s ∗ *pSrc*, Npp16s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

64-bit signed integer signal square root, scale, then clamp to 16-bit signed integer saturated value.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.383 NppStatus nppsSqrt_64s_ISfs (Npp64s ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

64-bit signed integer signal square root, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.384 NppStatus nppsSqrt_64s_Sfs (const Npp64s ∗ *pSrc*, Npp64s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

64-bit signed integer signal square root, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.385 NppStatus nppsSqrt_8u_ISfs (Npp8u * *pSrcDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned char signal square root, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.386 NppStatus nppsSqrt_8u_Sfs (const Npp8u * *pSrc*, Npp8u * *pDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned char signal square root, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.387 NppStatus nppsSub_16s (const Npp16s * *pSrc1*, const Npp16s * *pSrc2*, Npp16s * *pDst*, int *nLength*)

16-bit signed short signal subtract signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal1 elements to be subtracted from signal2 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.388 NppStatus nppsSub_16s32f (const Npp16s $*$ *pSrc1*, const Npp16s $*$ *pSrc2*, Npp32f $*$ *pDst*, int *nLength*)

16-bit signed short signal subtract 16-bit signed short signal, then clamp and convert to 32-bit floating point saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal1 elements to be subtracted from signal2 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.389 NppStatus nppsSub_16s_I (const Npp16s $*$ *pSrc*, Npp16s $*$ *pSrcDst*, int *nLength*)

16-bit signed short in place signal subtract signal, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 elements to be subtracted from signal2 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.390 NppStatus nppsSub_16s_ISfs (const Npp16s $*$ *pSrc*, Npp16s $*$ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short in place signal subtract signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 elements to be subtracted from signal2 elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.391 NppStatus nppsSub_16s_Sfs (const Npp16s ∗ *pSrc1*, const Npp16s ∗ *pSrc2*, Npp16s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short signal subtract signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal1 elements to be subtracted from signal2 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.392 NppStatus nppsSub_16sc_ISfs (const Npp16sc ∗ *pSrc*, Npp16sc ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit complex signed short in place signal subtract signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 elements to be subtracted from signal2 elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.393 NppStatus nppsSub_16sc_Sfs (const Npp16sc ∗ *pSrc1*, const Npp16sc ∗ *pSrc2*, Npp16sc ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit signed complex short signal subtract signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal1 elements to be subtracted from signal2 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.394 NppStatus nppsSub_16u_ISfs (const Npp16u ∗ *pSrc*, Npp16u ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short in place signal subtract signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 elements to be subtracted from signal2 elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.395 NppStatus nppsSub_16u_Sfs (const Npp16u ∗ *pSrc1*, const Npp16u ∗ *pSrc2*, Npp16u ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short signal subtract signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal1 elements to be subtracted from signal2 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.396 NppStatus nppsSub_32f (const Npp32f ∗ *pSrc1*, const Npp32f ∗ *pSrc2*, Npp32f ∗ *pDst*, int *nLength*)

32-bit floating point signal subtract signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal1 elements to be subtracted from signal2 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.397 NppStatus nppsSub_32f_I (const Npp32f * *pSrc*, Npp32f * *pSrcDst*, int *nLength*)

32-bit floating point in place signal subtract signal, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 elements to be subtracted from signal2 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.398 NppStatus nppsSub_32fc (const Npp32fc * *pSrc1*, const Npp32fc * *pSrc2*, Npp32fc * *pDst*, int *nLength*)

32-bit complex floating point signal subtract signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal1 elements to be subtracted from signal2 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.399 NppStatus nppsSub_32fc_I (const Npp32fc * *pSrc*, Npp32fc * *pSrcDst*, int *nLength*)

