Computer Vision And Image Processing
Project 1

Hariprasath Parthasarathy
Department of Computer Science
Person Number:50289737
hparthas@buffalo.edu

1.Edge Detection - With sobel filtering

Edge detection reduces the overhead of processing huge amounts of data in image processing. Edge detection in sobel filtering is done by detecting changes in frequencies, A X . It works by approximately detecting the gradient of the images. The sobel filter is as follows

The steps for sobel filtering is as follows

1. apply convolution of sobel matrix - in x axis
2. apply convolution of sobel matrix - in y-axis
3. apply gradient with results of 1 and 2

the X sobel matrix

-1	0	+1
-2	0	+2
-1	0	+1

Gx

the y sobel matrix is

+1	+2	+1
0	0	0
-1	-2	-1

Gy

The output is as follows

Sobel X

Sobel Y

code

```
from lib import basics

import cv2
import numpy as np
import math
img = cv2.imread("task1.png",0)
kernel_v = np.array([[1,2,1],[0,0,0],[-1,-2,-1]])
kernel_h = np.array([[1,0,-1],[2,0,-2],[1,0,-1]])

edge_detected_image_v = basics.apply_convolution1(img,kernel_v)
edge_detected_image_h = basics.apply_convolution1(img,kernel_h)

cv2.imshow('img',img)
# cv2.imshow('blurred_image',blurred_image)
cv2.imshow('edge_detected_image_v',edge_detected_image_v)
cv2.imshow('edge_detected_image_h',edge_detected_image_h)
cv2.imshow('gradient',basics.find_gradient(edge_detected_image_v,edge_detected_image_h))
cv2.waitKey()
def apply_convolution1(image,kernel,size=3):
    rows,cols = image.shape
    new_image = np.zeros((rows,cols), dtype='uint8')
    for i in range(0,3):
        for j in range(0,3):
            total = 0
            for k in range(0,size):
                for m in range(0,size):
                    total = total + kernel[k,m] * image[i +k,j+m]
            new_image[i+math.floor(size/2)+1,j+math.floor(size/2)+1] = total

    return new_image
def find_gradient(image1,image2):
    rows,cols = image1.shape[0],image1.shape[1]
    for i in range(rows):
        for j in range(cols):
            image1[i,j] = math.sqrt(image1[i,j]**2 + image2[i,j]**2)
    return image1
```

Task 2: Key point detection

Keypoints are major points of interest in image processing

Keypoint detection is done by the method of scalar invariant fourier transform by the following steps

1. the images are downscaled to 4 octaves .
2. In each octave multiple images the gaussian blurring is applied progressively with 5 sigma values
3. the difference of gaussian for the images are found with respect to the next image in the same octave resulting in 4 difference of gaussian images for each octave
4. these difference of gaussian images are used to detect keypoints by comparing the the each other on ignoring the outermost octave.we compare with matrices above and below we get a list of keypoints. the key points of higher score are obtained by thresholding which ignore key points of no interest

