vImage Alpha Compositing Reference

Performance > Graphics & Imaging

2007-07-12

Contents

vImage Alpha Compositing Reference

Framework:	Accelerate/vImage
Companion guide	vImage Programming Guide
Declared in	Alpha.h

Overview

Alpha compositing (also known as alpha blending) is the process of layering multiple images, with the alpha value for a pixel in a given layer indicating what fraction of the colors from lower layers are seen through the color at the given level. The functions described in this reference operate on the alpha values of pixels either by blending alpha values or clipping them.

Most of the alpha compositing functions blend two input images—a top image and a bottom image—to create a composite image. The vImage framework computes the alpha values of the composite image from the alpha values of the input images. Some functions operate on interleaved formats (ARGB8888, ARGBFFFF, RGBA8888, RGBAFFFF) while others operate on planar formats. Interleaved formats contain an alpha value for each pixel, but planar formats do not. To perform alpha compositing with planar images, you need to supply the alpha information separately.

Alpha compositing functions by default perform tiling internally and may multithread internally as well. If you plan to perform your own tiling or multithreading, you must turn off vImage internal tiling and multithreading by supplying the `kvImageDoNotTile` flag as an option to the functions you use.

The vImage framework provides functions for alpha compositing for both the premultiplied alpha case and the nonpremultiplied alpha case. Mac OS X v10.4 adds some alpha compositing functions for common mixed cases. Premultiplying pixel color values by the associated alpha value results in greater computational efficiency than providing nonpremultiplied data, especially when you composite more than two images. When you use premultiplied alpha, you still need to maintain the original alpha information, so that you can retrieve the original, nonpremultiplied values of the pixels when you need them. You also need to supply the original alpha value for the bottom layer in a compositing operation.

For floating-point formats, you can multiply the color value by the alpha value directly. For integer formats in which both values are in the range of 0 to 255, you multiply the color and alpha values, then you must scale the result so that it is in the 0 to 255 range. The scaling calculation is:

```
scaledColor = (alpha * color + 127) / 255
```

Alpha compositing functions use a vImage buffer structure (`vImage_Buffer`—see *vImage Data Types and Constants Reference*) to receive and supply image data. This buffer contains a pointer to image data, the height and width (in pixels) of the image data, and the number of row bytes. You actually pass a pointer to a vImage buffer structure. You can provide a pointer to the same vImage buffer structure for one of the source images and the destination image because alpha compositing functions “work in place”. That is , the source and destination images can occupy the same memory if the they are strictly aligned pixel for pixel.

Functions by Task

Performing Nonpremultiplied Alpha Compositing

`vImageAlphaBlend_ARGBFFFF` (page 9)
: Performs nonpremultiplied alpha compositing of two ARGBFFFF images, placing the result in a destination buffer.

`vImageAlphaBlend_ARGB8888` (page 8)
: Performs nonpremultiplied alpha compositing of two ARGB8888 images, placing the result in a destination buffer.

`vImageAlphaBlend_PlanarF` (page 15)
: Performs nonpremultiplied alpha compositing of two PlanarF images, placing the result in a destination buffer.

`vImageAlphaBlend_Planar8` (page 14)
: Performs nonpremultiplied alpha compositing of two Planar8 images, placing the result in a destination buffer.

Performing Premultiplied Alpha Compositing

`vImagePremultipliedAlphaBlend_ARGBFFFF` (page 20)
: Performs premultiplied alpha compositing of two ARGBFFFF images, placing the result in a destination buffer.

`vImagePremultipliedAlphaBlend_ARGB8888` (page 20)
: Performs premultiplied alpha compositing of two ARGB8888 images, placing the result in a destination buffer.

`vImagePremultipliedAlphaBlend_PlanarF` (page 22)
: Performs premultiplied alpha compositing of two PlanarF images, placing the result in a destination buffer.

`vImagePremultipliedAlphaBlend_Planar8` (page 21)
: Performs premultiplied alpha compositing of two Planar8 images, placing the result in a destination buffer.

Performing Nonpremultiplied Alpha Compositing With a Single Alpha Value

`vImagePremultipliedConstAlphaBlend_ARGBFFFF` (page 23)
: Performs premultiplied alpha compositing of two ARGBFFFF images, using a single alpha value for the whole image and placing the result in a destination buffer.

`vImagePremultipliedConstAlphaBlend_ARGB8888` (page 23)
: Performs premultiplied alpha compositing of two ARGB8888 images, using a single alpha value for the whole image and placing the result in a destination buffer.

`vImagePremultipliedConstAlphaBlend_PlanarF` (page 25)
: Performs premultiplied alpha compositing of a two PlanarF images, using a single alpha value for the whole image and placing the result in a destination buffer.

`vImagePremultipliedConstAlphaBlend_Planar8` (page 24)
: Performs premultiplied alpha compositing of two Planar8 images, using a single alpha value for the entire image and placing the result in a destination buffer.

Performing Nonpremultiplied to Premultiplied Alpha Compositing

`vImageAlphaBlend_NonpremultipliedToPremultiplied_ARGBFFFF` (page 11)
: Performs mixed alpha compositing of a nonpremultiplied ARGBFFFF image over a premultiplied ARGBFFFF image, placing the premultiplied result in a destination buffer.

`vImageAlphaBlend_NonpremultipliedToPremultiplied_ARGB8888` (page 10)
: Performs mixed alpha compositing of a nonpremultiplied ARGB8888 image over a premultiplied ARGB8888 image, placing the premultiplied result in a destination buffer.

`vImageAlphaBlend_NonpremultipliedToPremultiplied_PlanarF` (page 13)
: Performs mixed alpha compositing of a nonpremultiplied PlanarF image over a premultiplied PlanarF image, placing the premultiplied result in a destination buffer.

`vImageAlphaBlend_NonpremultipliedToPremultiplied_Planar8` (page 12)
: Performs mixed alpha compositing of a nonpremultiplied Planar8 image over a premultiplied Planar8 image, placing the premultiplied result in a destination buffer.

Converting from Unpremultiplied to Premultiplied Format

`vImagePremultiplyData_ARGBFFFF` (page 27)
: Takes an ARGBFFFF image in nonpremultiplied alpha format and transforms it into an image in premultiplied alpha format.

`vImagePremultiplyData_RGBAFFFF` (page 30)
: Takes an RGBAFFFF image in nonpremultiplied alpha format and transforms it into an image in premultiplied alpha format.

