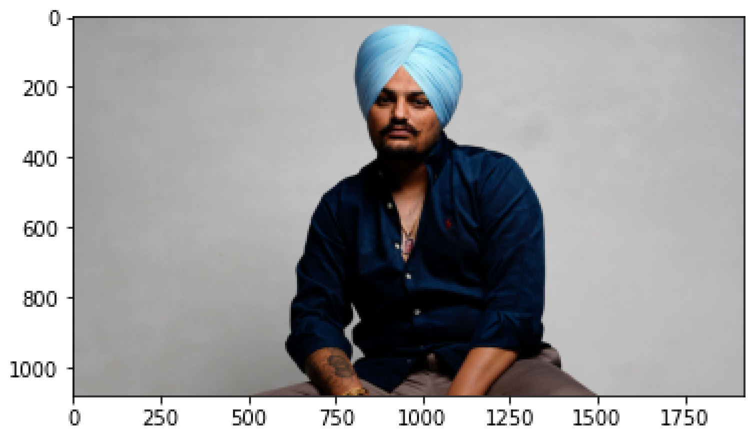
2. Image Processing (using OpenCV)

In [1]:

```
import cv2
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
import numpy as np
```

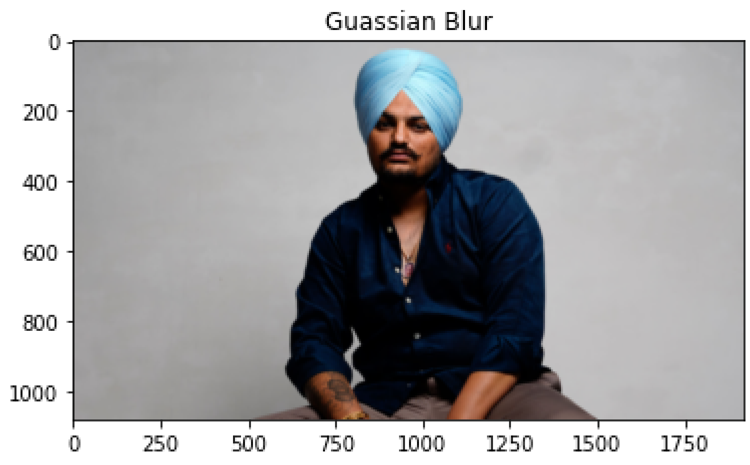
In [2]:

```
file = "jet.jpg"
img1 = cv2.imread(file)
img = cv2.cvtColor(img1, cv2.COLOR_BGR2RGB)
imgplot = plt.imshow(img)
plt.show()
```



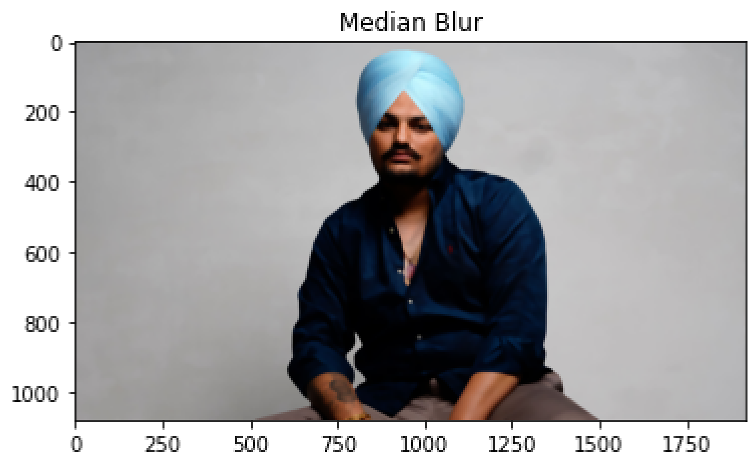
In [3]:

```
#basic bluring operations on Image
blur = cv2.GaussianBlur(img,(9,9),0)
plt.imshow(blur),plt.title('Guassian Blur')
plt.show()
```