32-bit complex floating point in place signal subtract signal, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 elements to be subtracted from signal2 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.400 NppStatus nppsSub_32s_ISfs (const Npp32s * *pSrc*, Npp32s * *pSrcDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer in place signal subtract signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 elements to be subtracted from signal2 elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.401 NppStatus nppsSub_32s_Sfs (const Npp32s ∗ *pSrc1*, const Npp32s ∗ *pSrc2*, Npp32s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer signal subtract signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal1 elements to be subtracted from signal2 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.402 NppStatus nppsSub_32sc_ISfs (const Npp32sc ∗ *pSrc*, Npp32sc ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

32-bit complex signed integer in place signal subtract signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 elements to be subtracted from signal2 elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.403 NppStatus nppsSub_32sc_Sfs (const Npp32sc ∗ *pSrc1*, const Npp32sc ∗ *pSrc2*, Npp32sc ∗ *pDst*, int *nLength*, int *nScaleFactor*)

32-bit signed complex integer signal subtract signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal1 elements to be subtracted from signal2 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.404 NppStatus nppsSub_64f (const Npp64f ∗ *pSrc1*, const Npp64f ∗ *pSrc2*, Npp64f ∗ *pDst*, int *nLength*)

64-bit floating point signal subtract signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal1 elements to be subtracted from signal2 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.405 NppStatus nppsSub_64f_I (const Npp64f ∗ *pSrc*, Npp64f ∗ *pSrcDst*, int *nLength*)

64-bit floating point in place signal subtract signal, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 elements to be subtracted from signal2 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.406 NppStatus nppsSub_64fc (const Npp64fc * *pSrc1*, const Npp64fc * *pSrc2*, Npp64fc * *pDst*, int *nLength*)

64-bit complex floating point signal subtract signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal1 elements to be subtracted from signal2 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.407 NppStatus nppsSub_64fc_I (const Npp64fc * *pSrc*, Npp64fc * *pSrcDst*, int *nLength*)

64-bit complex floating point in place signal subtract signal, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 elements to be subtracted from signal2 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.408 NppStatus nppsSub_8u_ISfs (const Npp8u * *pSrc*, Npp8u * *pSrcDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned char in place signal subtract signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 elements to be subtracted from signal2 elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.409 NppStatus nppsSub_8u_Sfs (const Npp8u ∗ *pSrc1*, const Npp8u ∗ *pSrc2*, Npp8u ∗ *pDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned char signal subtract signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal1 elements to be subtracted from signal2 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.410 NppStatus nppsSubC_16s_ISfs (Npp16s *nValue*, Npp16s ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short in place signal subtract constant, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be subtracted from each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.411 NppStatus nppsSubC_16s_Sfs (const Npp16s ∗ *pSrc*, Npp16s *nValue*, Npp16s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short signal subtract constant, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be subtracted from each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.412 NppStatus nppsSubC_16sc_ISfs (Npp16sc *nValue*, Npp16sc * *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit integer complex number (16 bit real, 16 bit imaginary) signal subtract constant, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be subtracted from each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.413 NppStatus nppsSubC_16sc_Sfs (const Npp16sc * *pSrc*, Npp16sc *nValue*, Npp16sc * *pDst*, int *nLength*, int *nScaleFactor*)

16-bit integer complex number (16 bit real, 16 bit imaginary) signal subtract constant, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be subtracted from each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.414 NppStatus nppsSubC_16u_ISfs (Npp16u *nValue*, Npp16u * *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short in place signal subtract constant, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be subtracted from each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.415 NppStatus nppsSubC_16u_Sfs (const Npp16u ∗ *pSrc*, Npp16u *nValue*, Npp16u ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short signal subtract constant, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be subtracted from each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.416 NppStatus nppsSubC_32f (const Npp32f ∗ *pSrc*, Npp32f *nValue*, Npp32f ∗ *pDst*, int *nLength*)

32-bit floating point signal subtract constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be subtracted from each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.417 NppStatus nppsSubC_32f_I (Npp32f *nValue*, Npp32f ∗ *pSrcDst*, int *nLength*)

32-bit floating point in place signal subtract constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be subtracted from each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.418 NppStatus nppsSubC_32fc (const Npp32fc ∗ *pSrc*, Npp32fc *nValue*, Npp32fc ∗ *pDst*, int *nLength*)