`vImagePremultiplyData_ARGB8888` (page 26)
: Takes an ARGB8888 image in nonpremultiplied alpha format and transforms it into an image in premultiplied alpha format.

`vImagePremultiplyData_RGBA8888` (page 29)
: Takes an RGBA8888 image in nonpremultiplied alpha format and transforms it into an image in premultiplied alpha format.

`vImagePremultiplyData_PlanarF` (page 28)
: Takes a PlanarF image in nonpremultiplied alpha format, along with alpha information, and transforms it into an image in premultiplied alpha format.

`vImagePremultiplyData_Planar8` (page 28)
: Takes a Planar8 image in nonpremultiplied alpha format, along with alpha information, and transforms it into an image in premultiplied alpha format.

Converting from Premultiplied to Unpremultiplied Format

`vImageUnpremultiplyData_ARGBFFFF` (page 31)
: Takes an ARGBFFFF image in premultiplied alpha format and transforms it into an image in nonpremultiplied alpha format.

`vImageUnpremultiplyData_RGBAFFFF` (page 34)
: Takes an RGBAFFFF image in premultiplied alpha format and transforms it into an image in nonpremultiplied alpha format.

`vImageUnpremultiplyData_ARGB8888` (page 30)
: Takes an ARGB8888 image in premultiplied alpha format and transforms it into an image in nonpremultiplied alpha format.

`vImageUnpremultiplyData_RGBA8888` (page 33)
: Takes an RGBA8888 image in premultiplied alpha format and transforms it into an image in nonpremultiplied alpha format.

`vImageUnpremultiplyData_PlanarF` (page 33)
: Takes a PlanarF image in premultiplied alpha format, along with alpha information, and transforms it into an image in nonpremultiplied alpha format.

`vImageUnpremultiplyData_Planar8` (page 32)
: Takes a Planar8 image in premultiplied alpha format, along with alpha information, and transforms it into an image in nonpremultiplied alpha format.

Clipping Color Values to Alpha

`vImageClipToAlpha_Planar8` (page 18)
: Sets the color channel of each pixel in a Planar8 image to the smaller of two values—either the color channel or the alpha value for that pixel.

`vImageClipToAlpha_ARGB8888` (page 16)
: Sets the color channel of each pixel in an ARGB8888 image to the smaller of two values—either the color channel or the alpha value for that pixel.

`vImageClipToAlpha_PlanarF` (page 19)
: Sets the color channel of each pixel in a PlanarF image to the smaller of two values—either the color channel or the alpha value for that pixel.

`vImageClipToAlpha_ARGBFFFF` (page 17)
: Sets the color channel of each pixel in an ARGBFFFF image to the smaller of two values—either the color channel or the alpha value for that pixel.

Functions

vImageAlphaBlend_ARGB8888

Performs nonpremultiplied alpha compositing of two ARGB8888 images, placing the result in a destination buffer.

```
vImage_Error vImageAlphaBlend_ARGB8888 (
   const vImage_Buffer *srcTop,
   const vImage_Buffer *srcBottom,
   const vImage_Buffer *dest,
   vImage_Flags flags
);
```

Parameters

srcTop

A pointer to a vImage buffer structure that contains data for the top source image.

srcBottom

A pointer to a vImage buffer structure that contains data for the bottom source image.

dest

A pointer to a vImage buffer data structure. You are 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 pointed to by this structure contains the destination image data. When you no longer need the data buffer, you must deallocate the memory.

flags

The options to use when performing the compositing. Pass `kvImageDoNotTile` if you plan to perform your own tiling or use multithreading.

Return Value

`kvImageNoError`, otherwise one of the error codes described in *vImage Data Types and Constants Reference*.

Discussion

The vImage buffer structures for the source and destination images must use the same height and width.

The calculation for each color channel is:

```
resultAlpha = (topAlpha * 255 + (255 - topAlpha)
                 * bottomAlpha + 127) / 255
resultColor = (topAlpha * topColor + (((255 - topAlpha)
                 * bottomAlpha + 127) / 255) * bottomColor +  127)
                    / resultAlpha
```

Availability

Available in Mac OS X v10.3 and later.

Declared In

`Alpha.h`

vImageAlphaBlend_ARGBFFFF

Performs nonpremultiplied alpha compositing of two ARGBFFFF images, placing the result in a destination buffer.

```
vImage_Error vImageAlphaBlend_ARGBFFFF (
   const vImage_Buffer *srcTop,
   const vImage_Buffer *srcBottom,
   const vImage_Buffer *dest,
   vImage_Flags flags
);
```

Parameters

srcTop

A pointer to a vImage buffer structure that contains data for the top source image.

srcBottom

A pointer to a vImage buffer structure that contains data for the bottom source image.

dest

A pointer to a vImage buffer data structure. You are 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 pointed to by this structure contains the destination image data. When you no longer need the data buffer, you must deallocate the memory.

flags

The options to use when performing the compositing. Pass `kvImageDoNotTile` if you plan to perform your own tiling or use multithreading.

Return Value

`kvImageNoError`, otherwise one of the error codes described in *vImage Data Types and Constants Reference*.

Discussion

The vImage buffer structures for the source and destination images must use the same height and width.

The calculation for each color channel is:

```
resultAlpha = topAlpha + (1.0f - topAlpha) * bottomAlpha
resultColor = (topAlpha * topColor + (1.0f - topAlpha)
                  * bottomAlpha * bottomColor) / resultAlpha
```

Availability

Available in Mac OS X v10.3 and later.

Declared In

`Alpha.h`

vImageAlphaBlend_NonpremultipliedToPremultiplied_ARGB8888

Performs mixed alpha compositing of a nonpremultiplied ARGB8888 image over a premultiplied ARGB8888 image, placing the premultiplied result in a destination buffer.

```
vImage_Error vImageAlphaBlend_NonpremultipliedToPremultiplied_ARGB8888 (
   const vImage_Buffer *srcTop,
   const vImage_Buffer *srcBottom,
   const vImage_Buffer *dest,
   vImage_Flags flags
);
```

Parameters

srcTop

A pointer to a vImage buffer structure that contains the nonpremultiplied data for the top source image.

srcBottom

A pointer to a vImage buffer structure that contains data for the bottom source image.

dest

A pointer to a vImage buffer data structure. You are 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 pointed to by this structure contains the destination image data. When you no longer need the data buffer, you must deallocate the memory.

flags

The options to use when performing the compositing. Pass `kvImageDoNotTile` if you plan to perform your own tiling or use multithreading.

