In [ ]:

```
import cv2
import matplotlib.pyplot as plt
```

In [ ]:

```
# Read the image
image = cv2.imread('test_image.jpg')

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

# Display the image inline
plt.imshow(image)
plt.axis('off')
plt.show()
```

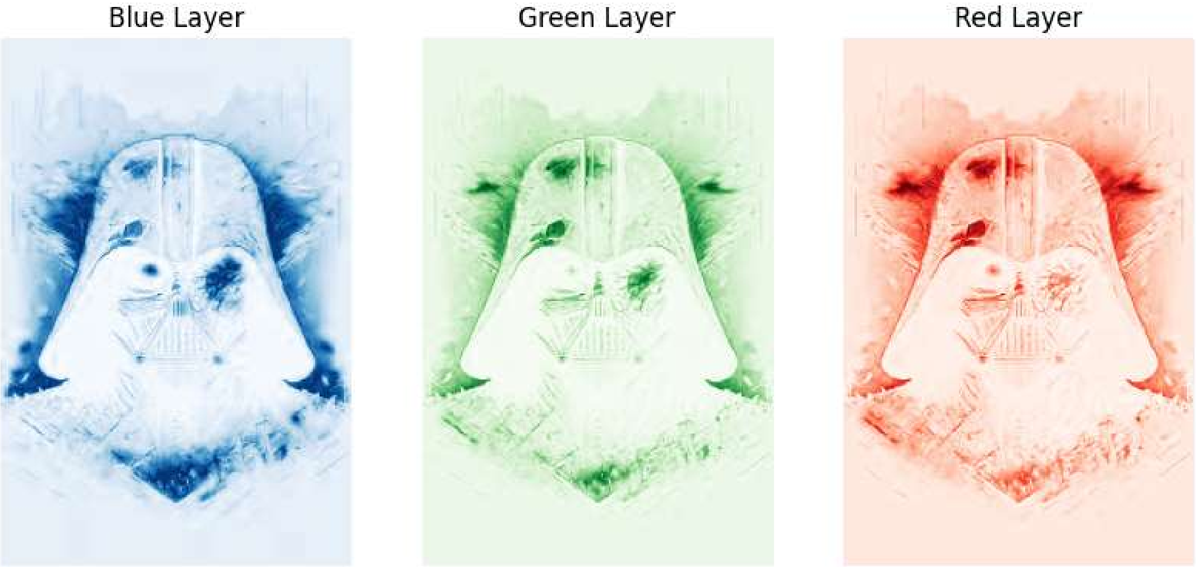
In [ ]:

```
# Print the properties of the image
print("Image shape:", image.shape)
print("Number of rows:", image.shape[0])
print("Number of columns:", image.shape[1])
print("Number of channels:", image.shape[2])
# Print the data type of the image
print("Data type:", image.dtype)

# Print the minimum and maximum pixel values
print("Minimum pixel value:", image.min())
print("Maximum pixel value:", image.max())

# Print the image size in bytes
print("Image size:", image.nbytes, "bytes")
```

```
Image shape: (275, 183, 3)
Number of rows: 275
Number of columns: 183
Number of channels: 3
Data type: uint8
Minimum pixel value: 0
Maximum pixel value: 255
Image size: 150975 bytes
```

In [ ]:

```
# Split the image into its component layers
b, g, r = cv2.split(image)

# Display the component layers
plt.figure(figsize=(10, 5))

plt.subplot(1, 3, 1)
plt.imshow(b, cmap='Blues')
plt.title('Blue Layer')
plt.axis('off')

plt.subplot(1, 3, 2)
plt.imshow(g, cmap='Greens')
plt.title('Green Layer')
plt.axis('off')

plt.subplot(1, 3, 3)
plt.imshow(r, cmap='Reds')
plt.title('Red Layer')
plt.axis('off')

plt.show()
```



Blue Layer

Green Layer

Red Layer

In [ ]:

```
# Convert the image to grayscale
gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

# Display the grayscale image
plt.imshow(gray_image, cmap='gray')
plt.axis('off')
plt.show()
```