32-bit floating point complex number (32 bit real, 32 bit imaginary) signal subtract constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be subtracted from each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.419 NppStatus nppsSubC_32fc_I (Npp32fc *nValue*, Npp32fc ∗ *pSrcDst*, int *nLength*)

32-bit floating point complex number (32 bit real, 32 bit imaginary) in place signal subtract constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be subtracted from each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.420 NppStatus nppsSubC_32s_ISfs (Npp32s *nValue*, Npp32s ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer in place signal subtract constant and scale.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be subtracted from each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.421 NppStatus nppsSubC_32s_Sfs (const Npp32s ∗ *pSrc*, Npp32s *nValue*, Npp32s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer signal subtract constant and scale.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be subtracted from each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.422 NppStatus nppsSubC_32sc_ISfs (Npp32sc *nValue*, Npp32sc ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

32-bit integer complex number (32 bit real, 32 bit imaginary) in place signal subtract constant and scale.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be subtracted from each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.423 NppStatus nppsSubC_32sc_Sfs (const Npp32sc ∗ *pSrc*, Npp32sc *nValue*, Npp32sc ∗ *pDst*, int *nLength*, int *nScaleFactor*)

32-bit integer complex number (32 bit real, 32 bit imaginary) signal subtract constant and scale.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be subtracted from each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.424 NppStatus nppsSubC_64f (const Npp64f ∗ *pSrc*, Npp64f *nValue*, Npp64f ∗ *pDst*, int *nLength*)

64-bit floating point signal subtract constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be subtracted from each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.425 NppStatus nppsSubC_64f_I (Npp64f *nValue*, Npp64f ∗ *pSrcDst*, int *nLength*)

64-bit floating point, in place signal subtract constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be subtracted from each vector element

nLength Length of the vectors, number of items.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.426 NppStatus nppsSubC_64fc (const Npp64fc ∗ *pSrc*, Npp64fc *nValue*, Npp64fc ∗ *pDst*, int *nLength*)

64-bit floating point complex number (64 bit real, 64 bit imaginary) signal subtract constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be subtracted from each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.427 NppStatus nppsSubC_64fc_I (Npp64fc *nValue*, Npp64fc ∗ *pSrcDst*, int *nLength*)

64-bit floating point complex number (64 bit real, 64 bit imaginary) in place signal subtract constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be subtracted from each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.428 NppStatus nppsSubC_8u_ISfs (Npp8u *nValue*, Npp8u ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned char in place signal subtract constant, scale, then clamp to saturated value

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be subtracted from each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.429 NppStatus nppsSubC_8u_Sfs (const Npp8u ∗ *pSrc*, Npp8u *nValue*, Npp8u ∗ *pDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned char signal subtract constant, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be subtracted from each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.430 NppStatus nppsSubCRev_16s_ISfs (Npp16s *nValue*, Npp16s * *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short in place signal subtract from constant, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value each vector element is to be subtracted from

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.431 NppStatus nppsSubCRev_16s_Sfs (const Npp16s * *pSrc*, Npp16s *nValue*, Npp16s * *pDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short signal subtract from constant, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value each vector element is to be subtracted from

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.432 NppStatus nppsSubCRev_16sc_ISfs (Npp16sc *nValue*, Npp16sc * *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit integer complex number (16 bit real, 16 bit imaginary) signal subtract from constant, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value each vector element is to be subtracted from

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.433 NppStatus nppsSubCRev_16sc_Sfs (const Npp16sc ∗ *pSrc*, Npp16sc *nValue*, Npp16sc ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit integer complex number (16 bit real, 16 bit imaginary) signal subtract from constant, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value each vector element is to be subtracted from

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.434 NppStatus nppsSubCRev_16u_ISfs (Npp16u *nValue*, Npp16u ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short in place signal subtract from constant, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value each vector element is to be subtracted from