Return Value

`kvImageNoError`, otherwise one of the error codes described in *vImage Data Types and Constants Reference*.

Discussion

The vImage buffer structures for the source images must be at least as wide and at least as high as the destination buffer.

Availability

Available in Mac OS X v10.4 and later.

Declared In

`Alpha.h`

vImageAlphaBlend_NonpremultipliedToPremultiplied_ARGBFFFF

Performs mixed alpha compositing of a nonpremultiplied ARGBFFFF image over a premultiplied ARGBFFFF image, placing the premultiplied result in a destination buffer.

```
vImage_Error vImageAlphaBlend_NonpremultipliedToPremultiplied_ARGBFFFF (
   const vImage_Buffer *srcTop,
   const vImage_Buffer *srcBottom,
   const vImage_Buffer *dest,
   vImage_Flags flags
);
```

Parameters

srcTop

A pointer to a vImage buffer structure that contains the nonpremultiplied data for the top source image.

srcBottom

A pointer to a vImage buffer structure that contains data for the bottom source image.

dest
: A pointer to a vImage buffer data structure. You are 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 pointed to by this structure contains the destination image data. When you no longer need the data buffer, you must deallocate the memory.

flags
: The options to use when performing the compositing. Pass `kvImageDoNotTile` if you plan to perform your own tiling or use multithreading.

Return Value
`kvImageNoError`, otherwise one of the error codes described in *vImage Data Types and Constants Reference*.

Discussion
The vImage buffer structures for the source images must be at least as wide and at least as high as the destination buffer.

Availability
Available in Mac OS X v10.4 and later.

Declared In
`Alpha.h`

vImageAlphaBlend_NonpremultipliedToPremultiplied_Planar8

Performs mixed alpha compositing of a nonpremultiplied Planar8 image over a premultiplied Planar8 image, placing the premultiplied result in a destination buffer.

```
vImage_Error vImageAlphaBlend_NonpremultipliedToPremultiplied_Planar8 (
   const vImage_Buffer *srcTop,
   const vImage_Buffer *srcTopAlpha,
   const vImage_Buffer *srcBottom,
   const vImage_Buffer *dest,
   vImage_Flags flags
);
```

Parameters

srcTop
: A pointer to a vImage buffer structure that contains the nonpremultiplied data for the top source image.

srcTopAlpha
: A pointer to a vImage buffer structure that contains data for the alpha values of the top source image.

srcBottom
: A pointer to a vImage buffer structure that contains data for the bottom source image.

dest
: A pointer to a vImage buffer data structure. You are 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 pointed to by this structure contains the destination image data. When you no longer need the data buffer, you must deallocate the memory.

flags
: The options to use when performing the compositing. Pass `kvImageDoNotTile` if you plan to perform your own tiling or use multithreading.

Return Value
`kvImageNoError`, otherwise one of the error codes described in *vImage Data Types and Constants Reference*.

Discussion
The vImage buffer structures for the source images must be at least as wide and at least as high as the destination buffer.

Availability
Available in Mac OS X v10.4 and later.

Declared In
`Alpha.h`

vImageAlphaBlend_NonpremultipliedToPremultiplied_PlanarF

Performs mixed alpha compositing of a nonpremultiplied PlanarF image over a premultiplied PlanarF image, placing the premultiplied result in a destination buffer.

```
vImage_Error vImageAlphaBlend_NonpremultipliedToPremultiplied_PlanarF (
   const vImage_Buffer *srcTop,
   const vImage_Buffer *srcTopAlpha,
   const vImage_Buffer *srcBottom,
   const vImage_Buffer *dest,
   vImage_Flags flags
);
```

Parameters

`srcTop`
A pointer to a vImage buffer structure that contains the nonpremultiplied data for the top source image.

`srcTopAlpha`
A pointer to a vImage buffer structure that contains data for the alpha values of the top source image.

`srcBottom`
A pointer to a vImage buffer structure that contains data for the bottom source image.

`dest`
A pointer to a vImage buffer data structure. You are 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 pointed to by this structure contains the destination image data. When you no longer need the data buffer, you must deallocate the memory.

`flags`
The options to use when performing the compositing. Pass `kvImageDoNotTile` if you plan to perform your own tiling or use multithreading.

Return Value
`kvImageNoError`, otherwise one of the error codes described in *vImage Data Types and Constants Reference*.

Discussion
The vImage buffer structures for the source images must be at least as wide and at least as high as the destination buffer.

Availability
Available in Mac OS X v10.4 and later.

Declared In
`Alpha.h`

vImageAlphaBlend_Planar8

Performs nonpremultiplied alpha compositing of two Planar8 images, placing the result in a destination buffer.

```
vImage_Error vImageAlphaBlend_Planar8 (
   const vImage_Buffer *srcTop,
   const vImage_Buffer *srcTopAlpha,
   const vImage_Buffer *srcBottom,
   const vImage_Buffer *srcBottomAlpha,
   const vImage_Buffer *alpha,
   const vImage_Buffer *dest,
   vImage_Flags flags
);
```

Parameters

srcTop
A pointer to a vImage buffer structure that contains data for the top source image.

srcTopAlpha
A pointer to a vImage buffer structure that contains data for the alpha values of the top source image.

srcBottom
A pointer to a vImage buffer structure that contains data for the bottom source image.

srcBottomAlpha
A pointer to a vImage buffer structure that contains data for the alpha values of the bottom source image.

alpha
A pointer to a vImage buffer structure that contains data for the precalculated alpha values of the composite image. You must precalculate these values by calling the function `vPremultipliedAlphaBlend_PlanarF`. See the Discussion for details on using this function.

dest
A pointer to a vImage buffer data structure. You are 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 pointed to by this structure contains the destination image data. When you no longer need the data buffer, you must deallocate the memory.

flags
The options to use when performing the compositing. Pass `kvImageDoNotTile` if you plan to perform your own tiling or use multithreading.

Return Value
`kvImageNoError`, otherwise one of the error codes described in *vImage Data Types and Constants Reference*.

Discussion
The vImage buffer structures for the source, alpha values, destination, and composite alpha values must contain the same height and width.