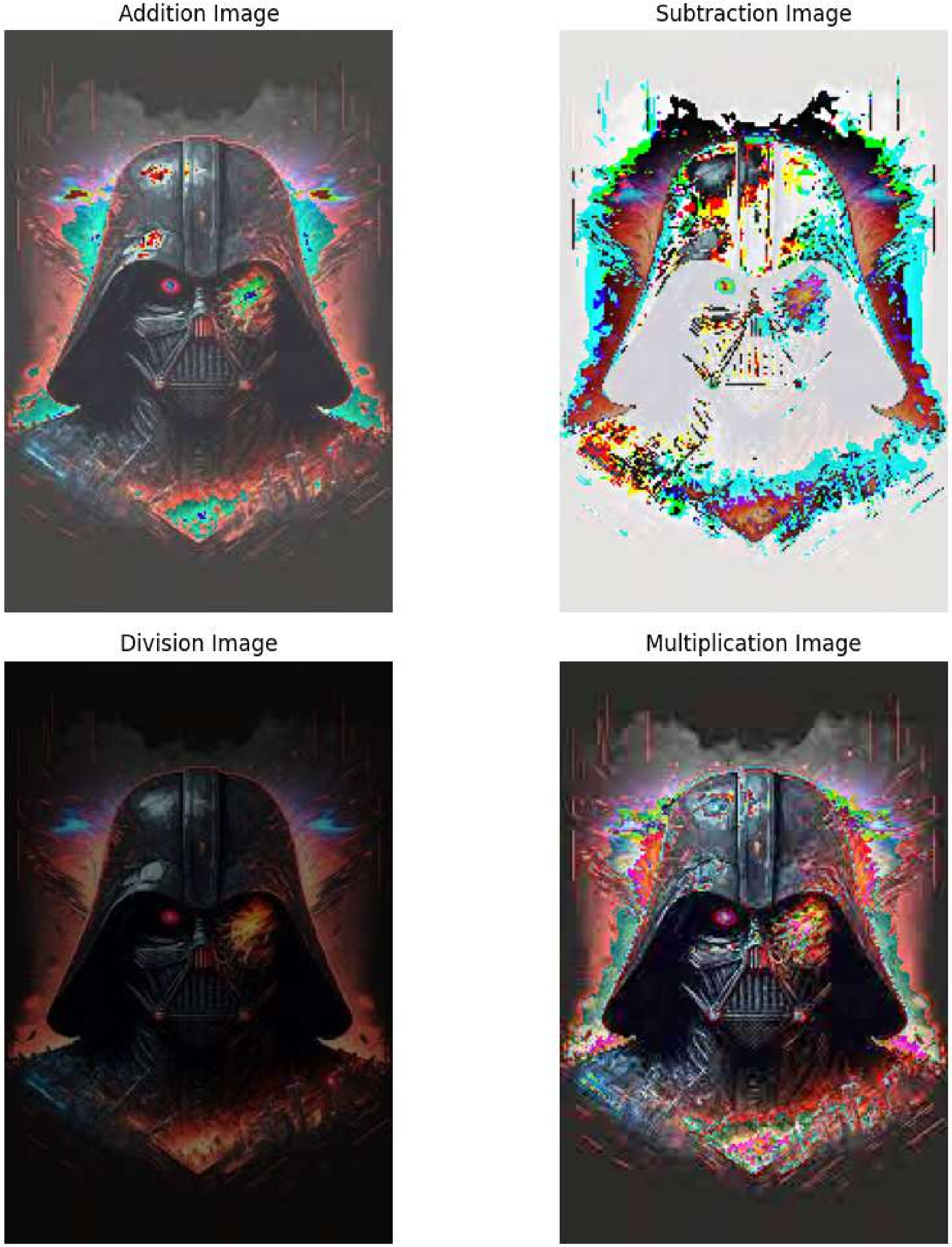
In [ ]:

```
# Define the coordinates of the region to crop
x = 100  # starting x-coordinate
y = 100  # starting y-coordinate
width = 200  # width of the cropped region
height = 200  # height of the cropped region

# Crop the image
cropped_image = image[y:y+height, x:x+width]

# Display the cropped image
plt.imshow(cropped_image)
plt.axis('off')
plt.show()
```

In [ ]:

```python
import numpy as np

# Add a constant value to each pixel
addition_image = image + 50
subtraction_image = image - 50
division_image = image // 2
multiplication_image = image * 2

# Create a figure with subplots
fig, axes = plt.subplots(2, 2, figsize=(10, 10))

# Display the addition image
axes[0, 0].imshow(addition_image)
axes[0, 0].set_title('Addition Image')
axes[0, 0].axis('off')

# Display the subtraction image
axes[0, 1].imshow(subtraction_image)
axes[0, 1].set_title('Subtraction Image')
axes[0, 1].axis('off')

# Display the division image
axes[1, 0].imshow(division_image)
axes[1, 0].set_title('Division Image')
axes[1, 0].axis('off')

# Display the multiplication image
axes[1, 1].imshow(multiplication_image)
axes[1, 1].set_title('Multiplication Image')
axes[1, 1].axis('off')

# Adjust the spacing between subplots
```

```
plt.tight_layout()

# Show the plot
plt.show()
```