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.435 NppStatus nppsSubCRev_16u_Sfs (const Npp16u ∗ *pSrc*, Npp16u *nValue*, Npp16u ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short signal subtract from constant, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value each vector element is to be subtracted from

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.436 NppStatus nppsSubCRev_32f (const Npp32f ∗ *pSrc*, Npp32f *nValue*, Npp32f ∗ *pDst*, int *nLength*)

32-bit floating point signal subtract from constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value each vector element is to be subtracted from

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.437 NppStatus nppsSubCRev_32f_I (Npp32f *nValue*, Npp32f ∗ *pSrcDst*, int *nLength*)

32-bit floating point in place signal subtract from constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value each vector element is to be subtracted from

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.438 NppStatus nppsSubCRev_32fc (const Npp32fc ∗ *pSrc*, Npp32fc *nValue*, Npp32fc ∗ *pDst*, int *nLength*)

32-bit floating point complex number (32 bit real, 32 bit imaginary) signal subtract from constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value each vector element is to be subtracted from

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.439 NppStatus nppsSubCRev_32fc_I (Npp32fc *nValue*, Npp32fc ∗ *pSrcDst*, int *nLength*)

32-bit floating point complex number (32 bit real, 32 bit imaginary) in place signal subtract from constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value each vector element is to be subtracted from

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.440 NppStatus nppsSubCRev_32s_ISfs (Npp32s *nValue*, Npp32s ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer in place signal subtract from constant and scale.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value each vector element is to be subtracted from

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.441 NppStatus nppsSubCRev_32s_Sfs (const Npp32s ∗ *pSrc*, Npp32s *nValue*, Npp32s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integersignal subtract from constant and scale.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value each vector element is to be subtracted from

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.442 NppStatus nppsSubCRev_32sc_ISfs (Npp32sc *nValue*, Npp32sc ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

32-bit integer complex number (32 bit real, 32 bit imaginary) in place signal subtract from constant and scale.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value each vector element is to be subtracted from

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.443 NppStatus nppsSubCRev_32sc_Sfs (const Npp32sc ∗ *pSrc*, Npp32sc *nValue*, Npp32sc ∗ *pDst*, int *nLength*, int *nScaleFactor*)

32-bit integer complex number (32 bit real, 32 bit imaginary) signal subtract from constant and scale.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value each vector element is to be subtracted from

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.444 NppStatus nppsSubCRev_64f (const Npp64f ∗ *pSrc*, Npp64f *nValue*, Npp64f ∗ *pDst*, int *nLength*)

64-bit floating point signal subtract from constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value each vector element is to be subtracted from

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.445 NppStatus nppsSubCRev_64f_I (Npp64f *nValue*, Npp64f ∗ *pSrcDst*, int *nLength*)

64-bit floating point, in place signal subtract from constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value each vector element is to be subtracted from

nLength Length of the vectors, number of items.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.446 NppStatus nppsSubCRev_64fc (const Npp64fc ∗ *pSrc*, Npp64fc *nValue*, Npp64fc ∗ *pDst*, int *nLength*)

64-bit floating point complex number (64 bit real, 64 bit imaginary) signal subtract from constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value each vector element is to be subtracted from

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.447 NppStatus nppsSubCRev_64fc_I (Npp64fc *nValue*, Npp64fc ∗ *pSrcDst*, int *nLength*)

64-bit floating point complex number (64 bit real, 64 bit imaginary) in place signal subtract from constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value each vector element is to be subtracted from

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.448 NppStatus nppsSubCRev_8u_ISfs (Npp8u *nValue*, Npp8u ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned char in place signal subtract from constant, scale, then clamp to saturated value

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value each vector element is to be subtracted from

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.449 NppStatus nppsSubCRev_8u_Sfs (const Npp8u ∗ *pSrc*, Npp8u *nValue*, Npp8u ∗ *pDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned char signal subtract from constant, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value each vector element is to be subtracted from

