Artryan Nero A. Logarta
MCS614_DM01

Assignment #1 Linear Filtering

Code Implementation

Imports

In [1]:
```
import cv2
import numpy as np
import matplotlib.pyplot as plt
```

Loading

In [2]:
```
image = cv2.imread('image.jpg')
image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)

normalized = image / 255.0
```

"Rio de Janeiro"-Inspired Linear Filter

In [3]:
```
def rio_filter(img):
    pink_tint = np.array([1.1, 0.9, 1.05])
    pink_shifted = img * pink_tint
    brighter = np.clip(pink_shifted + 0.05, 0, 1)
    contrast_soft = 0.8 * brighter + 0.2 * 0.5
    return np.clip(contrast_soft, 0, 1)
```

Application

In [4]:
```
filtered = rio_filter(normalized)
filtered_img = (filtered * 255).astype(np.uint8)
```

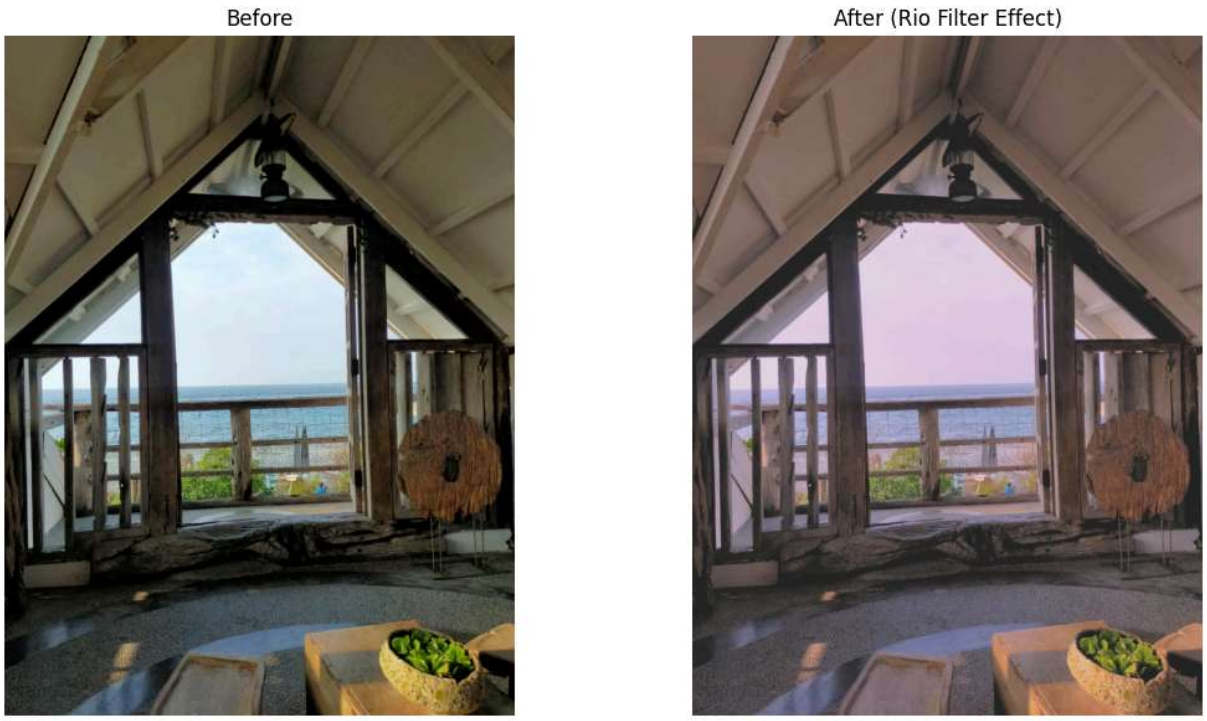
Comparison

In [5]:
```
plt.figure(figsize=(12, 6))
plt.subplot(1, 2, 1)
plt.title("Before")
plt.imshow(image)
```

```
plt.axis("off")

plt.subplot(1, 2, 2)
plt.title("After (Rio Filter Effect)")
plt.imshow(filtered_img)
plt.axis("off")

plt.tight_layout()
plt.show()
```



Explanation

The "Rio de Janeiro"-style filter was recreated using linear operations to simulate its signature warm, pinkish hue and soft contrast. By slightly increasing the red and blue channels while reducing the green, the image shifts toward a magenta tone, giving it a dreamy, vibrant aesthetic. A small brightness boost was applied to mimic the increased exposure often seen in such filters, while contrast was subtly flattened by interpolating the pixel values closer to mid-gray, softening shadows and highlights. These adjustments were done using basic arithmetic and linear color transformations, demonstrating how simple mathematical operations can dramatically change an image's mood and tone.