output

output of keypoints

13,22

8,23

7,24

25,22

23,36

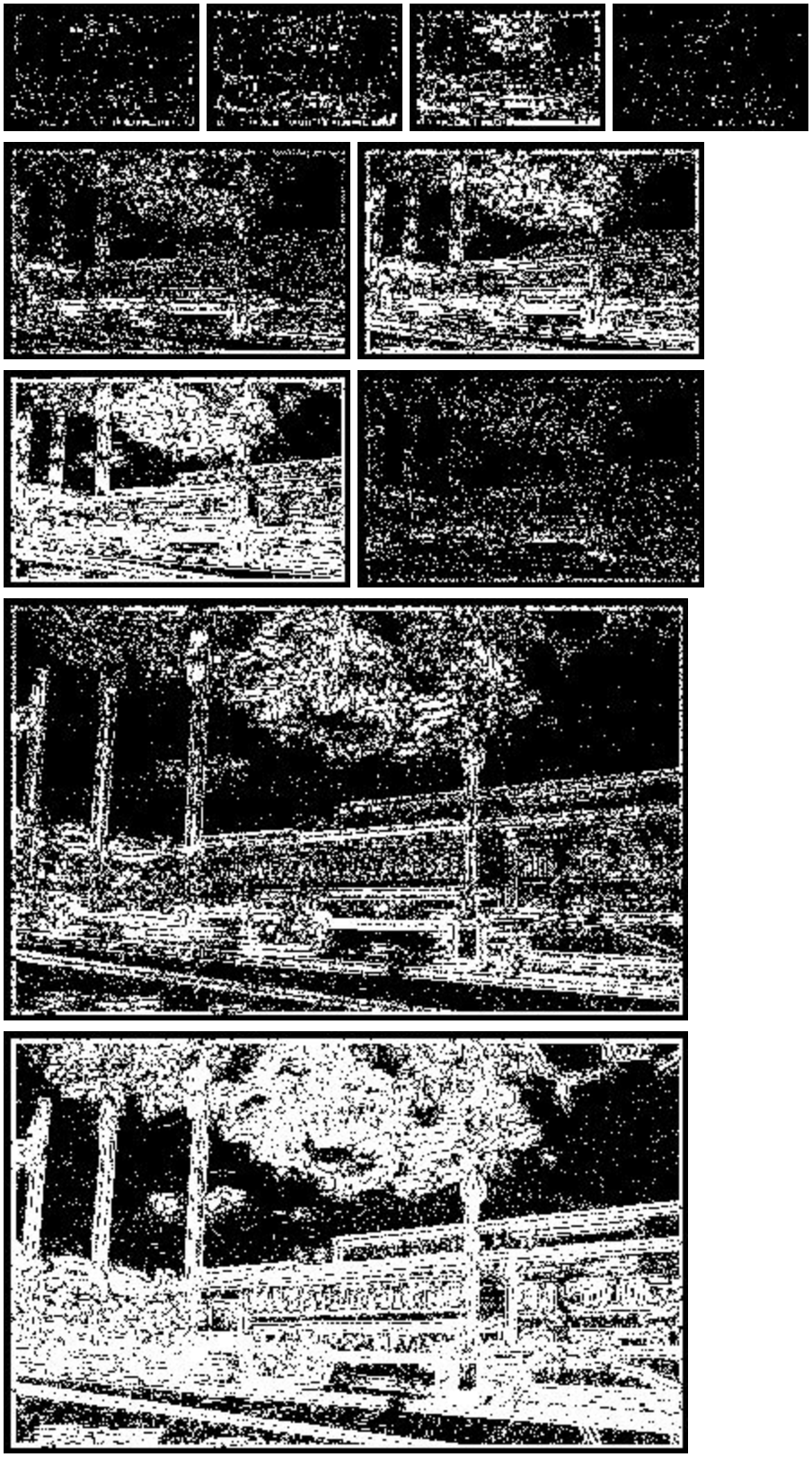
octave 4

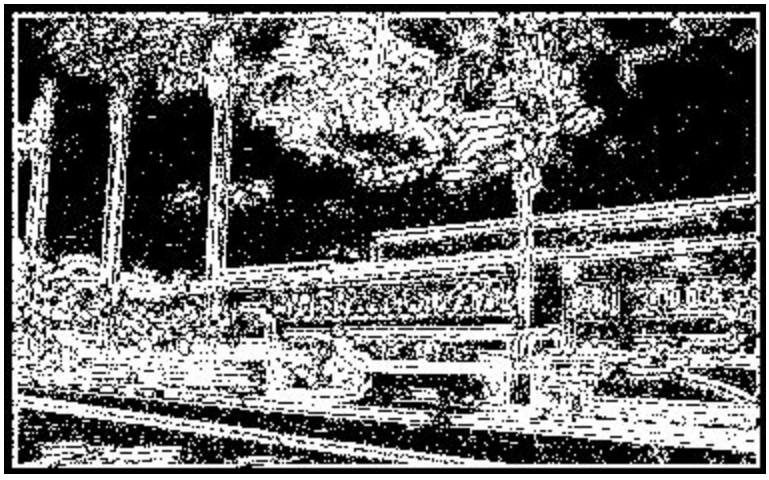
octave 3

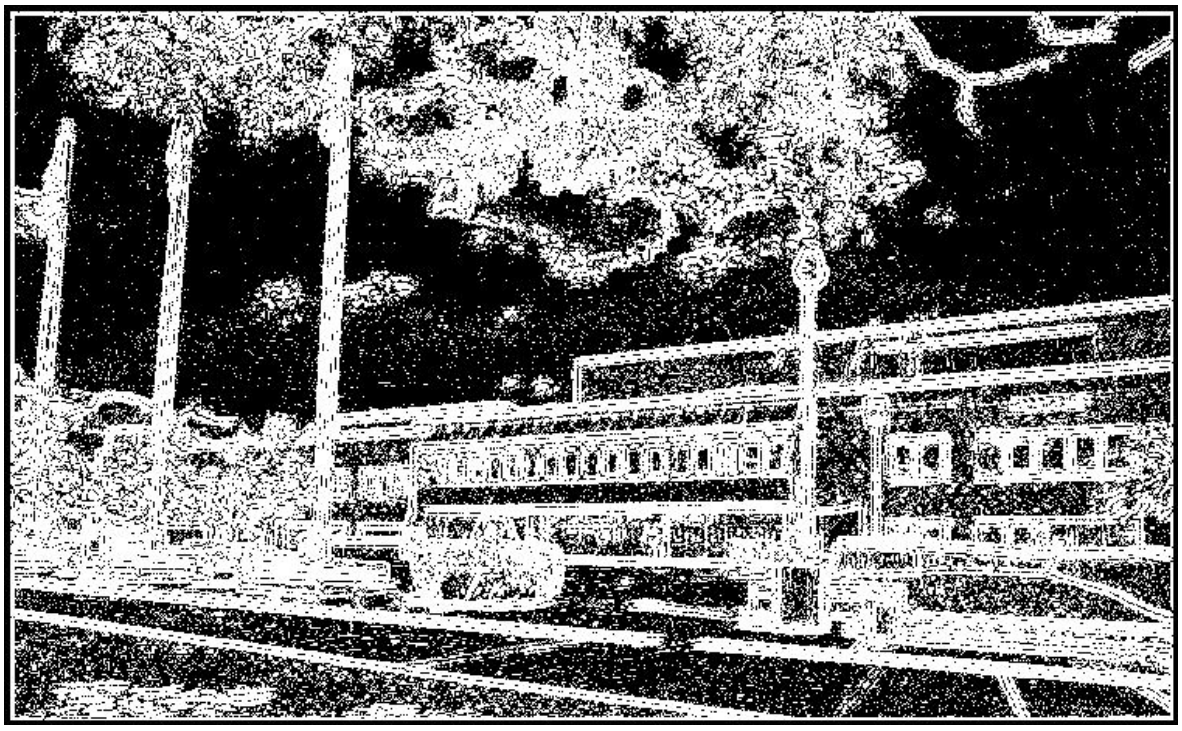
octave 2

ocatve 1

the difference of gaussian







code:

```
from lib import basics
import numpy as np
import cv2
import math
rgb_image = cv2.imread("task2.jpg")
original_image = cv2.imread("task2.jpg",0)

img_scaled = np.empty(shape=5,dtype=object)
img_scaled[0] = original_image.copy()
img_scaled[1] =  np.array(basics.scale_half(img_scaled[0])).copy()
img_scaled[2] =  np.array(basics.scale_half(img_scaled[1])).copy()
img_scaled[3] =  np.array(basics.scale_half(img_scaled[2])).copy()


gaussian_matrix = [[1/math.sqrt(2),1,math.sqrt(2),2,2*math.sqrt(2)],
                   [math.sqrt(2),2,2*math.sqrt(2),4,4*math.sqrt(2)],
                   [2*math.sqrt(2),4,4*math.sqrt(2),8,8*math.sqrt(2)],
                   [4*math.sqrt(2),8,8*math.sqrt(2),16,16*math.sqrt(2)]
                   ]
image_array = np.empty((4,5), dtype = object)
DoG_array = np.empty((4,4),dtype = object)
for i in range(0,4):
    for j in range(0,5):
        gaussian_kernel = basics.calculate_gaussian(gaussian_matrix[i][j],7)
        image_array[i,j] =
basics.apply_convolution(basics.add_padding(img_scaled[i],7//2),gaussian_kernel,7)
        if j > 0:
```

```
            DoG_array[i,j-1] = basics.computer_difference(image_array[i,j],image_array[i,j-1])
        print(gaussian_kernel)

key_point_array = np.empty((4,2),dtype = object)
for j in range (0,2):
    key_point_array_1 = basics.find_keys(DoG_array[0][0],DoG_array[0][1],DoG_array[0][2])
    key_point_array_2 = basics.find_keys(DoG_array[0][1],DoG_array[0][2],DoG_array[0][3])
    key_point_array_21 =
basics.find_keys(DoG_array[1][0],DoG_array[1][1],DoG_array[1][2])
    key_point_array_22 =
basics.find_keys(DoG_array[1][1],DoG_array[1][2],DoG_array[1][3])
    key_point_array_31 =
basics.find_keys(DoG_array[2][0],DoG_array[2][1],DoG_array[2][2])
    key_point_array_32 =
basics.find_keys(DoG_array[2][1],DoG_array[2][2],DoG_array[2][3])
    key_point_array_41 =
basics.find_keys(DoG_array[3][0],DoG_array[3][1],DoG_array[3][2])
    key_point_array_42 =
basics.find_keys(DoG_array[3][1],DoG_array[3][2],DoG_array[3][3])

plot_image_1 = rgb_image.copy()
img_copy_1 =img_scaled[0].copy()
img_copy_2 =np.array(img_scaled[1]).copy()
img_copy_3 =np.array(img_scaled[2]).copy()
img_copy_4 =np.array(img_scaled[3]).copy()

for point in key_point_array_1:
        cv2.circle(img_scaled[0],point,1,(0,255,0))
for point in key_point_array_2:
       cv2.circle(img_copy_1,point,1,(255,0,0))

for point in key_point_array_21:
        cv2.circle(img_scaled[1],point,1,(0,255,0))
for point in key_point_array_22:
       cv2.circle(img_copy_2,point,1,(255,0,0))

for point in key_point_array_31:
        cv2.circle(img_scaled[2],point,1,(0,255,0))
for point in key_point_array_32:
       cv2.circle(img_copy_3,point,1,(255,0,0))

for point in key_point_array_41:
        cv2.circle(img_scaled[3],point,1,(0,255,0))
for point in key_point_array_42:
       cv2.circle(img_copy_4,point,1,(255,0,0))


cv2.imshow('image1',img_scaled[0])

cv2.imshow('image12',img_copy_1)