For performance reasons, this function does not calculate alpha values for the composite image; you must provide them. You'll typically call this function three times, once for each color channel (red, green, blue). Because each call uses the same alpha value, it is much more efficient for you to precalculate the alpha values using the function `vImagePremultipliedAlphaBlend_Planar8`, rather than have the calculation repeated three times by the`vImageAlphaBlend_Planar8` function. Call the function `vPremultipliedAlphaBlend_Planar8` using the parameters shown:

```
vImagePremultipliedAlphaBlend_Planar8(srcTopAlpha,
              srcTopAlpha, // Yes, use this twice
              srcBottomAlpha,
              alpha, //On return, contains the composite alpha values
              kvImageNoFlags );
```

After calling the `vPremultipliedAlphaBlend_Planar8` function, the resulting values for each color channel are:

```
resultAlpha = topAlpha + (1.0f - topAlpha) * bottomAlpha
resultColor = (topAlpha * topColor + (1.0f - topAlpha)
                   * bottomAlpha * bottomColor) / resultAlpha
```

Availability
Available in Mac OS X v10.3 and later.

Declared In
`Alpha.h`

vImageAlphaBlend_PlanarF

Performs nonpremultiplied alpha compositing of two PlanarF images, placing the result in a destination buffer.

```
vImage_Error vImageAlphaBlend_PlanarF (
   const vImage_Buffer *srcTop,
   const vImage_Buffer *srcTopAlpha,
   const vImage_Buffer *srcBottom,
   const vImage_Buffer *srcBottomAlpha,
   const vImage_Buffer *alpha,
   const vImage_Buffer *dest,
   vImage_Flags flags
);
```

Parameters

srcTop
A pointer to a vImage buffer structure that contains data for the top source image.

srcTopAlpha
A pointer to a vImage buffer structure that contains data for the alpha values of the top source image.

srcBottom
A pointer to a vImage buffer structure that contains data for the bottom source image.

srcBottomAlpha
A pointer to a vImage buffer structure that contains data for the alpha values of the bottom source image.

alpha
A pointer to a vImage buffer structure that contains data for the precalculated alpha values of the composite image. You must precalculate these values by calling the function `vPremultipliedAlphaBlend_PlanarF`. See the Discussion for details on using this function.

dest

A pointer to a vImage buffer data structure. You are 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 pointed to by this structure contains the destination image data. When you no longer need the data buffer, you must deallocate the memory.

flags

The options to use when performing the compositing. Pass `kvImageDoNotTile` if you plan to perform your own tiling or use multithreading.

Return Value

`kvImageNoError`, otherwise one of the error codes described in *vImage Data Types and Constants Reference*.

Discussion

The vImage buffer structures for the source, alpha values, destination, and composite alpha values must contain the same height and width.

For performance reasons, this function does not calculate alpha values for the composite image; you must provide them. You'll typically call this function three times, once for each color channel (red, green, blue). Because each call uses the same alpha value, it is much more efficient for you to precalculate the alpha values using the function `vImagePremultipliedAlphaBlend_PlanarF`, rather than to have the calculation repeated three times by the`vImageAlphaBlend_PlanarF` function. Call the function `vPremultipliedAlphaBlend_PlanarF` using the parameters shown:

```
vImagePremultipliedAlphaBlend_PlanarF(srcTopAlpha,
              srcTopAlpha, // Yes, use this twice
              srcBottomAlpha,
              alpha, //On return, contains the composite alpha values
              kvImageNoFlags );
```

After calling the `vPremultipliedAlphaBlend_PlanarF` function, the resulting values for each color channel are:

```
resultAlpha = topAlpha + (1.0f - topAlpha) * bottomAlpha
resultColor = (topAlpha * topColor + (1.0f - topAlpha)
                   * bottomAlpha * bottomColor) / resultAlpha
```

Availability

Available in Mac OS X v10.3 and later.

Declared In

`Alpha.h`

vImageClipToAlpha_ARGB8888

Sets the color channel of each pixel in an ARGB8888 image to the smaller of two values—either the color channel or the alpha value for that pixel.

```
vImage_Error vImageClipToAlpha_ARGB8888 (
   const vImage_Buffer *src,
   const vImage_Buffer *dest,
   vImage_Flags flags
);
```

Parameters

src

A pointer to a vImage buffer structure that contains data for the source image.

dest

A pointer to a vImage buffer data structure. You are 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 pointed to by this structure contains the destination image data. When you no longer need the data buffer, you must deallocate the memory.

flags

The options to use when performing the compositing. Pass `kvImageDoNotTile` if you plan to perform your own tiling or use multithreading.

Return Value

`kvImageNoError`, otherwise one of the error codes described in *vImage Data Types and Constants Reference*.

Discussion

For each pixel:

```
alpha_result = sourceAlpha;
color_result = MIN(sourceColor, sourceAlpha);
```

Availability

Available in Mac OS X v10.4 and later.

Declared In

`Alpha.h`

vImageClipToAlpha_ARGBFFFF

Sets the color channel of each pixel in an ARGBFFFF image to the smaller of two values—either the color channel or the alpha value for that pixel.

```
vImage_Error vImageClipToAlpha_ARGBFFFF (
   const vImage_Buffer *src,
   const vImage_Buffer *dest,
   vImage_Flags flags
);
```

Parameters

src

A pointer to a vImage buffer structure that contains data for the source image.

dest

A pointer to a vImage buffer data structure. You are 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 pointed to by this structure contains the destination image data. When you no longer need the data buffer, you must deallocate the memory.

flags

The options to use when performing the compositing. Pass `kvImageDoNotTile` if you plan to perform your own tiling or use multithreading.

Return Value

`kvImageNoError`, otherwise one of the error codes described in *vImage Data Types and Constants Reference*.

Discussion

For each pixel:

```
alpha_result = sourceAlpha;
color_result = MIN(sourceColor, sourceAlpha);
```

Availability

Available in Mac OS X v10.4 and later.

Declared In

`Alpha.h`

vImageClipToAlpha_Planar8

Sets the color channel of each pixel in a Planar8 image to the smaller of two values—either the color channel or the alpha value for that pixel.

```
vImage_Error vImageClipToAlpha_Planar8 (
   const vImage_Buffer *src,
   const vImage_Buffer *alpha,
   const vImage_Buffer *dest,
   vImage_Flags flags
);
```

Parameters

src

A pointer to a vImage buffer structure that contains data for the top source image.

alpha

A pointer to a vImage buffer structure that contains data for alpha values of the source image. The planar source image does not contain its own alpha information, so you must supply the alpha information.

dest

A pointer to a vImage buffer data structure. You are 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 pointed to by this structure contains the destination image data. When you no longer need the data buffer, you must deallocate the memory.

flags

The options to use when performing the compositing. Pass `kvImageDoNotTile` if you plan to perform your own tiling or use multithreading.

