Rahma Fairuz Rania J0303201065 TPL 57 A2

1. Gaussian filter scipy

In [1]:
```
from scipy import misc,ndimage
from matplotlib import pyplot as plt
```

In [2]:
```
# import gambar dari library scipy
face = misc.face()
blurred_face = ndimage.gaussian_filter(face, sigma=3)
fig, ax = plt.subplots(1, 2, figsize=(16, 8)) # figur 1 baris 2 kolom

# buat perbandingan img yang diberi filter gaussian blur dan yang original
ax[0].imshow(face)
ax[0].set_title("Original Image")
ax[0].set_xticks([])
ax[0].set_yticks([])
ax[1].imshow(blurred_face)
ax[1].set_title("Blurred Image")
ax[1].set_xticks([])
ax[1].set_yticks([])
```

Out[2]: []

2. Linier filtering

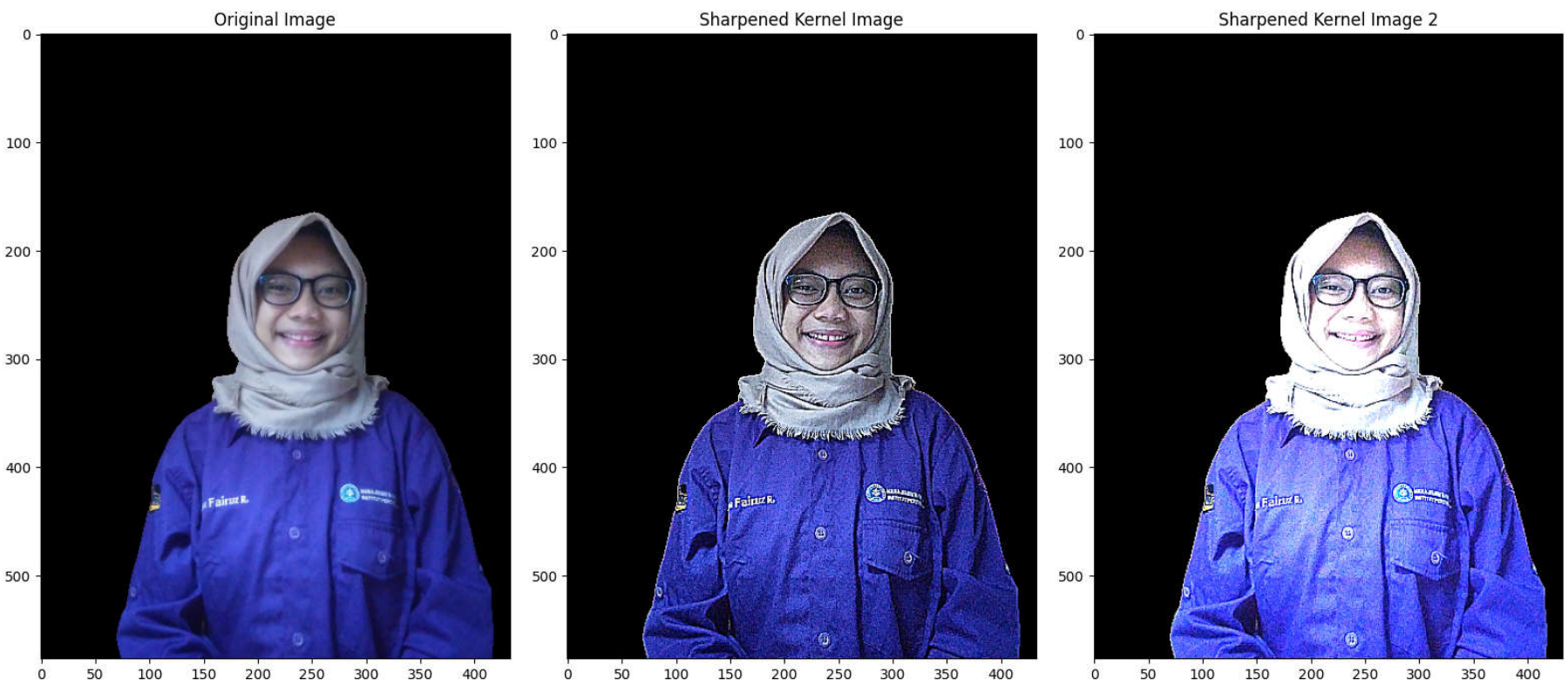
In [3]:
```
import cv2 as cv
import numpy as np
import matplotlib.pyplot as plt
```