```

```
cv2.imshow('image21',img_scaled[1])
cv2.imshow('image22',img_copy_2)
cv2.imshow('image31',img_scaled[2])
cv2.imshow('image32',img_copy_3)
cv2.imshow('image41',img_scaled[3])
cv2.imshow('image42',img_copy_4)
cv2.waitKey(0)


def apply_convolution(image,kernel,size=3):
    rows,cols = image.shape
    new_image = np.zeros((rows,cols), dtype='float32')
    sum_total =0.0
    for i in range(0,size):
        for j in range(0,size):
          sum_total = sum_total + kernel[i][j]
    for i in range(0,rows-size):
        for j in range(0,cols-size):
            total = 0.0
            for k in range(0,size):
                for m in range(0,size):
                    total = total + kernel[k,m] * image[i +k,j+m]
            new_image[i+math.floor(size/2)+1,j+math.floor(size/2)+1] = abs((total)/(sum_total))
    return new_image
def scale_half(image):
    scaled_x = image[::2]
    scaled_xy = scaled_x[:,1::2]
    return scaled_xy


def calculate_gaussian(sigma_value,size):
    print(size)
    sum = 0.0
    print(sigma_value)
    new_kernel = np.zeros((size,size),dtype = float)
    for i in range(int(-math.floor(size/2)),int(math.floor(size/2)+1)):
        for j in range(int(-math.floor(size/2)),int(math.floor(size/2)+1))
            numerator =  math.exp(-float((i*i+j*j))/float(2*sigma_value*sigma_value))
            denominator = (2*(math.pi)* math.pow(sigma_value,2))
```

```
            new_kernel[(i+math.floor(size/2)),(j+math.floor(size/2))] = numerator/denominator
            sum = sum + new_kernel[i+math.floor(size/2),j+math.floor(size/2)]
    return new_kernel

def computer_difference(image1,image2):
    rows,cols = image1.shape
    new_image = np.zeros((rows,cols),dtype = np.float)
    for i in range(0,rows):
        for j in range (0 , cols):
            new_image[i,j] = abs(int(image1[i,j])-int(image2[i,j]))
    return new_image

def find_keys(d1,d2,d3):
    keys = []
    for i in range(1,d1.shape[0]-1):
        for j in range(1,d1.shape[1]-1):
            d11 = d1[i-1:i+2,j-1:j+2] #3X3 matrix in dog1
            d22 = d2[i-1:i+2,j-1:j+2] #3X3 matrix in dog2
            d33 = d3[i-1:i+2,j-1:j+2] #3X3 matrix in dog3
            center = d22[1,1] #Center element in dog2 - d22
            d22 = np.delete(d22[1], 1) #Delete center element from the dog2 - d22
            mn1,mx1 = d11.min(),d11.max() #Min and Max element in d11
            mn2,mx2 = d22.min(),d22.max() #Min and Max element in d22
            mn3,mx3 = d33.min(),d33.max() #Min and Max element in d33
            mins = min([mn1,mn2,mn3]) #Mins between the 3 mins above
            maxs = max([mx1,mx2,mx3]) #Maxs between the 3 maxs above
            #Keypoint should be either less than the min or greater than the max
            if((center<mins) or (center>maxs)) and center > 0.2:
                keys.append((j,i))
    return keys
def add_padding(img, padding):
    img = np.pad(img, (padding, padding), mode='constant', constant_values=0)
    return img
```

Task 3: Cursor detection

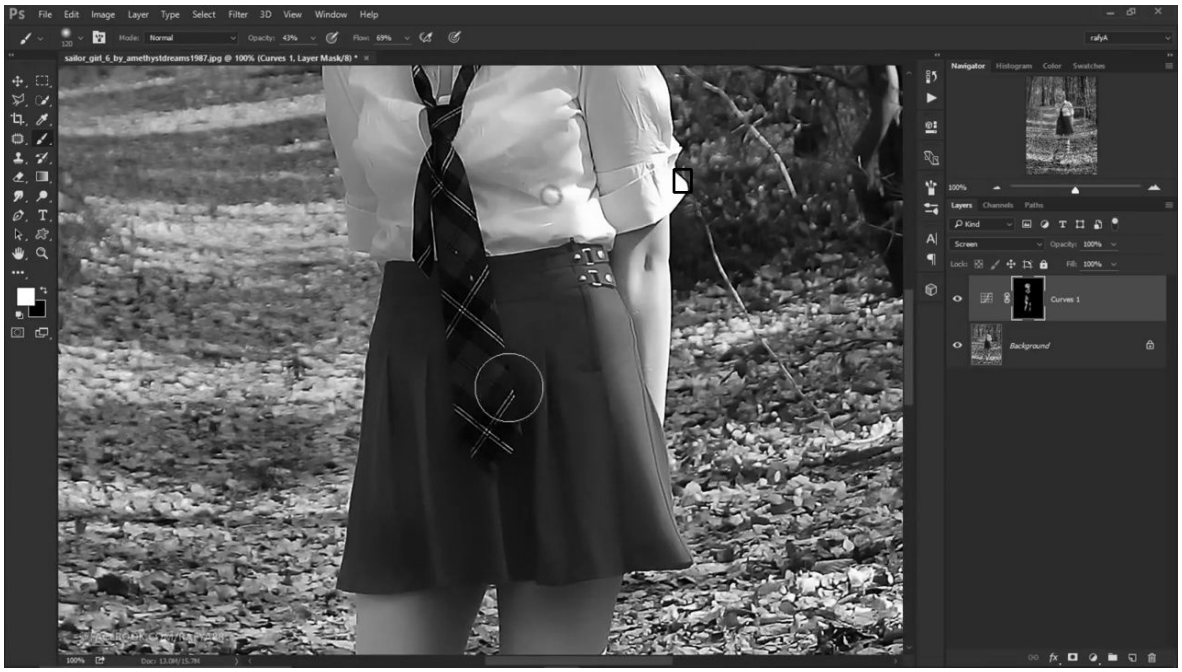
Cursor detection using template matching involves a series of steps of image processing followed by multi scale template matching