Return Value

`kvImageNoError`, otherwise one of the error codes described in *vImage Data Types and Constants Reference*.

Discussion

For each pixel:

```
alpha_result = sourceAlpha;
```

```
color_result = MIN(sourceColor, sourceAlpha);
```

Availability
Available in Mac OS X v10.4 and later.

Declared In
`Alpha.h`

vImageClipToAlpha_PlanarF

Sets the color channel of each pixel in a PlanarF image to the smaller of two values—either the color channel or the alpha value for that pixel.

```
vImage_Error vImageClipToAlpha_PlanarF (
   const vImage_Buffer *src,
   const vImage_Buffer *alpha,
   const vImage_Buffer *dest,
   vImage_Flags flags
);
```

Parameters

src
A pointer to a vImage buffer structure that contains data for the source image.

alpha
A pointer to a vImage buffer structure that contains data for alpha values of the source image. The planar source image does not contain its own alpha information, so you must supply the alpha information.

dest
A pointer to a vImage buffer data structure. You are 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 pointed to by this structure contains the destination image data. When you no longer need the data buffer, you must deallocate the memory.

flags
The options to use when performing the compositing. Pass `kvImageDoNotTile` if you plan to perform your own tiling or use multithreading.

Return Value
`kvImageNoError`, otherwise one of the error codes described in *vImage Data Types and Constants Reference*.

Discussion
For each pixel:

```
alpha_result = sourceAlpha;
color_result = MIN(sourceColor, sourceAlpha);
```

Availability
Available in Mac OS X v10.4 and later.

Declared In
`Alpha.h`

vImagePremultipliedAlphaBlend_ARGB8888

Performs premultiplied alpha compositing of two ARGB8888 images, placing the result in a destination buffer.

```
vImage_Error vImagePremultipliedAlphaBlend_ARGB8888 (
   const vImage_Buffer *srcTop,
   const vImage_Buffer *srcBottom,
   const vImage_Buffer *dest,
   vImage_Flags flags
);
```

Parameters

srcTop

A pointer to a vImage buffer structure that contains data for the top source image.

srcBottom

A pointer to a vImage buffer structure that contains data for the bottom source image.

dest

A pointer to a vImage buffer data structure. You are 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 pointed to by this structure contains the destination image data. When you no longer need the data buffer, you must deallocate the memory.

flags

The options to use when performing the compositing. Pass `kvImageDoNotTile` if you plan to perform your own tiling or use multithreading.

Return Value

`kvImageNoError`, otherwise one of the error codes described in *vImage Data Types and Constants Reference*.

Discussion

The vImage buffer structures for the source and destination images must use the same height and width.

Availability

Available in Mac OS X v10.3 and later.

Declared In

`Alpha.h`

vImagePremultipliedAlphaBlend_ARGBFFFF

Performs premultiplied alpha compositing of two ARGBFFFF images, placing the result in a destination buffer.

```
vImage_Error vImagePremultipliedAlphaBlend_ARGBFFFF (
   const vImage_Buffer *srcTop,
   const vImage_Buffer *srcBottom,
   const vImage_Buffer *dest,
   vImage_Flags flags
);
```

Parameters

srcTop

A pointer to a vImage buffer structure that contains data for the top source image.

srcBottom

A pointer to a vImage buffer structure that contains data for the bottom source image.

dest

A pointer to a vImage buffer data structure. You are 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 pointed to by this structure contains the destination image data. When you no longer need the data buffer, you must deallocate the memory.

flags

The options to use when performing the compositing. Pass `kvImageDoNotTile` if you plan to perform your own tiling or use multithreading.

Return Value

`kvImageNoError`, otherwise one of the error codes described in *vImage Data Types and Constants Reference*.

Discussion

The vImage buffer structures for the source and destination images must use the same height and width.

Availability

Available in Mac OS X v10.3 and later.

Declared In

`Alpha.h`

vImagePremultipliedAlphaBlend_Planar8

Performs premultiplied alpha compositing of two Planar8 images, placing the result in a destination buffer.

```
vImage_Error vImagePremultipliedAlphaBlend_Planar8 (
   const vImage_Buffer *srcTop,
   const vImage_Buffer *srcTopAlpha,
   const vImage_Buffer *srcBottom,
   const vImage_Buffer *dest,
   vImage_Flags flags
);
```

Parameters

srcTop

A pointer to a vImage buffer structure that contains data for the top source image.

srcTopAlpha

A pointer to a vImage buffer structure that contains data for the alpha values of the top source image. Even though the alpha values are already premultiplied into the pixel values, the function also requires the original alpha information for the top image to do its calculations. There is no way to extract this information from the premultiplied planar values, so you must provide it.

srcBottom

A pointer to a vImage buffer structure that contains data for the bottom source image.

dest

A pointer to a vImage buffer data structure. You are 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 pointed to by this structure contains the destination image data. When you no longer need the data buffer, you must deallocate the memory.

flags

The options to use when performing the compositing. Pass `kvImageDoNotTile` if you plan to perform your own tiling or use multithreading.

Return Value
`kvImageNoError`, otherwise one of the error codes described in *vImage Data Types and Constants Reference*.

Discussion
The vImage buffer structures for the source and destination images must use the same height and width.

Availability
Available in Mac OS X v10.3 and later.

Declared In
`Alpha.h`

vImagePremultipliedAlphaBlend_PlanarF

Performs premultiplied alpha compositing of two PlanarF images, placing the result in a destination buffer.

```
vImage_Error vImagePremultipliedAlphaBlend_PlanarF (
   const vImage_Buffer *srcTop,
   const vImage_Buffer *srcTopAlpha,
   const vImage_Buffer *srcBottom,
   const vImage_Buffer *dest,
   vImage_Flags flags
);
```

Parameters

srcTop
A pointer to a vImage buffer structure that contains data for the top source image.

srcTopAlpha
A pointer to a vImage buffer structure that contains data for the alpha values of the top source image. Even though the alpha values are already premultiplied into the pixel values, the function also requires the original alpha information for the top image to do its calculations. There is no way to extract this information from the premultiplied planar values, so you must provide it.

srcBottom
A pointer to a vImage buffer structure that contains data for the bottom source image.

dest
A pointer to a vImage buffer data structure. You are 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 pointed to by this structure contains the destination image data. When you no longer need the data buffer, you must deallocate the memory.

flags
The options to use when performing the compositing. Pass `kvImageDoNotTile` if you plan to perform your own tiling or use multithreading.