In [6]:
```
image = cv.imread("dt/rahma.png")
fig, ax = plt.subplots(1, 3, figsize=(16, 8)) # figur 1 baris 3 kolom
fig.tight_layout()

# To conovolve the kernel on an image we can use cv.filter2D
# sharpen filter
ax[0].imshow(cv.cvtColor(image, cv.COLOR_BGR2RGB))
ax[0].set_title('Original Image')
kernel_sharpening = np.array([[-1, -1, -1],
 [-1, 9, -1],
 [-1, -1, -1]])
kernel_sharpening_2 = np.array([[-1, -1, -1],
 [-1, 10, -1],
 [-1, -1, -1]])

sharpened = cv.filter2D(image, -1, kernel_sharpening)
ax[1].imshow(cv.cvtColor(sharpened, cv.COLOR_BGR2RGB))
ax[1].set_title('Sharpened Kernel Image')

sharpened_2 = cv.filter2D(image, -1, kernel_sharpening_2)
ax[2].imshow(cv.cvtColor(sharpened_2, cv.COLOR_BGR2RGB))
ax[2].set_title('Sharpened Kernel Image 2')
plt.show()
```



3. Manipulation w/ PIL

In [7]:
```
from PIL import Image
im = Image.open("dt/rahma.png")
im.getbands( ) # lihat channel pada gambar
```

Out[7]:
```
('R', 'G', 'B', 'A')
```

In [8]:
```
im.size
```

Out[8]:
```
(433, 577)
```

In [10]:
```
import base64
with open("dt/rahma.png", "rb") as image:
 image_string = base64.b64encode(image.read()) # img encoding

import io
image = io.BytesIO(base64.b64decode(image_string))
Image.open(image)
```

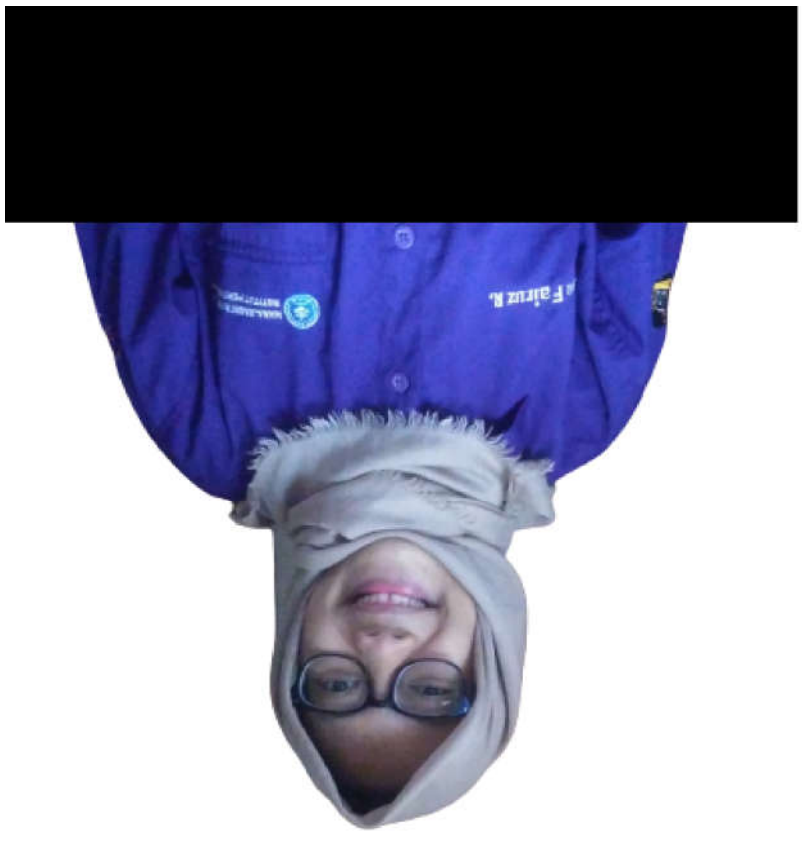
Out[10]:

In [12]:
```
# crop img berdasarkan value left, upper, right, and lower pixel coordinate pada var box
im = Image.open('dt/rahma.png')
box = (100, 150, 300, 300)
cropped_image = im.crop(box)
cropped_image
```

Out[12]:

In [59]:
```
# putar 180 derajat
rotated = im.rotate(180)
rotated
```

Out[59]:



In [121]:
```
# logo = Image.open('dt/rahma.png') # gabisa krn ada transparency

logo = Image.open('dt/ip.png')
```

In [128]:
```
logo = logo.resize((50, 50))
```

In [104]:
```
print(logo.size)
```

```
(1770, 416)
```

In [129]:
```
# atur posisi objek ke-2 (logo) agar posisinya sesuai pada gambar utama
position = (38, 469)
im.paste(logo, position)
im.save('merged.png')
```

In [127]:
```
position
```

Out[127]:
```
(-1615, -1471)
```

In [130]:
```
image_copy.paste(logo, position, logo)
```

In [131]:
```
# insert gambar lain ke dalam gambar
im = Image.open("dt/rahma.png")
image_copy = im.copy()
position = ((image_copy.width - logo.width),
(image_copy.height - logo.height))
image_copy.paste(logo, position,logo)
image_copy
```

Out[131]:

In [36]:
```
# menambah kualitas gambar
from PIL import ImageEnhance
enhancer = ImageEnhance.Sharpness(im)
enhancer.enhance(2.0)
```

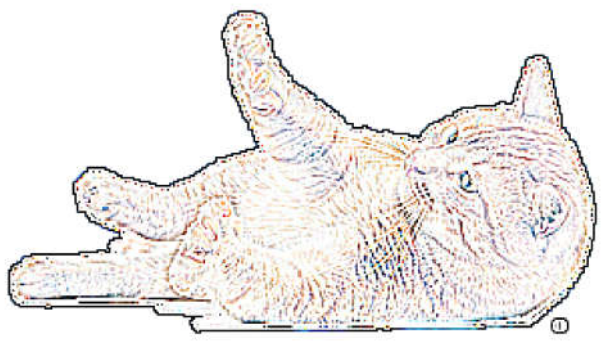
Out[36]:

In [37]:
```
# tambahkan contrast level pada gambar
enhancer = ImageEnhance.Contrast(im)
enhancer.enhance(2)
```

Out[37]:

In [40]:
```
# filter dengan laplace
from PIL import ImageFilter
im2 = Image.open('dt/meng.png')
im2.filter(ImageFilter.CONTOUR)
```

Out[40]:

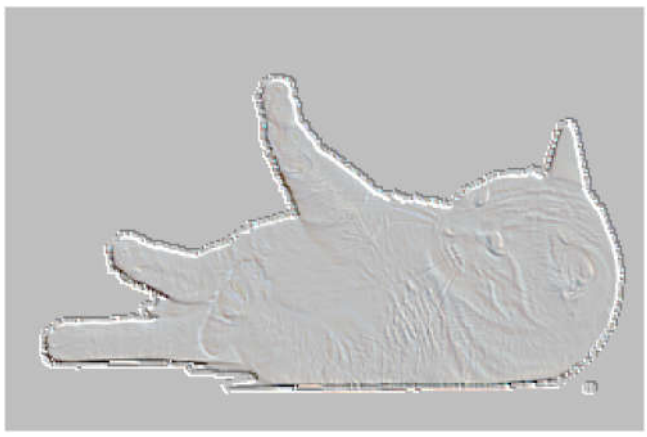


In [41]:
```
# filter mempertajam garis object (edge) pada gambar
im2.filter(ImageFilter.EDGE_ENHANCE)
```

Out[41]:

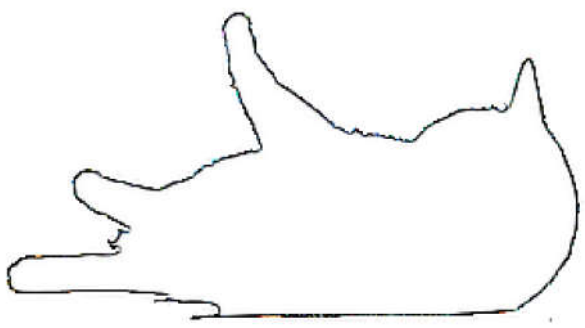
In [42]:
```
# filter emboss, menampilkan object dengan batas edge
im2.filter(ImageFilter.EMBOSS)
```

Out[42]:



In [43]:
```
# filter image edge detectionS
im2.filter(ImageFilter.FIND_EDGES)
```

Out[43]:



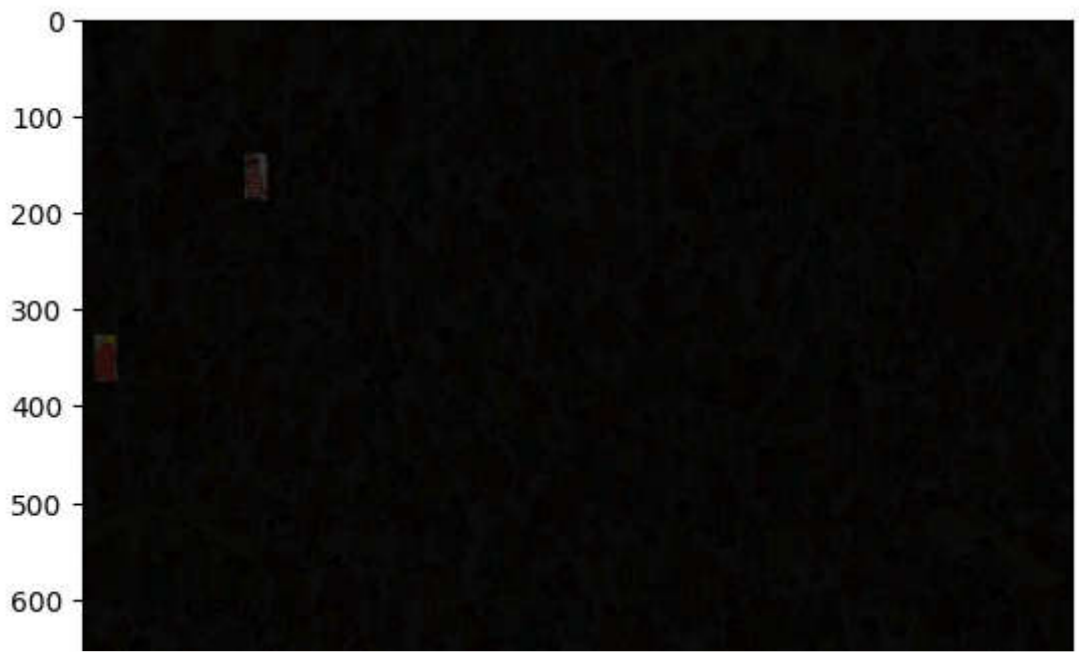
In [47]:
```
from PIL import ImageSequence
```

In [48]:
```
frame_num = 1
for frame in ImageSequence.Iterator(im2):
 frame.save("frame%d.png" % frame_num)
 frame_num = frame_num + 1
 if frame_num == 3:
     break
```

4. Temukan wally d tengah keramaian

In [49]:
```
from pylab import imshow, show
import mahotas
import mahotas.demos
import numpy as np
```

In [50]:
```
# lib mahotas untuk mendeteksi objek
wally = mahotas.demos.load('Wally')
wfloat = wally.astype(float)
r,g,b = wfloat.transpose((2,0,1))
w = wfloat.mean(2)
pattern = np.ones((24,16), float)
for i in range(2):
 pattern[i::4] = -1
 v = mahotas.convolve(r-w, pattern)
 mask = (v == v.max())
 mask = mahotas.dilate(mask,
np.ones((48,24)))
 np.subtract(wally, .8*wally *
~mask[:,:,None], out=wally, casting='unsafe')
imshow(wally)
show()
```



In [54]:
```
imshow(mahotas.demos.load('Wally'))
```

Out[54]: <matplotlib.image.AxesImage at 0x2414aad35b0>