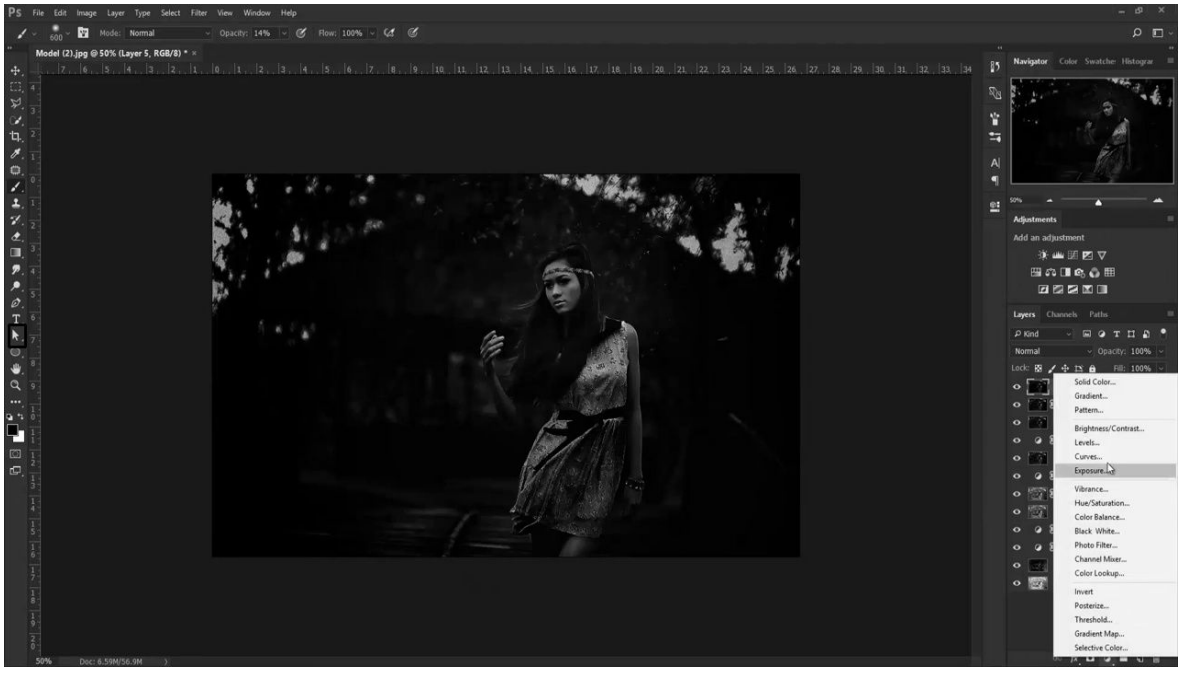
My first attempt was thresholding the images since most cursor are white. for that a lot of false positives appeared.

Thus the next attempt was made to do gaussian blur followed by laplacian transform on the images and the template. scale the template to multiple size and do template matching

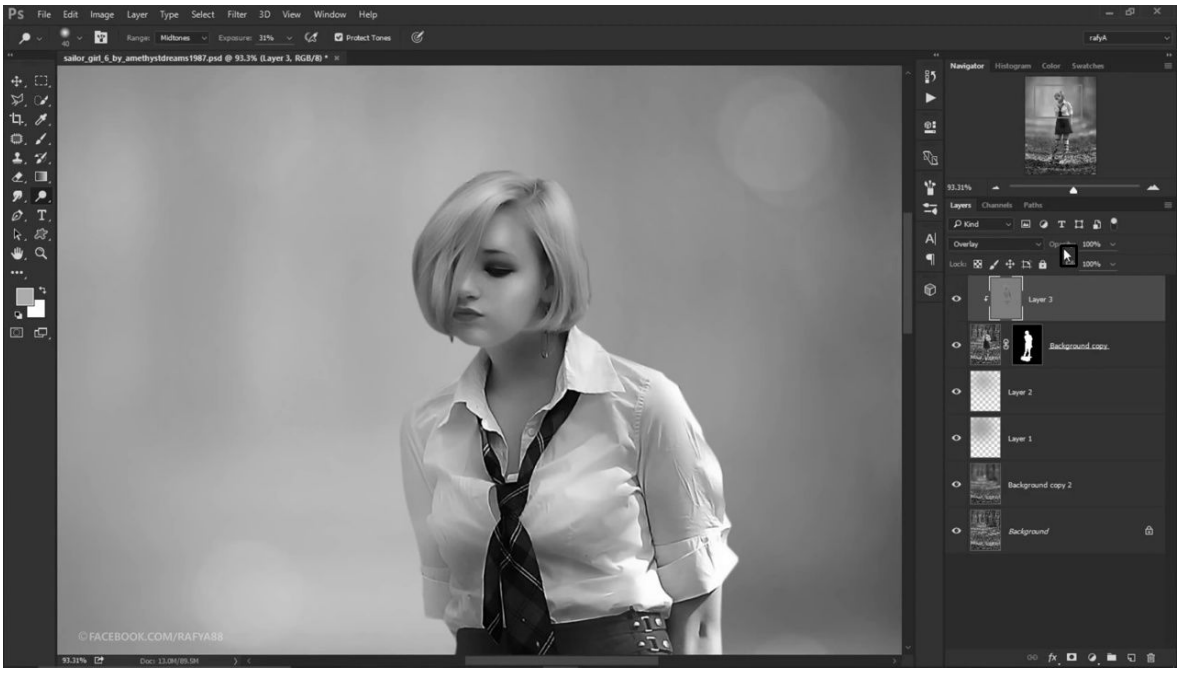
the gaussian blur would smoothen out noise in image and the laplacian transform will detect edges in the images

