Function

vImageHistogramSpecification_ARGB8888(_:_:_:_:)

Specifies the histogram of an 8-bit-per-channel, 4-channel interleaved buffer.

iOS 5.0+ | iPadOS 5.0+ | Mac Catalyst 13.1+ | macOS 10.3+ | tvOS 5.0+ | visionOS 1.0+ | watchOS 1.0+

```
func vImageHistogramSpecification_ARGB8888(
    _ src: UnsafePointer<vImage_Buffer>,
    _ dest: UnsafePointer<vImage_Buffer>,
    _ desired_histogram: UnsafeMutablePointer<UnsafePointer<vImagePixel
Count>?>,
    _ flags: vImage_Flags
) -> vImage_Error
```

Parameters

src
The source vImage buffer.

dest
A pointer to the destination vImage buffer structure. You're responsible for filling out the height, width, and rowBytes fields of this structure, and for allocating a data buffer of the appropriate size. On return, the data buffer this structure points to contains the destination image data. When you no longer need the data buffer, deallocate the memory to prevent memory leaks.

desired_histogram
The histograms that the operation applies to the source buffer.

`flags`

The options to use when performing the operation. If your code implements its own tiling or its own multithreading, pass `kvImageDoNotTile`; otherwise, pass `kvImageNoFlags`.

To specify that the function doesn't apply the operation to the alpha channel, set the `kvImageLeaveAlphaUnchanged` flag.

Return Value

`kvImageNoError`; otherwise, one of the error codes in Data Types and Constants.

Discussion

Use this function to apply a histogram — that you've generated using `vImageHistogramSpecification_ARGBFFFF(_:_:_:_:_:_:_:)` — to each channel of another image.

The following code applies a histogram to a 4-channel interleaved vImage buffer:

```
// `histogramAlpha`, `histogramRed`, `histogramGreen`, and `histogramBlue` are `entr
histogramAlpha.withUnsafeBufferPointer { zeroPtr in
    histogramRed.withUnsafeBufferPointer { onePtr in
        histogramGreen.withUnsafeBufferPointer { twoPtr in
            histogramBlue.withUnsafeBufferPointer { threePtr in

                var histogramBins = [zeroPtr.baseAddress, onePtr.baseAddress,
                                     twoPtr.baseAddress, threePtr.baseAddress]

                histogramBins.withUnsafeMutableBufferPointer { histogramBinsPtr in
                    // `buffer` is a `vImage_Buffer` structure
                    _ = vImageHistogramSpecification_ARGB8888(&buffer, &buffer,
                                                              histogramBinsPtr.baseA
                                                              vImage_Flags(kvImageNo
                }
            }
        }
    }
}
```

See Also

Related Documentation

Enhancing image contrast with histogram manipulation

Enhance and adjust the contrast of an image with histogram equalization and contrast stretching.

{} Specifying histograms with vImage

Calculate the histogram of one image, and apply it to a second image.

Specifying a histogram

```
func vImageHistogramSpecification_Planar8(UnsafePointer<vImage_Buffer>, UnsafePointer<vImage_Buffer>, UnsafePointer<vImagePixelCount>, vImage_Flags) -> vImage_Error
```

Specifies the histogram of an 8-bit planar buffer.

```
func vImageHistogramSpecification_PlanarF(UnsafePointer<vImage_Buffer>, UnsafePointer<vImage_Buffer>, UnsafeMutableRawPointer!, UnsafePointer<vImagePixelCount>, UInt32, Pixel_F, Pixel_F, vImage_Flags) -> vImage_Error
```

Specifies the histogram of a 32-bit planar buffer.

```
func vImageHistogramSpecification_ARGBFFFF(UnsafePointer<vImage_Buffer>, UnsafePointer<vImage_Buffer>, UnsafeMutableRawPointer!, UnsafeMutablePointer<UnsafePointer<vImagePixelCount>?>!, UInt32, Pixel_F, Pixel_F, vImage_Flags) -> vImage_Error
```

Specifes the histogram of a 32-bit-per-channel, 4-channel interleaved buffer.