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.450 NppStatus nppsSum_16s32s_Sfs (const Npp16s ∗ *pSrc*, int *nLength*, Npp32s ∗ *pSum*, int *nScaleFactor*, Npp8u ∗ *pDeviceBuffer*)

16-bit integer vector sum (32bit) with integer scaling method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pSum Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation.

nScaleFactor Integer-result scale factor.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.451 NppStatus nppsSum_16s_Sfs (const Npp16s ∗ *pSrc*, int *nLength*, Npp16s ∗ *pSum*, int *nScaleFactor*, Npp8u ∗ *pDeviceBuffer*)

16-bit short vector sum with integer scaling method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pSum Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation.

nScaleFactor Integer-result scale factor.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.452 NppStatus nppsSum_16sc32sc_Sfs (const Npp16sc ∗ *pSrc*, int *nLength*, Npp32sc ∗ *pSum*, int *nScaleFactor*, Npp8u ∗ *pDeviceBuffer*)

16-bit short complex vector sum (32bit int complex) with integer scaling method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pSum Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation.

nScaleFactor Integer-result scale factor.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.453 NppStatus nppsSum_16sc_Sfs (const Npp16sc ∗ *pSrc*, int *nLength*, Npp16sc ∗ *pSum*, int *nScaleFactor*, Npp8u ∗ *pDeviceBuffer*)

16-bit short complex vector sum with integer scaling method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pSum Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation.

nScaleFactor Integer-result scale factor.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.454 NppStatus nppsSum_32f (const Npp32f ∗ *pSrc*, int *nLength*, Npp32f ∗ *pSum*, NppHintAlgorithm *eHint*, Npp8u ∗ *pDeviceBuffer*)

32-bit float vector sum method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pSum Pointer to the output result.

eHint Suggests using specific code.

pDeviceBuffer Pointer to the required device memory allocation.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.455 NppStatus nppsSum_32fc (const Npp32fc ∗ *pSrc*, int *nLength*, Npp32fc ∗ *pSum*, NppHintAlgorithm *eHint*, Npp8u ∗ *pDeviceBuffer*)

32-bit float complex vector sum method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pSum Pointer to the output result.

eHint Suggests using specific code.

pDeviceBuffer Pointer to the required device memory allocation.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.456 NppStatus nppsSum_32s_Sfs (const Npp32s ∗ *pSrc*, int *nLength*, Npp32s ∗ *pSum*, int *nScaleFactor*, Npp8u ∗ *pDeviceBuffer*)

32-bit integer vector sum with integer scaling method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pSum Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation.

nScaleFactor Integer-result scale factor.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.457 NppStatus nppsSum_64f (const Npp64f ∗ *pSrc*, int *nLength*, Npp64f ∗ *pSum*, Npp8u ∗ *pDeviceBuffer*)

64-bit double vector sum method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pSum Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.458 NppStatus nppsSum_64fc (const Npp64fc ∗ *pSrc*, int *nLength*, Npp64fc ∗ *pSum*, Npp8u ∗ *pDeviceBuffer*)

64-bit double complex vector sum method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pSum Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.459 NppStatus nppsXor_16u (const Npp16u ∗ *pSrc1*, const Npp16u ∗ *pSrc2*, Npp16u ∗ *pDst*, int *nLength*)

16-bit unsigned short signal exclusive or with signal.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be exclusive ored with signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.460 NppStatus nppsXor_16u_I (const Npp16u * *pSrc*, Npp16u * *pSrcDst*, int *nLength*)

16-bit unsigned short in place signal exclusive or with signal.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be exclusive ored with signal1 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.461 NppStatus nppsXor_32u (const Npp32u * *pSrc1*, const Npp32u * *pSrc2*, Npp32u * *pDst*, int *nLength*)

32-bit unsigned integer signal exclusive or with signal.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be exclusive ored with signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.462 NppStatus nppsXor_32u_I (const Npp32u * *pSrc*, Npp32u * *pSrcDst*, int *nLength*)

32-bit unsigned integer in place signal exclusive or with signal.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be exclusive ored with signal1 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.463 NppStatus nppsXor_8u (const Npp8u * *pSrc1*, const Npp8u * *pSrc2*, Npp8u * *pDst*, int *nLength*)

8-bit unsigned char signal exclusive or with signal.