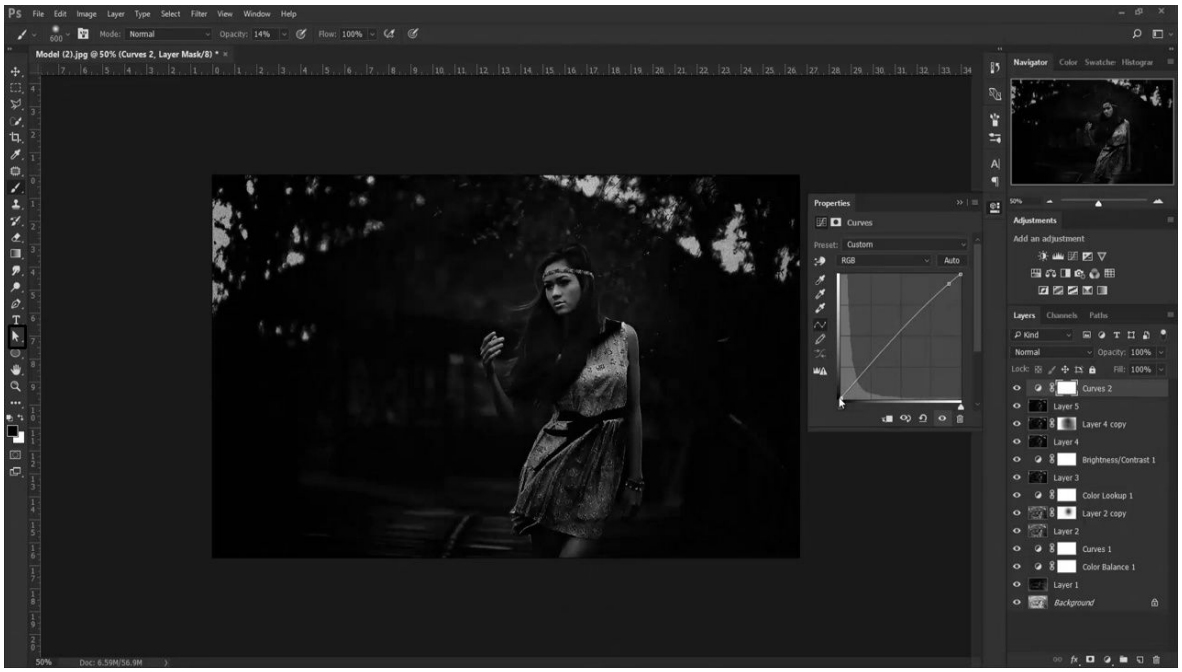
the final output is as followsc


File Edit Image Layer Type Select Filter 3D View Window Help
sailor_girl_6_by_amethystdreams1987.jpg @ 100% (Curves 1, Layer Mask/8) *
Navigator Histogram Color Swatches
Layers Channels Paths
Curves 1
Background


File Edit Image Layer Type Select Filter View Window Help
Model (2).jpg @ 50% (Layer 5, RGB/8) *
Navigator Color Swatches Histogra
Adjustments
Add an adjustment
Layers Channels Paths
Solid Color...
Gradient...
Pattern...
Brightness/Contrast...
Levels...
Curves...
Exposure...
Vibrance...
Hue/Saturation...
Color Balance...
Black White...
Photo Filter...
Channel Mixer...
Color Lookup...
Invert
Posterize...
Threshold...
Gradient Map...
Selective Color...


File Edit Image Layer Type Select Filter 3D View Window Help
bottom-of-the-sea-16913.jpg @ 51.4% (org, RGB/8#) *
Color Swatches
Adjustments Styles
Add an adjustment
Layers Channels Paths
org
Background copy
ENG 03:53 PM

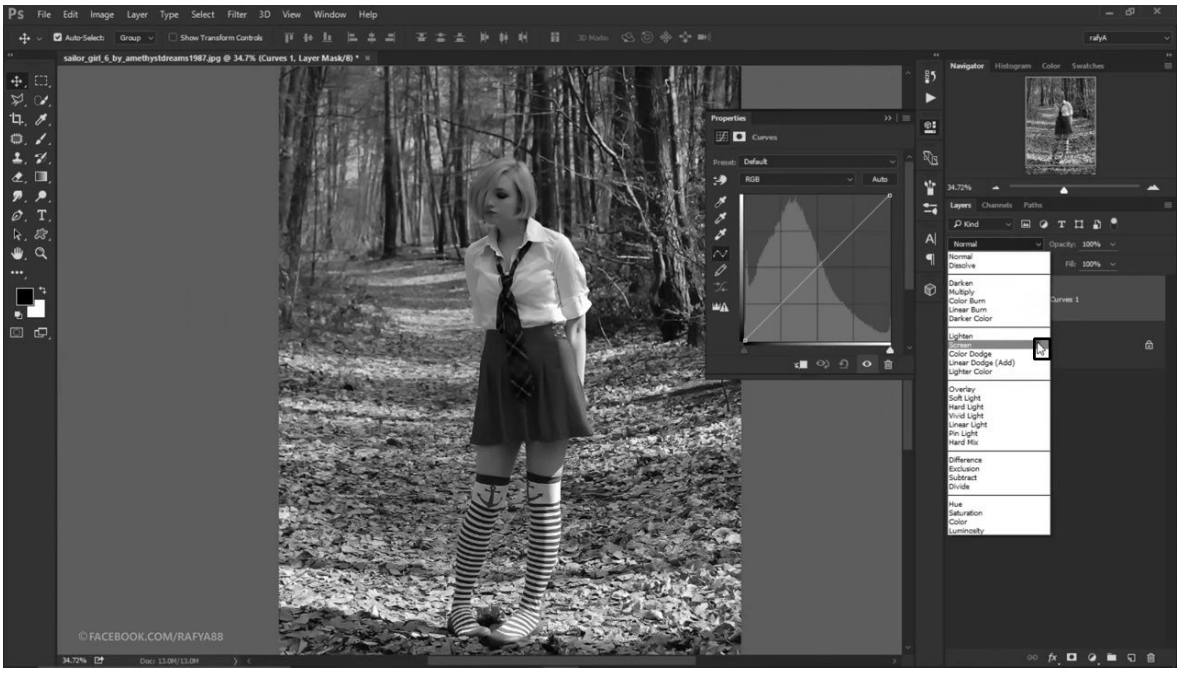

File Edit Image Layer Type Select Filter 3D View Window Help
sailor_girl_6_by_amethystdreams1987.psd @ 93.3% (Layer 3, RGB/8) *
Navigator Histogram Color Swatches
Layers Channels Paths
Layer 3
Background copy
Layer 2
Layer 1
Background copy 2
Background
© FACEBOOK.COM/RAFYA88