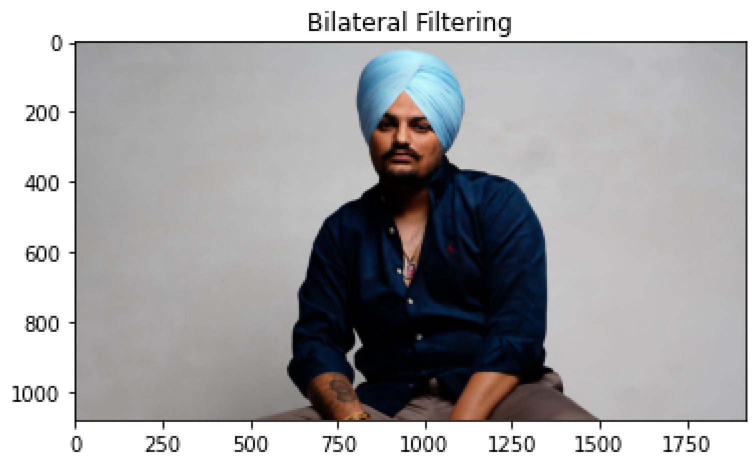
In [4]:

```
median = cv2.medianBlur(img,13)
plt.imshow(median),plt.title('Median Blur')
plt.show()
```



In [5]:

```
Bilateral_Filtering = cv2.bilateralFilter(img,9,75,75)
plt.imshow(Bilateral_Filtering),plt.title('Bilateral Filtering')
plt.show()
```

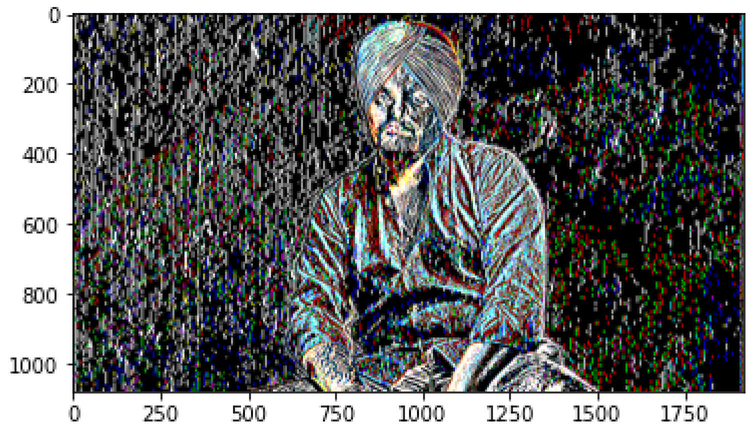


In [6]:

```
#Sobel Edge Detection
sobelx = cv2.Sobel(src=img, ddepth=cv2.CV_64F, dx=1, dy=0, ksize=5)
sobely = cv2.Sobel(src=img, ddepth=cv2.CV_64F, dx=0, dy=1, ksize=5)
sobelxy = cv2.Sobel(src=img, ddepth=cv2.CV_64F, dx=1, dy=1, ksize=5)

plt.imshow(sobelx)
plt.show()
```

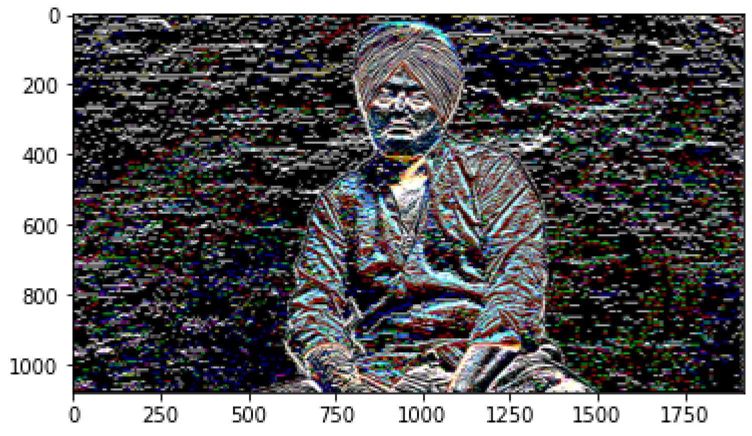
```
Clipping input data to the valid range for imshow with RGB data ([0..1] for
floats or [0..255] for integers).
```