Return Value
`kvImageNoError`, otherwise one of the error codes described in *vImage Data Types and Constants Reference*.

Availability
Available in Mac OS X v10.3 and later.

Declared In
`Alpha.h`

vImagePremultipliedConstAlphaBlend_ARGB8888

Performs premultiplied alpha compositing of two ARGB8888 images, using a single alpha value for the whole image and placing the result in a destination buffer.

```
vImage_Error vImagePremultipliedConstAlphaBlend_ARGB8888 (
   const vImage_Buffer *srcTop,
   Pixel_8 constAlpha,
   const vImage_Buffer *srcBottom,
   const vImage_Buffer *dest,
   vImage_Flags flags
);
```

Parameters

srcTop

A pointer to a vImage buffer structure that contains data for the top source image.

constAlpha

The alpha value you want to apply to the image.

srcBottom

A pointer to a vImage buffer structure that contains data for the bottom source image.

dest

A pointer to a vImage buffer data structure. You are 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 pointed to by this structure contains the destination image data. When you no longer need the data buffer, you must deallocate the memory.

flags

The options to use when performing the compositing. Pass `kvImageDoNotTile` if you plan to perform your own tiling or use multithreading.

Return Value

`kvImageNoError`, otherwise one of the error codes described in *vImage Data Types and Constants Reference*.

Discussion

The vImage buffer structures for the source and destination images must use the same height and width.

Availability

Available in Mac OS X v10.4 and later.

Declared In

`Alpha.h`

vImagePremultipliedConstAlphaBlend_ARGBFFFF

Performs premultiplied alpha compositing of two ARGBFFFF images, using a single alpha value for the whole image and placing the result in a destination buffer.

```
vImage_Error vImagePremultipliedConstAlphaBlend_ARGBFFFF (
   const vImage_Buffer *srcTop,
   Pixel_F constAlpha,
   const vImage_Buffer *srcBottom,
   const vImage_Buffer *dest,
   vImage_Flags flags
);
```

Parameters

srcTop

A pointer to a vImage buffer structure that contains data for the top source image.

constAlpha

The alpha value you want to apply to the image.

srcBottom

A pointer to a vImage buffer structure that contains data for the bottom source image.

dest

A pointer to a vImage buffer data structure. You are 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 pointed to by this structure contains the destination image data. When you no longer need the data buffer, you must deallocate the memory.

flags

The options to use when performing the compositing. Pass `kvImageDoNotTile` if you plan to perform your own tiling or use multithreading.

Return Value

`kvImageNoError`, otherwise one of the error codes described in *vImage Data Types and Constants Reference*.

Discussion

The vImage buffer structures for the source and destination images must use the same height and width.

Availability

Available in Mac OS X v10.4 and later.

Declared In

`Alpha.h`

vImagePremultipliedConstAlphaBlend_Planar8

Performs premultiplied alpha compositing of two Planar8 images, using a single alpha value for the entire image and placing the result in a destination buffer.

```
vImage_Error vImagePremultipliedConstAlphaBlend_Planar8 (
   const vImage_Buffer *srcTop,
   Pixel_8 constAlpha,
   const vImage_Buffer *srcTopAlpha,
   const vImage_Buffer *srcBottom,
   const vImage_Buffer *dest,
   vImage_Flags flags
);
```

Parameters

srcTop

A pointer to a vImage buffer structure that contains data for the top source image.

constAlpha

The alpha value you want to apply to the image.

srcTopAlpha

A pointer to a vImage buffer structure that contains data for the alpha values of the top source image. Even though the alpha values are already premultiplied into the pixel values, the function also requires the original alpha information for the top image to do its calculations. There is no way to extract this information from the premultiplied planar values, so you must provide it.

srcBottom

A pointer to a vImage buffer structure that contains data for the bottom source image.

dest

A pointer to a vImage buffer data structure. You are 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 pointed to by this structure contains the destination image data. When you no longer need the data buffer, you must deallocate the memory.

flags

The options to use when performing the compositing. Pass `kvImageDoNotTile` if you plan to perform your own tiling or use multithreading.

Return Value

`kvImageNoError`, otherwise one of the error codes described in *vImage Data Types and Constants Reference*.

Discussion

The vImage buffer structures for the source and destination images must use the same height and width.

Availability

Available in Mac OS X v10.4 and later.

Declared In

`Alpha.h`

vImagePremultipliedConstAlphaBlend_PlanarF

Performs premultiplied alpha compositing of a two PlanarF images, using a single alpha value for the whole image and placing the result in a destination buffer.

```
vImage_Error vImagePremultipliedConstAlphaBlend_PlanarF (
   const vImage_Buffer *srcTop,
   Pixel_F constAlpha,
   const vImage_Buffer *srcTopAlpha,
   const vImage_Buffer *srcBottom,
   const vImage_Buffer *dest,
   vImage_Flags flags
);
```

Parameters

srcTop

A pointer to a vImage buffer structure that contains data for the top source image.

constAlpha

The alpha value you want to apply to the image.

srcTopAlpha
: A pointer to a vImage buffer structure that contains data for the alpha values of the top source image. Even though the alpha values are already premultiplied into the pixel values, the function also requires the original alpha information for the top image to do its calculations. There is no way to extract this information from the premultiplied planar values, so you must provide it.

srcBottom
: A pointer to a vImage buffer structure that contains data for the bottom source image.

dest
: A pointer to a vImage buffer data structure. You are 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 pointed to by this structure contains the destination image data. When you no longer need the data buffer, you must deallocate the memory.

flags
: The options to use when performing the compositing. Pass `kvImageDoNotTile` if you plan to perform your own tiling or use multithreading.

Return Value
`kvImageNoError`, otherwise one of the error codes described in *vImage Data Types and Constants Reference*.

Discussion
The vImage buffer structures for the source and destination images must use the same height and width.

Availability
Available in Mac OS X v10.4 and later.

Declared In
`Alpha.h`

vImagePremultiplyData_ARGB8888

Takes an ARGB8888 image in nonpremultiplied alpha format and transforms it into an image in premultiplied alpha format.

```
vImage_Error vImagePremultiplyData_ARGB8888 (
   const vImage_Buffer *src,
   const vImage_Buffer *dest,
   vImage_Flags flags
);
```

Parameters

src
: A pointer to a vImage buffer structure that contains the nonpremultiplied data for the top source image.

dest
: A pointer to a vImage buffer data structure. You are 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 pointed to by this structure contains the destination image data. When you no longer need the data buffer, you must deallocate the memory.

flags
: The options to use when performing the compositing. Pass `kvImageDoNotTile` if you plan to perform your own tiling or use multithreading.