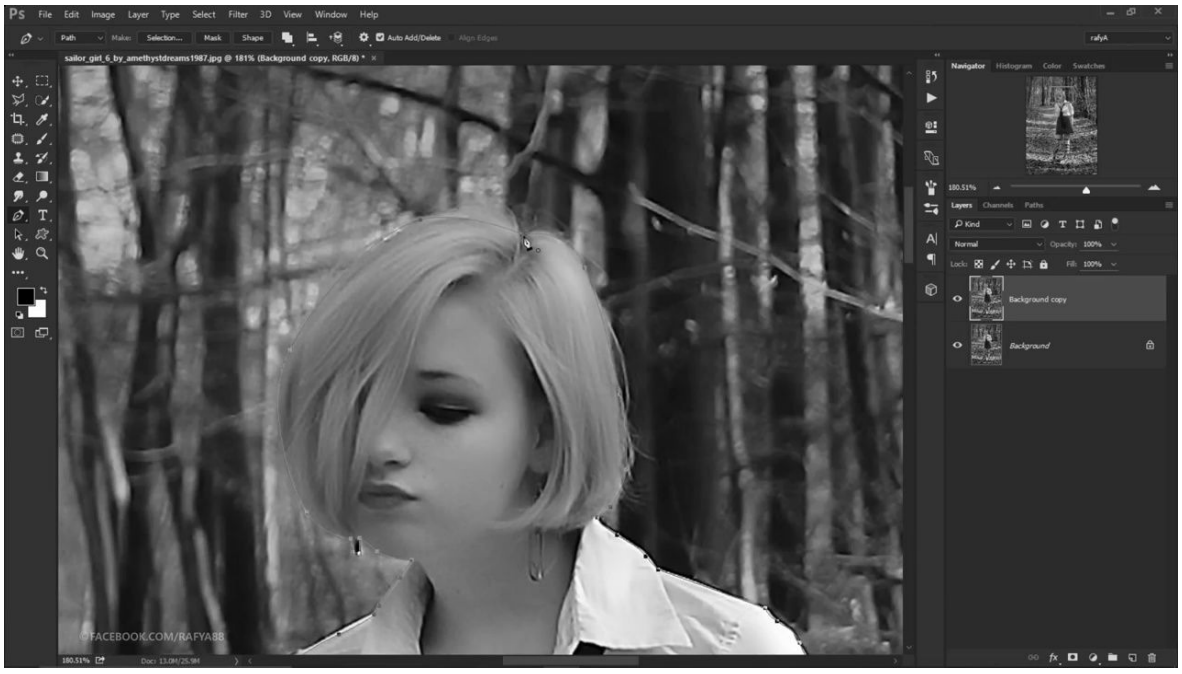
THANK YOU FOR WATCHING

LIKE AND SHARE VIDEO IF YOU LIKE

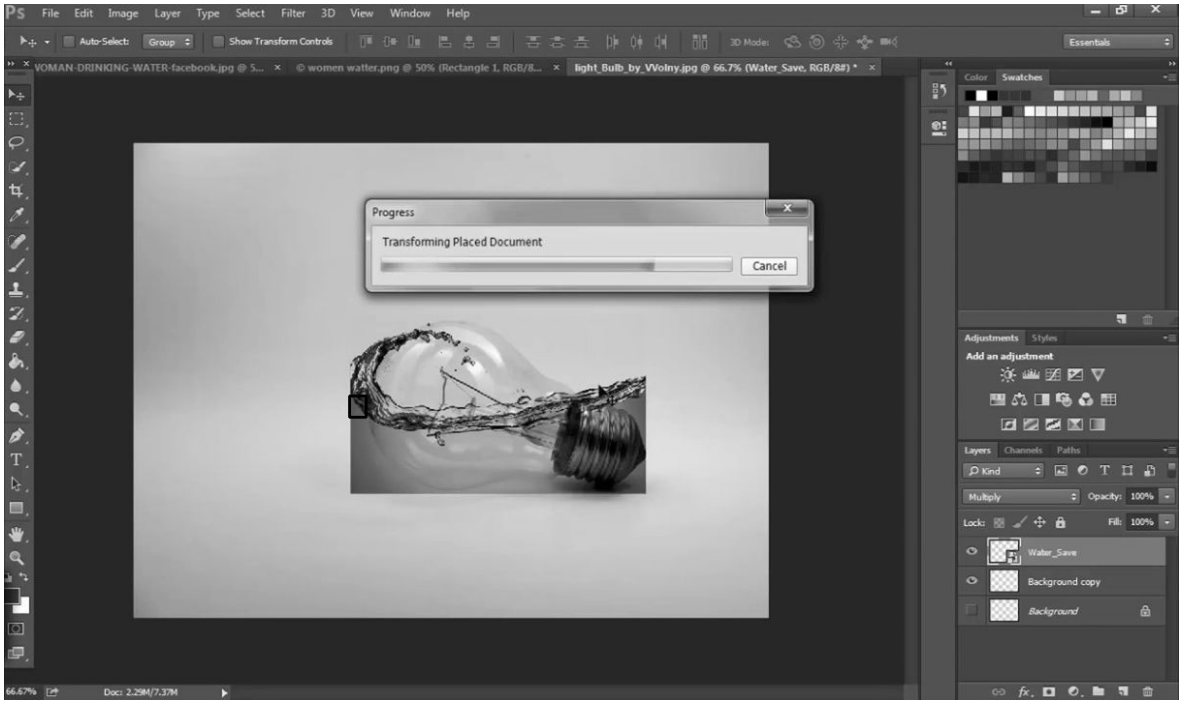

File Edit Image Layer Type Select Filter 3D View Window Help
bottom-of-the-sea-16913.jpg @ 51.4% (bubbles, RGB/8#) *
Color Swatches
Adjustments Styles
Add an adjustment
Layers Channels Paths
bubbles
org
Background copy
ENG 03:53 PM


File Edit Image Layer Type Select Filter 3D View Window Help
sailor_girl_6_by_amethystdreams1987.jpg @ 34.7% (Curves 1, Layer Mask/8) *
Properties
Curves
Navigator Histogram Color Swatches
Layers Channels Paths
© FACEBOOK.COM/RAFYA88