In [7]:

```
plt.imshow(sobely)
plt.show()
```

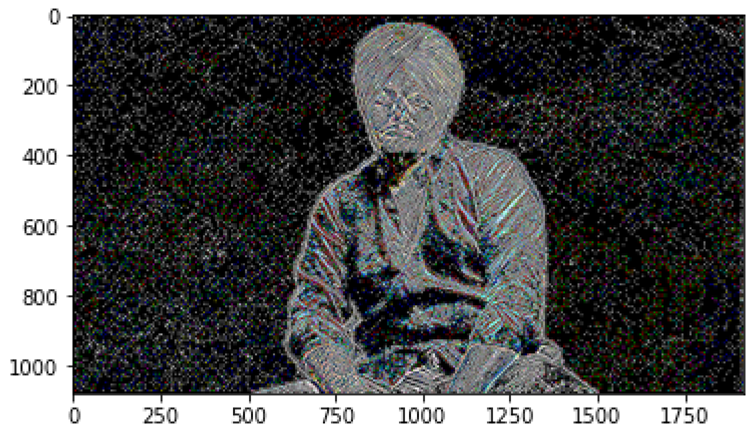
```
Clipping input data to the valid range for imshow with RGB data ([0..1] for
floats or [0..255] for integers).
```



In [8]:

```
plt.imshow(sobelxy)
plt.show()
```

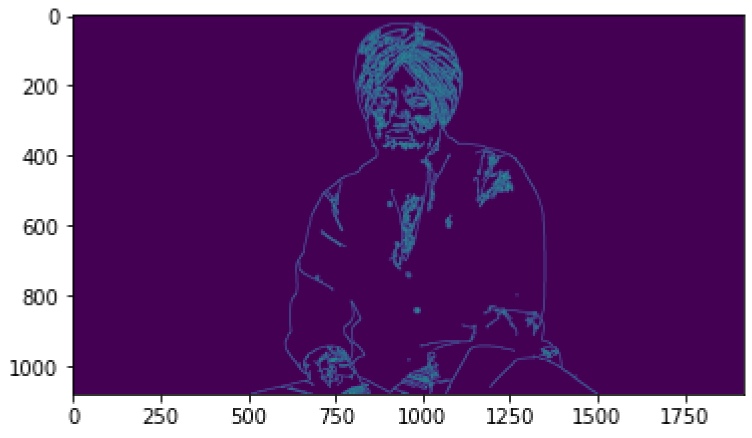
```
Clipping input data to the valid range for imshow with RGB data ([0..1] for
floats or [0..255] for integers).
```



Edge Detection

In [9]:

```
# Canny Edge Detection
edges = cv2.Canny(image=img, threshold1=10, threshold2=250)
plt.imshow(edges)
plt.show()
```



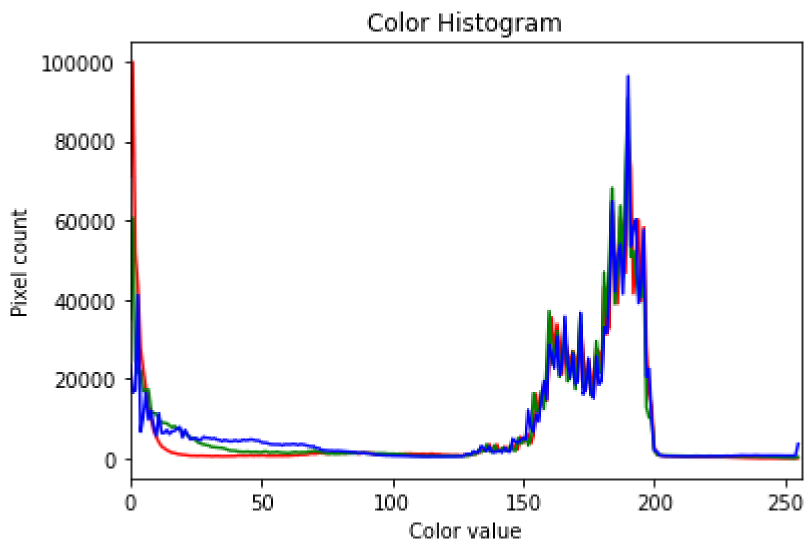
In [10]:

```
colors = ("red", "green", "blue")
plt.figure()
plt.xlim([0, 256])
for i, color in enumerate(colors):
    histogram, bin_edges = np.histogram(
        img[:, :, i], bins=256, range=(0, 256)
    )
    plt.plot(bin_edges[0:-1], histogram, color=color)

plt.title("Color Histogram")
plt.xlabel("Color value")
plt.ylabel("Pixel count")
```

