OpenCV Cheat Sheet for Reading and Writing Images with Python

Reading and Writing Images with OpenCV

OpenCV provides various methods to read and write images in Python. The most commonly used functions are `cv2.imread()` to read an image, and `cv2.imwrite()` to save an image. Here's an example:

```
import cv2

# Load an image in color mode
img = cv2.imread('image.jpg')

# Display the loaded image
cv2.imshow('Image', img)

# Save the loaded image
cv2.imwrite('new_image.jpg', img)

# Wait for a key press and then close the window
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Using OpenCV to Load Images in Grayscale

Grayscale images are images that contain only shades of gray and no color information. You can use OpenCV to load images in grayscale mode by specifying the `cv2.IMREAD_GRAYSCALE` flag when calling the `cv2.imread()` function:

```
import cv2

# Load an image in grayscale mode
img_gray = cv2.imread('image.jpg', cv2.IMREAD_GRAYSCALE)

# Display the grayscale image
cv2.imshow('Grayscale Image', img_gray)

# Wait for a key press and then close the window
```

```
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Recognizing the Use of BGR and RGB Color Spaces in OpenCV and Pillow Libraries

OpenCV and Pillow libraries use different color space conventions. OpenCV uses the BGR color space, while Pillow uses the RGB color space. You can convert images from one color space to another using OpenCV's `cv2.cvtColor()` function:

```
import cv2

# Load an image in color mode
img = cv2.imread('image.jpg')

# Convert the image from BGR to RGB color space
img_rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

# Display the RGB image
cv2.imshow('RGB Image', img_rgb)

# Wait for a key press and then close the window
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Transforming Color Space of an Image in OpenCV

You can transform the color space of an image in OpenCV using the `cv2.cvtColor()` function. Here's an example of converting an image from BGR to HSV color space:

```
import cv2

# Load an image in color mode
img = cv2.imread('image.jpg')

# Convert the image from BGR to HSV color space
img_hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)

# Display the HSV image
cv2.imshow('HSV Image', img_hsv)

# Wait for a key press and then close the window
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Separating Channels of a Color Image in OpenCV

You can separate the color channels of a color image in OpenCV using the `cv2.split()` function. Here's an example:

```
import cv2

# Load an image in color mode
img = cv2.imread('image.jpg')

# Split the color channels of the image
b, g, r = cv2.split(img)

# Display the separated channels
cv2.imshow('Blue Channel', b)
cv2.imshow('Green Channel', g)
cv2.imshow('Red Channel', r)

# Wait for a key press and then close the windows
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Using OpenCV's Add Operation to Combine Images

You can use OpenCV's `cv2.add()` function to combine two images. Here's an example:

```
import cv2

# Load two images in color mode
img1 = cv2.imread('image1.jpg')
img2 = cv2.imread('image2.jpg')

# Combine the images using the add operation
added_img = cv2.add(img1, img2)

# Display the combined image
cv2.imshow('Added Image', added_img)

# Wait for a key press and then close the window
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Subtracting Images in OpenCV

You can subtract one image from another in OpenCV using the `cv2.subtract()` function. Here's an example:

```
import cv2

# Load two images in color mode
img1 = cv2.imread('image1.jpg')
img2 = cv2.imread('image2.jpg')

# Subtract the second image from the first image
subtracted_img = cv2.subtract(img1, img2)

# Display the subtracted image
cv2.imshow('Subtracted Image', subtracted_img)

# Wait for a key press and then close the window
cv2.waitKey(0)
cv2.destroyAllWindows()
```