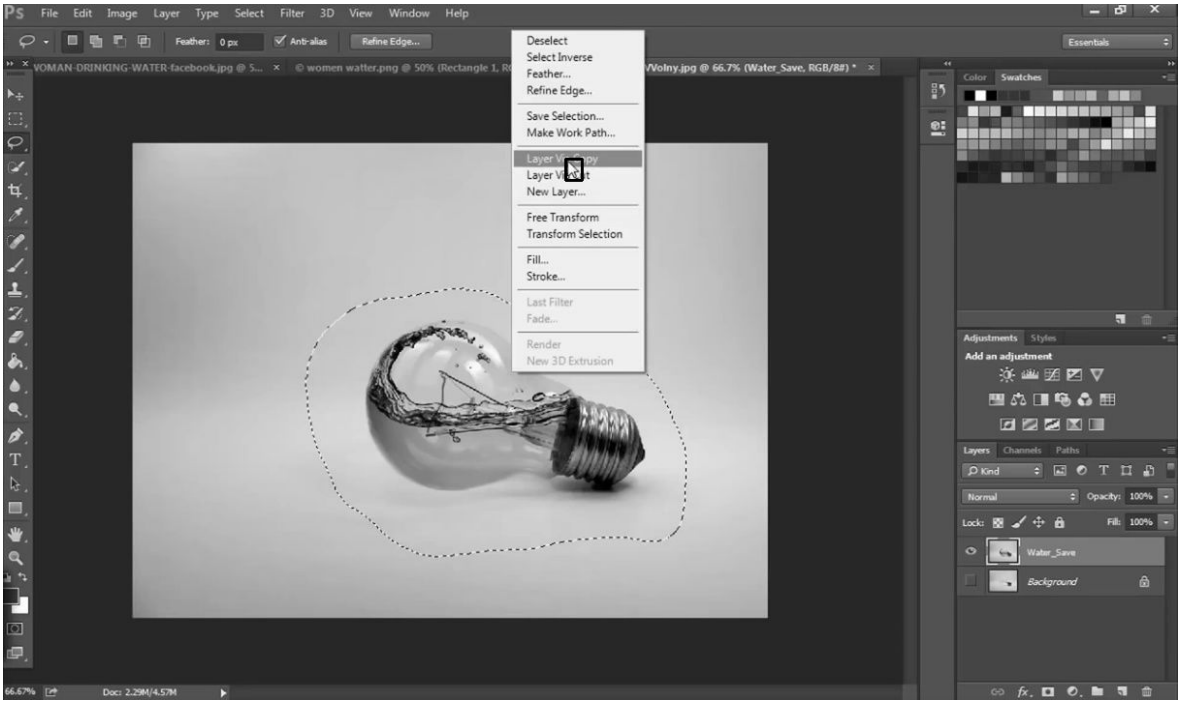

File Edit Image Layer Type Select Filter 3D View Window Help
light_Bulb_by_VVolny.jpg @ 66.7% (Layer 1 copy, RGB/8#) *
Color Swatches
Adjustments Styles
Add an adjustment
Layers Channels Paths
Normal
Opacity: 100%
Fill: 100%
Layer 1 copy
Layer 1
Water_Save
Background


File Edit Image Layer Type Select Filter 3D View Window Help
sailor_girl_6_by_amethystdreams1987.jpg @ 181% (Background copy, RGB/8) *
Navigator Histogram Color Swatches
Layers Channels Paths
Background copy
Background
©FACEBOOK.COM/RAFYA88

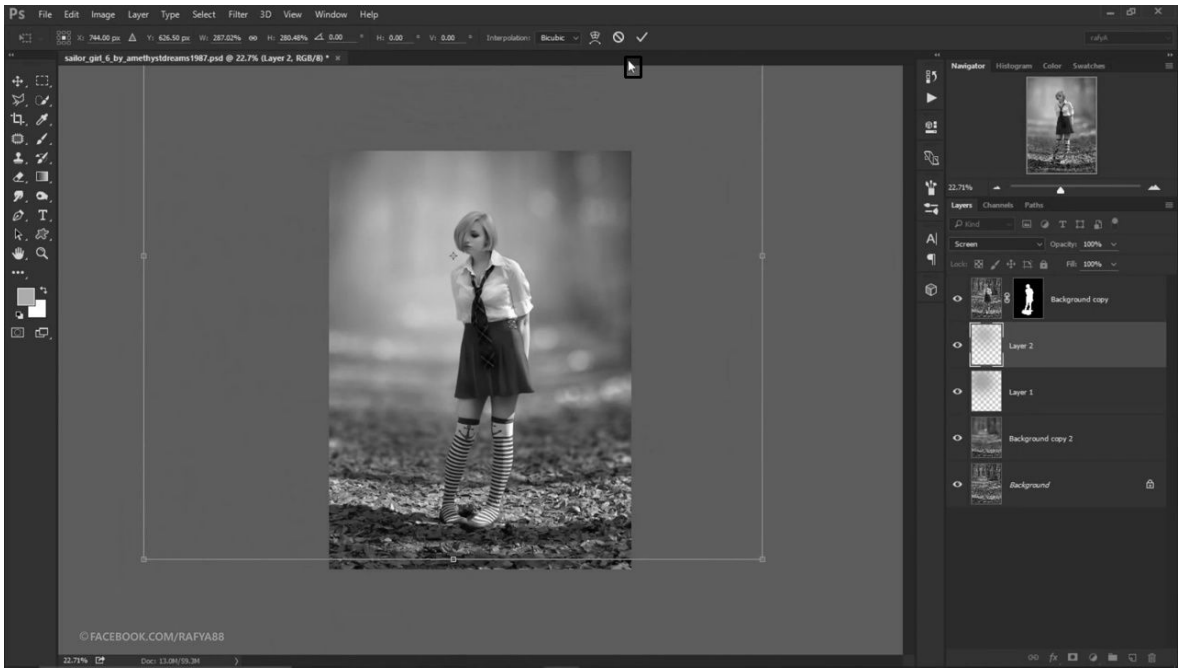

PHOTOSHOP

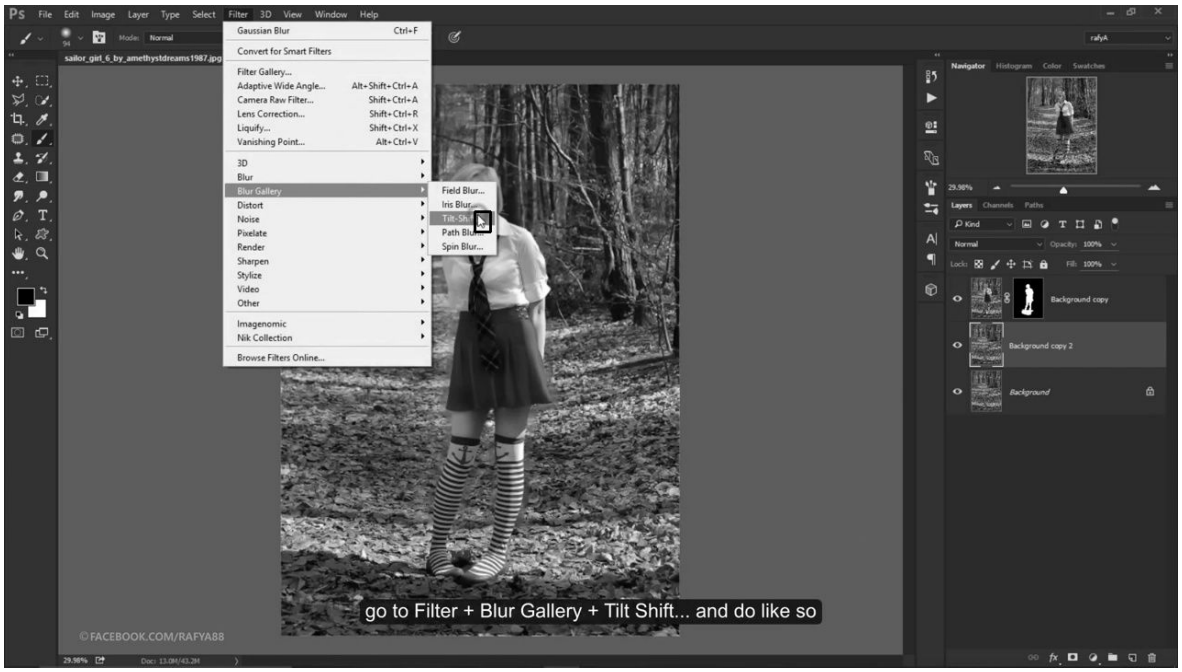

File Edit Image Layer Type Select Filter 3D View Window Help
Progress
Transforming Placed Document
Cancel
Color Swatches
Adjustments Styles
Add an adjustment
Layers Channels Paths
Multiply
Opacity: 100%
Fill: 100%
Water_Save
Background copy
Background


