

Accelerate / vImageMultidimensionalTable_Create(_:_:_:_:_:_:_:_:_)

Function

Creates a multidimensional lookup table.

iOS 7.0+ | iPadOS 7.0+ | Mac Catalyst 13.1+ | macOS 10.9+ | tvOS 7.0+ | visionOS 1.0+ | watchOS 1.0+

```
func vImageMultidimensionalTable_Create(  
    _ tableData: UnsafePointer<UInt16>,  
    _ numSrcChannels: UInt32,  
    _ numDestChannels: UInt32,  
    _ table_entries_per_dimension: UnsafePointer<UInt8>,  
    _ hint: vImageMDTableUsageHint,  
    _ flags: vImage_Flags,  
    _ err: UnsafeMutablePointer<vImage_Error>!  
) -> vImage_MultidimensionalTable!
```

Parameters

tableData

A pointer to the lookup table data.

`numSrcChannels`

The number of channels in an input pixel.

numDestChannels

The number of channels in an output pixel

table entries per dimension

An array that contains the number of table entries for each channel in an input pixel.

hint

A constant that specifies whether this function sets up the table for particular transform functions. If you only use `vImageMultiDimensionalInterpolatedLookupTable_PlanarF(: : : : : :)`, pass `kvImageMDTableHint_Float`. If you only use `vImageMultiDimensionalInterpolatedLookupTable_Planar16Q12(: : : : : :)`, pass `kvImageMDTableHint_16Q12`. Pass both flags if you use both transform functions.

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`.

err

On output, `kvImageNoError`; otherwise, one of the error codes in [Data Types and Constants](#).

Return Value

`kvImageNoError`; otherwise, one of the error codes in [Data Types and Constants](#).

Mentioned in

 Applying color transforms to images with a multidimensional lookup table

Discussion

Use a multidimensional lookup table to transform the colors in an image. The lookup table defines an output color based on the input color values. The vImage multidimensional lookup table provides interpolation to compute output color values that don't have an explicit entry in the table for a specified input color.

A `vImage_MultidimensionalTable` applies transforms to 32-bit and 16Q12 planar pixel buffers.

The following is an example of a simple lookup table that implements the Rec. 709 luminance coefficients to convert from a three-channel RGB image to a single-channel grayscale image. The lookup table is a 3D cube with 32 entries per channel. Supply the lookup values as a contiguous array of samples that defines the lookup table values. The samples have range $0 \dots 65535$ that the vImage library interprets as the floating-point range $0 \dots 1$.

```

let entriesPerChannel = UInt8(32)
let srcChannelCount = 3
let destChannelCount = 1

let lookupTableElementCount = Int(pow(Float(entriesPerChannel),
                                    Float(srcChannelCount))) * Int(destChannelCount)

let tableData = [UInt16](unsafeUninitializedCapacity: lookupTableElementCount) {
    buffer, count in

        let multiplier = Float(UInt16.max)
        var bufferIndex = 0

        for red in (0 ..< entriesPerChannel) {
            for green in (0 ..< entriesPerChannel) {
                for blue in (0 ..< entriesPerChannel) {

                    let normalizedRed = Float(red) / Float(entriesPerChannel - 1)
                    let normalizedGreen = Float(green) / Float(entriesPerChannel - 1)
                    let normalizedBlue = Float(blue) / Float(entriesPerChannel - 1)

                    let gray = (normalizedRed * 0.2126) +
                               (normalizedGreen * 0.7152) +
                               (normalizedBlue * 0.0722)

                    buffer[bufferIndex] = UInt16(gray * multiplier)
                    bufferIndex += 1
                }
            }
        }

        count = lookupTableElementCount
    }

let entryCountPerSourceChannel = [UInt8](repeating: entriesPerChannel,
                                         count: srcChannelCount)

var error = kvImageNoError

guard let lookupTable = vImageMultidimensionalTable_Create(
    tableData,
    UInt32(srcChannelCount),
    UInt32(destChannelCount),

```

```

    entryCountPerSourceChannel,
    kvImageMDTableHint_Float,
    vImage_Flags(kvImageNoFlags),
    &error) else {
    fatalError("Unable to create multidimensional table \((error).")
}

defer {
    vImageMultidimensionalTable_Release(lookupTable)
}

```

See Also

Transforming with a multidimensional lookup table

- 📄 Applying color transforms to images with a multidimensional lookup table
Precompute translation values to optimize color space conversion and other pointwise operations.
- {} Cropping to the subject in a chroma-keyed image
Convert a chroma-key color to alpha values and trim transparent pixels using Accelerate.
- {} Applying transformations to selected colors in an image
Desaturate a range of colors in an image with a multidimensional lookup table.

```
func vImageMultiDimensionalInterpolatedLookupTable_PlanarF(Unsafe
Pointer<vImage_Buffer>, UnsafePointer<vImage_Buffer>, UnsafeMutableRaw
Pointer!, vImage_MultidimensionalTable, vImage_InterpolationMethod, v
Image_Flags) -> vImage_Error
```

Uses a multidimensional lookup table to transform a 32-bit planar image.

```
func vImageMultiDimensionalInterpolatedLookupTable_Planar16Q12(Unsafe
Pointer<vImage_Buffer>, UnsafePointer<vImage_Buffer>, UnsafeMutableRaw
Pointer!, vImage_MultidimensionalTable, vImage_InterpolationMethod, v
Image_Flags) -> vImage_Error
```

Uses a multidimensional lookup table to transform a 16Q12 planar image.

```
func vImageMultidimensionalTable_Retain(vImage_MultidimensionalTable!)
-> vImage_Error
```

Retains a multidimensional table.

```
func vImageMultidimensionalTable_Release(vImage_MultidimensionalTable!)
```

```
-> vImage_Error
```

Releases a multidimensional table.

```
typealias vImage_MultidimensionalTable
```

An opaque pointer that represents a multidimensional lookup table.

```
struct vImageMDTableUsageHint
```

Constants that indicate the use for a multidimensional lookup table.

```
struct vImage_InterpolationMethod
```

Constants that represent different interpolation methods.