Parameters:

pSrc1 Source Signal Pointer.
pSrc2 Source Signal Pointer. signal2 elements to be exclusive ored with signal1 elements
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.464 NppStatus nppsXor_8u_I (const Npp8u ∗ *pSrc*, Npp8u ∗ *pSrcDst*, int *nLength*)

8-bit unsigned char in place signal exclusive or with signal.

Parameters:

pSrc Source Signal Pointer.
pSrcDst In-Place Signal Pointer. signal2 elements to be exclusive ored with signal1 elements
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.465 NppStatus nppsXorC_16u (const Npp16u ∗ *pSrc*, Npp16u *nValue*, Npp16u ∗ *pDst*, int *nLength*)

16-bit unsigned short signal exclusive or with constant.

Parameters:

pSrc Source Signal Pointer.
nValue Constant value to be exclusive ored with each vector element
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.466 NppStatus nppsXorC_16u_I (Npp16u *nValue*, Npp16u ∗ *pSrcDst*, int *nLength*)

16-bit unsigned short in place signal exclusive or with constant.

Parameters:

pSrcDst In-Place Signal Pointer.
nValue Constant value to be exclusive ored with each vector element
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.467 NppStatus nppsXorC_32u (const Npp32u * *pSrc*, Npp32u *nValue*, Npp32u * *pDst*, int *nLength*)

32-bit unsigned integer signal exclusive or with constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be exclusive ored with each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.468 NppStatus nppsXorC_32u_I (Npp32u *nValue*, Npp32u * *pSrcDst*, int *nLength*)

32-bit unsigned signed integer in place signal exclusive or with constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be exclusive ored with each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.469 NppStatus nppsXorC_8u (const Npp8u * *pSrc*, Npp8u *nValue*, Npp8u * *pDst*, int *nLength*)

8-bit unsigned char signal exclusive or with constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be exclusive ored with each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.470 NppStatus nppsXorC_8u_I (Npp8u *nValue*, Npp8u ∗ *pSrcDst*, int *nLength*)

8-bit unsigned char in place signal exclusive or with constant.

Parameters:

pSrcDst In-Place Signal Pointer.
nValue Constant value to be exclusive ored with each vector element
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.471 NppStatus nppsZero_16s (Npp16s ∗ *pDst*, int *nLength*)

16-bit integer, vector zero method.

Parameters:

pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.472 NppStatus nppsZero_16sc (Npp16sc ∗ *pDst*, int *nLength*)

16-bit integer complex, vector zero method.

Parameters:

pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.473 NppStatus nppsZero_32f (Npp32f ∗ *pDst*, int *nLength*)

32-bit float, vector zero method.

Parameters:

pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.474 NppStatus nppsZero_32fc (Npp32fc $*$ *pDst*, int *nLength*)

32-bit float complex, vector zero method.

Parameters:

pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.475 NppStatus nppsZero_32s (Npp32s $*$ *pDst*, int *nLength*)

32-bit integer, vector zero method.

Parameters:

pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.476 NppStatus nppsZero_32sc (Npp32sc $*$ *pDst*, int *nLength*)

32-bit integer complex, vector zero method.

Parameters:

pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.477 NppStatus nppsZero_64f (Npp64f $*$ *pDst*, int *nLength*)

64-bit double, vector zero method.

Parameters:

pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.478 NppStatus nppsZero_64fc (Npp64fc ∗ *pDst*, int *nLength*)

64-bit double complex, vector zero method.

Parameters:

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.479 NppStatus nppsZero_64s (Npp64s ∗ *pDst*, int *nLength*)

64-bit long long integer, vector zero method.

Parameters:

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.480 NppStatus nppsZero_64sc (Npp64sc ∗ *pDst*, int *nLength*)

64-bit long long integer complex, vector zero method.