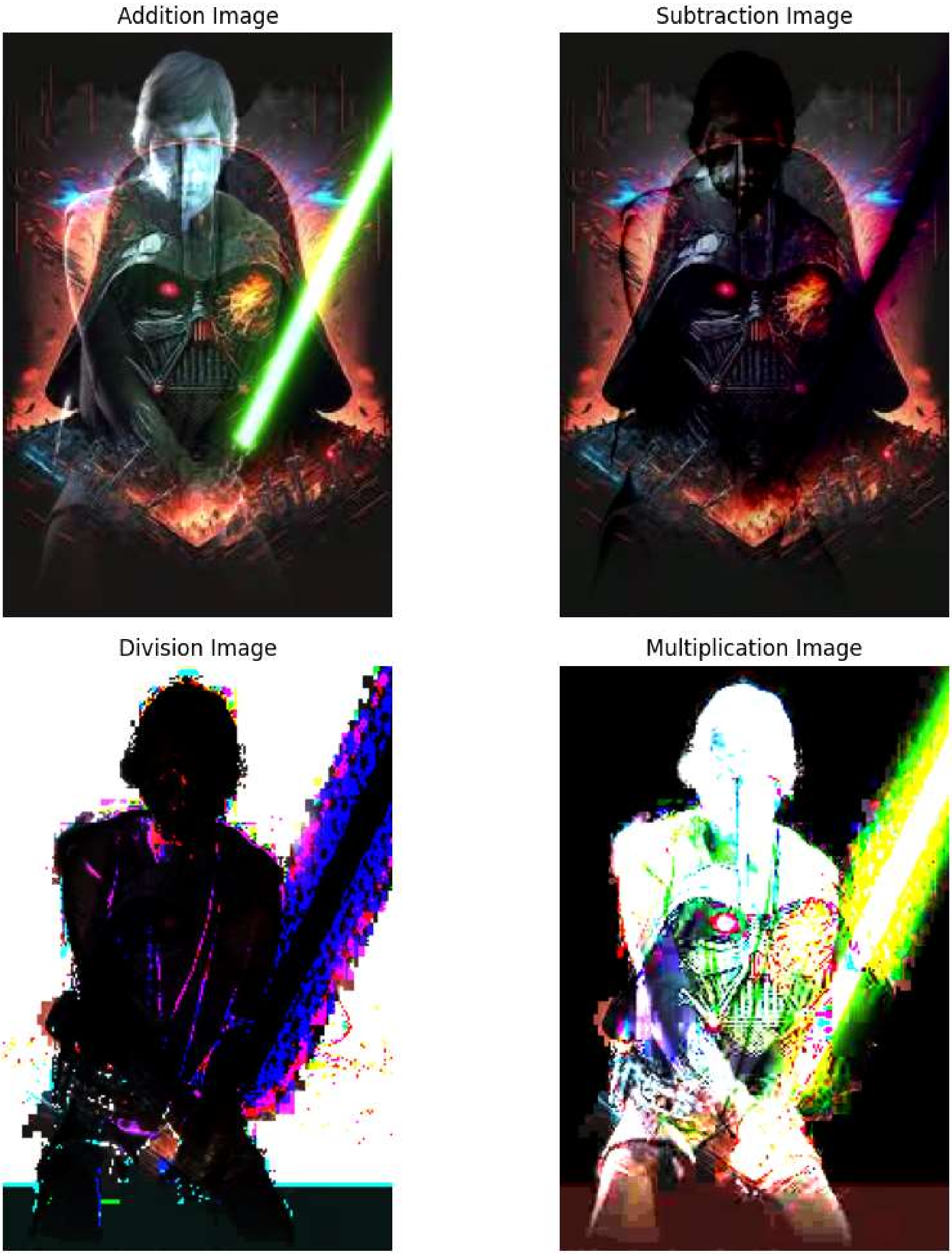


In [ ]:

```
img2 = cv2.imread('test_image_2.jpg')
img2 = cv2.resize(img2, (image.shape[1], image.shape[0]))
```

```
images_added = cv2.add(image, img2)
images_subtracted = cv2.subtract(image, img2)
images_divided = cv2.divide(image, img2, dtype=cv2.CV_32F)
images_multiplied = cv2.multiply(image, img2)

images_added_result = np.clip(images_added, 0, 255).astype(np.uint8)
images_subtracted_result = np.clip(images_subtracted, 0, 255).astype(np.uint8)
images_divided_result = np.clip(images_divided, 0, 255).astype(np.uint8)
images_multiplied_result = np.clip(images_multiplied, 0, 255).astype(np.uint8)


# Create a figure with subplots
fig, axes = plt.subplots(2, 2, figsize=(10, 10))

# Display the addition image
axes[0, 0].imshow(images_added_result)
axes[0, 0].set_title('Addition Image')
axes[0, 0].axis('off')

# Display the subtraction image
axes[0, 1].imshow(images_subtracted_result)
axes[0, 1].set_title('Subtraction Image')
axes[0, 1].axis('off')

# Display the division image
axes[1, 0].imshow(images_divided_result)
axes[1, 0].set_title('Division Image')
axes[1, 0].axis('off')

# Display the multiplication image
axes[1, 1].imshow(images_multiplied_result)
axes[1, 1].set_title('Multiplication Image')
axes[1, 1].axis('off')

# Adjust the spacing between subplots
plt.tight_layout()

# Show the plot
plt.show()
```

```
C:\Users\a21ma\AppData\Local\Temp\ipykernel_91712\536329118.py:11: RuntimeWarning: i
nvalid value encountered in cast
  images_divided_result = np.clip(images_divided, 0, 255).astype(np.uint8)
```