Return Value
`kvImageNoError`, otherwise one of the error codes described in *vImage Data Types and Constants Reference*.

Discussion
This function gets the required alpha information from the alpha channel of the original image. The alpha channel is copied over unchanged to the destination image.

Availability
Available in Mac OS X v10.3 and later.

Declared In
`Alpha.h`

vImagePremultiplyData_ARGBFFFF

Takes an ARGBFFFF image in nonpremultiplied alpha format and transforms it into an image in premultiplied alpha format.

```
vImage_Error vImagePremultiplyData_ARGBFFFF (
   const vImage_Buffer *src,
   const vImage_Buffer *dest,
   vImage_Flags flags
);
```

Parameters

`src`
: A pointer to a vImage buffer structure that contains the nonpremultiplied data for the top source image.

`dest`
: A pointer to a vImage buffer data structure. You are 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 pointed to by this structure contains the destination image data. When you no longer need the data buffer, you must deallocate the memory.

`flags`
: The options to use when performing the compositing. Pass `kvImageDoNotTile` if you plan to perform your own tiling or use multithreading.

Return Value
`kvImageNoError`, otherwise one of the error codes described in *vImage Data Types and Constants Reference*.

Discussion
This function gets the required alpha information from the alpha channel of the original image. The alpha channel is copied over unchanged to the destination image.

Availability
Available in Mac OS X v10.3 and later.

Declared In
`Alpha.h`

vImagePremultiplyData_Planar8

Takes a Planar8 image in nonpremultiplied alpha format, along with alpha information, and transforms it into an image in premultiplied alpha format.

```
vImage_Error vImagePremultiplyData_Planar8 (
   const vImage_Buffer *src,
   const vImage_Buffer *alpha,
   const vImage_Buffer *dest,
   vImage_Flags flags
);
```

Parameters

src

A pointer to a vImage buffer structure that contains the nonpremultiplied data for the top source image.

alpha

A pointer to a vImage buffer structure that contains data for alpha values of the source image. The planar source image does not contain its own alpha information, so you must supply the alpha information.

dest

A pointer to a vImage buffer data structure. You are 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 pointed to by this structure contains the destination image data. When you no longer need the data buffer, you must deallocate the memory.

flags

The options to use when performing the compositing. Pass `kvImageDoNotTile` if you plan to perform your own tiling or use multithreading.

Return Value

`kvImageNoError`, otherwise one of the error codes described in *vImage Data Types and Constants Reference*.

Availability

Available in Mac OS X v10.3 and later.

Declared In

`Alpha.h`

vImagePremultiplyData_PlanarF

Takes a PlanarF image in nonpremultiplied alpha format, along with alpha information, and transforms it into an image in premultiplied alpha format.

```
vImage_Error vImagePremultiplyData_PlanarF (
   const vImage_Buffer *src,
   const vImage_Buffer *alpha,
   const vImage_Buffer *dest,
   vImage_Flags flags
);
```

Parameters

src

A pointer to a vImage buffer structure that contains the nonpremultiplied data for the top source image.

`alpha`
: A pointer to a vImage buffer structure that contains data for alpha values of the source image.

`dest`
: A pointer to a vImage buffer data structure. You are 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 pointed to by this structure contains the destination image data. When you no longer need the data buffer, you must deallocate the memory.

`flags`
: The options to use when performing the compositing. Pass `kvImageDoNotTile` if you plan to perform your own tiling or use multithreading.

Return Value

`kvImageNoError`, otherwise one of the error codes described in *vImage Data Types and Constants Reference*.

Availability

Available in Mac OS X v10.3 and later.

Declared In

`Alpha.h`

vImagePremultiplyData_RGBA8888

Takes an RGBA8888 image in nonpremultiplied alpha format and transforms it into an image in premultiplied alpha format.

```
vImage_Error vImagePremultiplyData_RGBA8888 (
   const vImage_Buffer *src,
   const vImage_Buffer *dest,
   vImage_Flags flags
);
```

Parameters

`src`
: A pointer to a vImage buffer structure that contains the nonpremultiplied data for the top source image.

`dest`
: A pointer to a vImage buffer data structure. You are 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 pointed to by this structure contains the destination image data. When you no longer need the data buffer, you must deallocate the memory.

`flags`
: The options to use when performing the compositing. Pass `kvImageDoNotTile` if you plan to perform your own tiling or use multithreading.

Return Value

`kvImageNoError`, otherwise one of the error codes described in *vImage Data Types and Constants Reference*.

Discussion

This function gets the required alpha information from the alpha channel of the original image. The alpha channel is copied over unchanged to the destination image.

Availability

Available in Mac OS X v10.4 and later.

Declared In
`Alpha.h`

vImagePremultiplyData_RGBAFFFF

Takes an RGBAFFFF image in nonpremultiplied alpha format and transforms it into an image in premultiplied alpha format.

```
vImage_Error vImagePremultiplyData_RGBAFFFF (
   const vImage_Buffer *src,
   const vImage_Buffer *dest,
   vImage_Flags flags
);
```

Parameters

src
A pointer to a vImage buffer structure that contains the nonpremultiplied data for the top source image.

dest
A pointer to a vImage buffer data structure. You are 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 pointed to by this structure contains the destination image data. When you no longer need the data buffer, you must deallocate the memory.

flags
The options to use when performing the compositing. Pass `kvImageDoNotTile` if you plan to perform your own tiling or use multithreading.

Return Value
`kvImageNoError`, otherwise one of the error codes described in *vImage Data Types and Constants Reference*.

Discussion
This function gets the required alpha information from the alpha channel of the original image. The alpha channel is copied over unchanged to the destination image.

Availability
Available in Mac OS X v10.4 and later.

Declared In
`Alpha.h`

vImageUnpremultiplyData_ARGB8888

Takes an ARGB8888 image in premultiplied alpha format and transforms it into an image in nonpremultiplied alpha format.

```
vImage_Error vImageUnpremultiplyData_ARGB8888 (
   const vImage_Buffer *src,
   const vImage_Buffer *dest,
   vImage_Flags flags
);
```

Parameters

src

A pointer to a vImage buffer structure that contains the nonpremultiplied data for the top source image.

dest

A pointer to a vImage buffer data structure. You are 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 pointed to by this structure contains the destination image data. When you no longer need the data buffer, you must deallocate the memory.

flags

The options to use when performing the compositing. Pass `kvImageDoNotTile` if you plan to perform your own tiling or use multithreading.