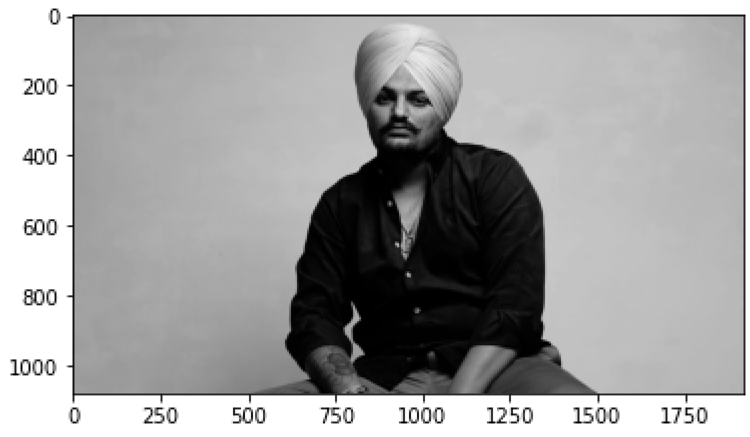
Out[10]:

```
Text(0, 0.5, 'Pixel count')
```



In [11]:

```
img_gray = cv2.cvtColor(img1, cv2.COLOR_BGR2GRAY)
plt.imshow(img_gray,cmap='gray')
plt.show()
```



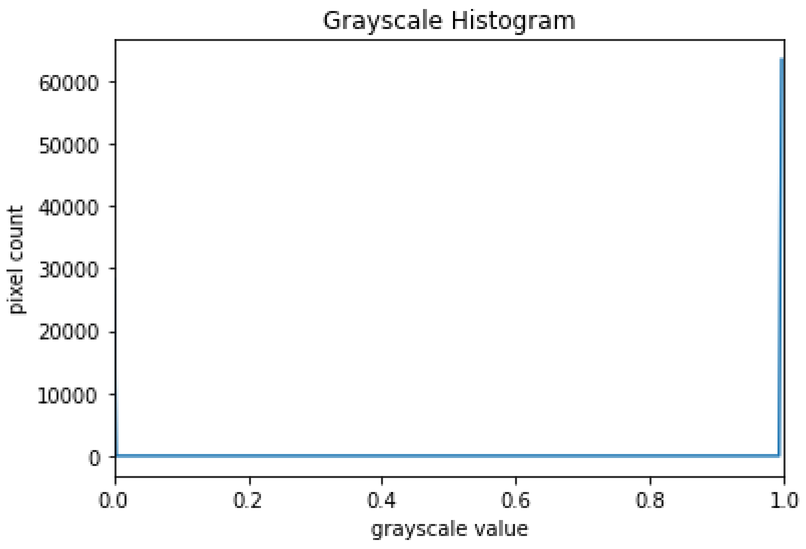
In [12]:

```
histogram, bin_edges = np.histogram(img_gray, bins=256, range=(0, 1))
plt.figure()
plt.title("Grayscale Histogram")
plt.xlabel("grayscale value")
plt.ylabel("pixel count")
plt.xlim([0.0, 1.0])

plt.plot(bin_edges[0:-1], histogram)
```

Out[12]:

```
[<matplotlib.lines.Line2D at 0x2406fb68f10>]
```



Thresholding

In [13]:

```
# threshoding
ret,thresh1 = cv2.threshold(img_gray,127,255,cv2.THRESH_BINARY)
ret,thresh2 = cv2.threshold(img_gray,127,255,cv2.THRESH_BINARY_INV)
ret,thresh3 = cv2.threshold(img_gray,127,255,cv2.THRESH_TRUNC)
ret,thresh4 = cv2.threshold(img_gray,127,255,cv2.THRESH_TOZERO)
ret,thresh5 = cv2.threshold(img_gray,127,255,cv2.THRESH_TOZERO_INV)
titles = ['Original Image','BINARY','BINARY_INV','TRUNC','TOZERO','TOZERO_INV']
images = [img, thresh1, thresh2, thresh3, thresh4, thresh5]
for i in range(6):
    plt.subplot(2,3,i+1),plt.imshow(images[i],'gray',vmin=0,vmax=255)
    plt.title(titles[i])
    plt.xticks([]),plt.yticks([])
plt.show()
```