Addition Image

Subtraction Image

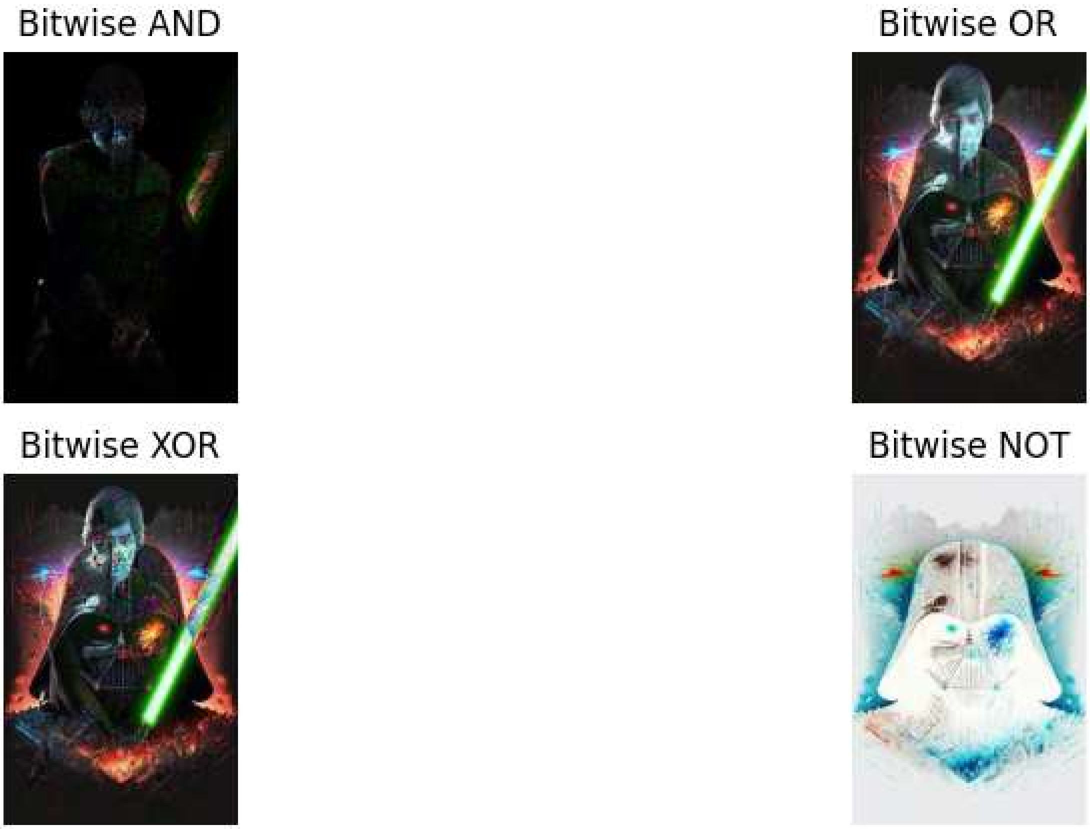
Division Image

Multiplication Image

In [ ]:

```
# Perform bitwise AND operation
result_and = cv2.bitwise_and(image, img2)

# Perform bitwise OR operation
result_or = cv2.bitwise_or(image, img2)

# Perform bitwise XOR operation
result_xor = cv2.bitwise_xor(image, img2)
```

```
# Perform bitwise NOT operation
result_not = cv2.bitwise_not(image)

# Display the results
plt.figure(figsize=(10, 5))

plt.subplot(2, 2, 1)
plt.imshow(result_and)
plt.title('Bitwise AND')
plt.axis('off')

plt.subplot(2, 2, 2)
plt.imshow(result_or)
plt.title('Bitwise OR')
plt.axis('off')

plt.subplot(2, 2, 3)
plt.imshow(result_xor)
plt.title('Bitwise XOR')
plt.axis('off')

plt.subplot(2, 2, 4)
plt.imshow(result_not)
plt.title('Bitwise NOT')
plt.axis('off')

plt.show()
```


Bitwise AND
Bitwise OR
Bitwise XOR
Bitwise NOT