Parameters:

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.5.1.481 NppStatus nppsZero_8u (Npp8u ∗ *pDst*, int *nLength*)

8-bit unsigned char, vector zero method.

Parameters:

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

Chapter 8

Data Structure Documentation

8.1 Npp16sc Struct Reference

Complex Number This struct represents a short complex number.

```
#include <nppdefs.h>
```

Data Fields

- Npp16s re

  Real part.

- Npp16s im

  Imaginary part.

8.1.1 Detailed Description

Complex Number This struct represents a short complex number.

8.1.2 Field Documentation

8.1.2.1 Npp16s Npp16sc::im

Imaginary part.

8.1.2.2 Npp16s Npp16sc::re

Real part.

The documentation for this struct was generated from the following file:

- C:/Perforce/sw/rel/gpgpu/toolkit/r4.0/NPP/npp/include/nppdefs.h

8.2 Npp32fc Struct Reference

Complex Number This struct represents a single floating-point complex number.

```
#include <nppdefs.h>
```

Data Fields

- Npp32f re

  Real part.
- Npp32f im

  Imaginary part.

8.2.1 Detailed Description

Complex Number This struct represents a single floating-point complex number.

8.2.2 Field Documentation

8.2.2.1 Npp32f Npp32fc::im

Imaginary part.

8.2.2.2 Npp32f Npp32fc::re

Real part.

The documentation for this struct was generated from the following file:

- C:/Perforce/sw/rel/gpgpu/toolkit/r4.0/NPP/npp/include/nppdefs.h

8.3 Npp32sc Struct Reference

Complex Number This struct represents a signed int complex number.

```
#include <nppdefs.h>
```

Data Fields

- Npp32s re

  Real part.

- Npp32s im

  Imaginary part.

8.3.1 Detailed Description

Complex Number This struct represents a signed int complex number.

8.3.2 Field Documentation

8.3.2.1 Npp32s Npp32sc::im

Imaginary part.

8.3.2.2 Npp32s Npp32sc::re

Real part.

The documentation for this struct was generated from the following file:

- C:/Perforce/sw/rel/gpgpu/toolkit/r4.0/NPP/npp/include/nppdefs.h

8.4 Npp64fc Struct Reference

Complex Number This struct represents a double floating-point complex number.

```
#include <nppdefs.h>
```

Data Fields

- Npp64f re

  Real part.

- Npp64f im

  Imaginary part.

8.4.1 Detailed Description

Complex Number This struct represents a double floating-point complex number.

8.4.2 Field Documentation

8.4.2.1 Npp64f Npp64fc::im

Imaginary part.

8.4.2.2 Npp64f Npp64fc::re

Real part.

The documentation for this struct was generated from the following file:

- C:/Perforce/sw/rel/gpgpu/toolkit/r4.0/NPP/npp/include/nppdefs.h

8.5 Npp64sc Struct Reference

Complex Number This struct represents a long long complex number.

```
#include <nppdefs.h>
```

Data Fields

- Npp64s re

  Real part.

- Npp64s im

  Imaginary part.

8.5.1 Detailed Description

Complex Number This struct represents a long long complex number.

8.5.2 Field Documentation

8.5.2.1 Npp64s Npp64sc::im

Imaginary part.

8.5.2.2 Npp64s Npp64sc::re

Real part.

The documentation for this struct was generated from the following file:

- C:/Perforce/sw/rel/gpgpu/toolkit/r4.0/NPP/npp/include/nppdefs.h

8.6 NppiHaarBuffer Struct Reference

```
#include <nppdefs.h>
```

Data Fields

- int haarBufferSize

  size of the buffer
- Npp32s ∗ haarBuffer

  buffer

8.6.1 Field Documentation

8.6.1.1 Npp32s∗ NppiHaarBuffer::haarBuffer

buffer

8.6.1.2 int NppiHaarBuffer::haarBufferSize

size of the buffer

The documentation for this struct was generated from the following file:

- C:/Perforce/sw/rel/gpgpu/toolkit/r4.0/NPP/npp/include/nppdefs.h

8.7 NppiHaarClassifier_32f Struct Reference

```
#include <nppdefs.h>
```

Data Fields

- int numClassifiers

  number of classifiers

- Npp32s ∗ classifiers

  packed classifier data 40 bytes each

- size_t classifierStep
- NppiSize classifierSize
- Npp32s ∗ counterDevice

8.7.1 Field Documentation

8.7.1.1 Npp32s∗ NppiHaarClassifier_32f::classifiers

packed classifier data 40 bytes each

8.7.1.2 NppiSize NppiHaarClassifier_32f::classifierSize

8.7.1.3 size_t NppiHaarClassifier_32f::classifierStep

8.7.1.4 Npp32s∗ NppiHaarClassifier_32f::counterDevice

8.7.1.5 int NppiHaarClassifier_32f::numClassifiers

number of classifiers

The documentation for this struct was generated from the following file:

- C:/Perforce/sw/rel/gpgpu/toolkit/r4.0/NPP/npp/include/nppdefs.h

8.8 NppiPoint Struct Reference

2D Point

```
#include <nppdefs.h>
```

Data Fields

- int x

  x-coordinate.

- int y

  y-coordinate.

8.8.1 Detailed Description

2D Point

8.8.2 Field Documentation

8.8.2.1 int NppiPoint::x

x-coordinate.

8.8.2.2 int NppiPoint::y

y-coordinate.

The documentation for this struct was generated from the following file:

- C:/Perforce/sw/rel/gpgpu/toolkit/r4.0/NPP/npp/include/nppdefs.h

8.9 NppiRect Struct Reference

2D Rectangle This struct contains position and size information of a rectangle in two space.

```
#include <nppdefs.h>
```

Data Fields

- int x

  x-coordinate of upper left corner.

- int y

  y-coordinate of upper left corner.

- int width

  Rectangle width.

- int height

  Rectangle height.

8.9.1 Detailed Description

2D Rectangle This struct contains position and size information of a rectangle in two space.

The rectangle's position is usually signified by the coordinate of its upper-left corner.

8.9.2 Field Documentation

8.9.2.1 int NppiRect::height

Rectangle height.

8.9.2.2 int NppiRect::width

Rectangle width.

8.9.2.3 int NppiRect::x

x-coordinate of upper left corner.

8.9.2.4 int NppiRect::y

y-coordinate of upper left corner.

The documentation for this struct was generated from the following file:

- C:/Perforce/sw/rel/gpgpu/toolkit/r4.0/NPP/npp/include/nppdefs.h

8.10 NppiSize Struct Reference

2D Size This struct typically represents the size of a a rectangular region in two space.

```
#include <nppdefs.h>
```

Data Fields

- int width

  Rectangle width.

- int height

  Rectangle height.

8.10.1 Detailed Description

2D Size This struct typically represents the size of a a rectangular region in two space.

8.10.2 Field Documentation

8.10.2.1 int NppiSize::height

Rectangle height.

8.10.2.2 int NppiSize::width

Rectangle width.

The documentation for this struct was generated from the following file:

- C:/Perforce/sw/rel/gpgpu/toolkit/r4.0/NPP/npp/include/nppdefs.h

8.11 NppLibraryVersion Struct Reference

```
#include <nppdefs.h>
```

Data Fields

- int major

  Major version number.

- int minor

  Minor version number.

- int build

  Build number. This reflects the nightly build this release was made from.

8.11.1 Field Documentation

8.11.1.1 int NppLibraryVersion::build

Build number. This reflects the nightly build this release was made from.

8.11.1.2 int NppLibraryVersion::major

Major version number.

8.11.1.3 int NppLibraryVersion::minor

Minor version number.

The documentation for this struct was generated from the following file:

- C:/Perforce/sw/rel/gpgpu/toolkit/r4.0/NPP/npp/include/nppdefs.h