Return Value

`kvImageNoError`, otherwise one of the error codes described in *vImage Data Types and Constants Reference*.

Discussion

This function gets the required alpha information from the alpha channel of the original image. The alpha channel is copied over unchanged to the destination image.

Availability

Available in Mac OS X v10.3 and later.

Declared In

`Alpha.h`

vImageUnpremultiplyData_ARGBFFFF

Takes an ARGBFFFF image in premultiplied alpha format and transforms it into an image in nonpremultiplied alpha format.

```
vImage_Error vImageUnpremultiplyData_ARGBFFFF (
   const vImage_Buffer *src,
   const vImage_Buffer *dest,
   vImage_Flags flags
);
```

Parameters

src

A pointer to a vImage buffer structure that contains the nonpremultiplied data for the top source image.

dest

A pointer to a vImage buffer data structure. You are 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 pointed to by this structure contains the destination image data. When you no longer need the data buffer, you must deallocate the memory.

flags

The options to use when performing the compositing. Pass `kvImageDoNotTile` if you plan to perform your own tiling or use multithreading.

Return Value

`kvImageNoError`, otherwise one of the error codes described in *vImage Data Types and Constants Reference*.

Discussion

This function gets the required alpha information from the alpha channel of the original image. The alpha channel is copied over unchanged to the destination image.

Availability

Available in Mac OS X v10.3 and later.

Declared In

`Alpha.h`

vImageUnpremultiplyData_Planar8

Takes a Planar8 image in premultiplied alpha format, along with alpha information, and transforms it into an image in nonpremultiplied alpha format.

```
vImage_Error vImageUnpremultiplyData_Planar8 (
   const vImage_Buffer *src,
   const vImage_Buffer *alpha,
   const vImage_Buffer *dest,
   vImage_Flags flags
);
```

Parameters

src

A pointer to a vImage buffer structure that contains the premultiplied data for the source image.]

alpha

A pointer to a vImage buffer structure that contains data for alpha values of the source image. The planar source image does not contain its own alpha information, so you must supply the alpha information.

dest

A pointer to a vImage buffer data structure. You are 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 pointed to by this structure contains the destination image data. When you no longer need the data buffer, you must deallocate the memory.

flags

The options to use when performing the compositing. Pass `kvImageDoNotTile` if you plan to perform your own tiling or use multithreading.

Return Value

`kvImageNoError`, otherwise one of the error codes described in *vImage Data Types and Constants Reference*.

Availability

Available in Mac OS X v10.3 and later.

Declared In

`Alpha.h`

vImageUnpremultiplyData_PlanarF

Takes a PlanarF image in premultiplied alpha format, along with alpha information, and transforms it into an image in nonpremultiplied alpha format.

```
vImage_Error vImageUnpremultiplyData_PlanarF (
   const vImage_Buffer *src,
   const vImage_Buffer *alpha,
   const vImage_Buffer *dest,
   vImage_Flags flags
);
```

Parameters

src

A pointer to a vImage buffer structure that contains the premultiplied data for the source image.]

alpha

A pointer to a vImage buffer structure that contains data for alpha values of the source image. The planar source image does not contain its own alpha information, so you must supply the alpha information.

dest

A pointer to a vImage buffer data structure. You are 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 pointed to by this structure contains the destination image data. When you no longer need the data buffer, you must deallocate the memory.

flags

The options to use when performing the compositing. Pass `kvImageDoNotTile` if you plan to perform your own tiling or use multithreading.

Return Value

`kvImageNoError`, otherwise one of the error codes described in *vImage Data Types and Constants Reference*.

Availability

Available in Mac OS X v10.3 and later.

Declared In

`Alpha.h`

vImageUnpremultiplyData_RGBA8888

Takes an RGBA8888 image in premultiplied alpha format and transforms it into an image in nonpremultiplied alpha format.

```
vImage_Error vImageUnpremultiplyData_RGBA8888 (
   const vImage_Buffer *src,
   const vImage_Buffer *dest,
   vImage_Flags flags
);
```

Parameters

src

A pointer to a vImage buffer structure that contains the premultiplied data for the source image.

dest

A pointer to a vImage buffer data structure. You are 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 pointed to by this structure contains the destination image data. When you no longer need the data buffer, you must deallocate the memory.

flags

The options to use when performing the compositing. Pass `kvImageDoNotTile` if you plan to perform your own tiling or use multithreading.

Return Value

`kvImageNoError`, otherwise one of the error codes described in *vImage Data Types and Constants Reference*.

Discussion

This function gets the required alpha information from the alpha channel of the original image. The alpha channel is copied over unchanged to the destination image.

Availability

Available in Mac OS X v10.4 and later.

Declared In

`Alpha.h`

vImageUnpremultiplyData_RGBAFFFF

Takes an RGBAFFFF image in premultiplied alpha format and transforms it into an image in nonpremultiplied alpha format.

```
vImage_Error vImageUnpremultiplyData_RGBAFFFF (
   const vImage_Buffer *src,
   const vImage_Buffer *dest,
   vImage_Flags flags
);
```

Parameters

src

A pointer to a vImage buffer structure that contains the nonpremultiplied data for the top source image.

dest

A pointer to a vImage buffer data structure. You are 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 pointed to by this structure contains the destination image data. When you no longer need the data buffer, you must deallocate the memory.

flags

The options to use when performing the compositing. Pass `kvImageDoNotTile` if you plan to perform your own tiling or use multithreading.

Return Value

`kvImageNoError`, otherwise one of the error codes described in *vImage Data Types and Constants Reference*.

Discussion

This function gets the required alpha information from the alpha channel of the original image. The alpha channel is copied over unchanged to the destination image.

Availability
Available in Mac OS X v10.4 and later.

Declared In
`Alpha.h`

Document Revision History

This table describes the changes to *vImage Alpha Compositing Reference*.

Date	Notes
2007-07-12	New document that describes the programming interface for high-performance alpha compositing operations.
	The content in this document was formerly part of *Optimizing Image Processing With vImage*.
	Added `vImageClipToAlpha_Planar8` (page 18), `vImageAlphaBlend_ARGBFFFF` (page 9), `vImageClipToAlpha_PlanarF` (page 19), and `vImageClipToAlpha_ARGBFFFF` (page 17).

Index