File Edit Image Layer Type Select Filter 3D View Window Help
Mode: Normal
Opacity: 100%
Flow: 100%
Essentials
light_Bulb_by_VVolny.jpg @ 200% (Layer 1, Layer Mask/8) *
Color Swatches
Adjustments Styles
Add an adjustment
Layers Channels Paths
Kind
Normal
Opacity: 100%
Lock:
Fill: 100%
Layer 1 copy
Layer 1
Water_Save
Background
200%
Doc: 2.29M/6.72M

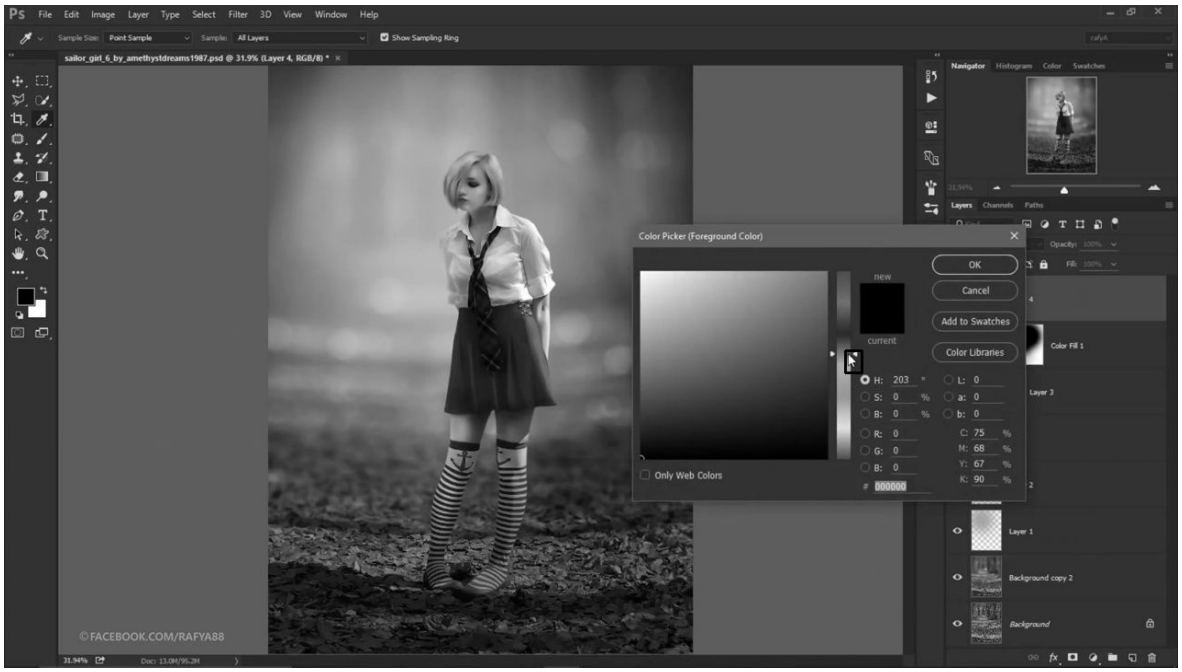

File Edit Image Layer Type Select Filter 3D View Window Help
Feather: 0 px
Anti-alias
Refine Edge...
Essentials
Deselect
Select Inverse
Feather...
Refine Edge...
Save Selection...
Make Work Path...
Layer via Copy
Layer via Cut
New Layer...
Free Transform
Transform Selection
Fill...
Stroke...
Last Filter
Fade...
Render
New 3D Extrusion
Color Swatches
Adjustments Styles
Add an adjustment
Layers Channels Paths
Kind
Normal
Opacity: 100%
Lock:
Fill: 100%
Water_Save
Background
66.67%
Doc: 2.29M/4.57M


File Edit Image Layer Type Select Filter 3D View Window Help
light_Bulb_by_VVolny.jpg @ 66.7% (Layer 1, RGB/8#) *
Adjustments
Add an adjustment
Layers
Layer 1 copy
Layer 1
Water_Save
Background

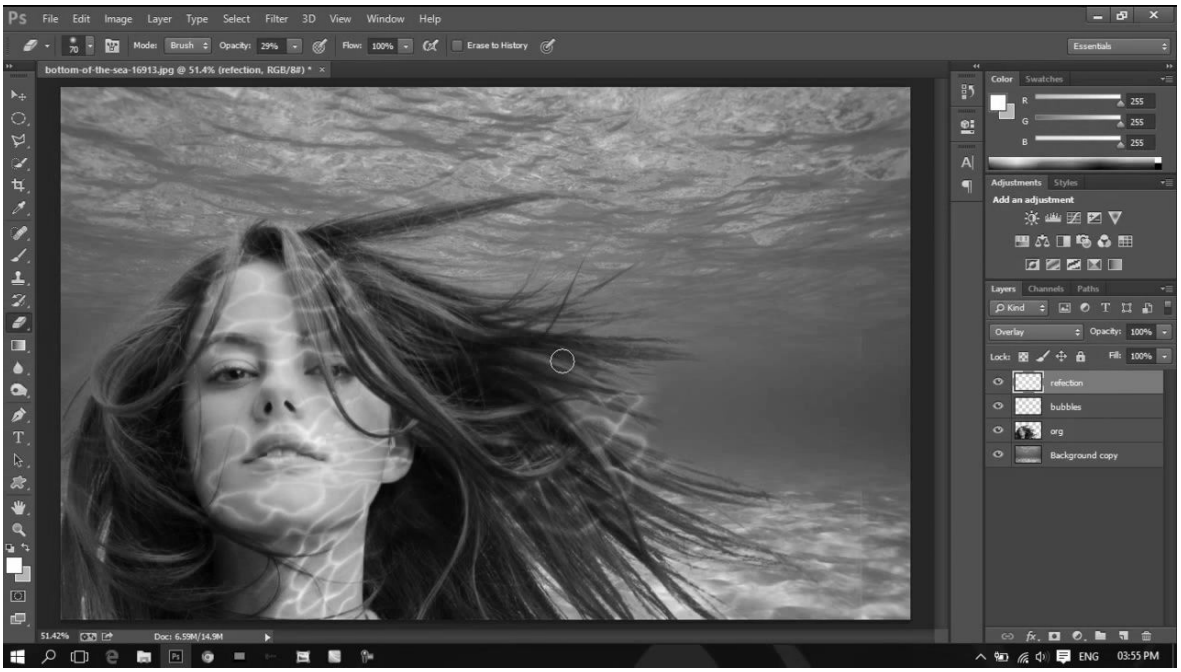

File Edit Image Layer Type Select Filter 3D View Window Help
sailor_girl_6_by_amethystdreams1987.psd @ 22.7% (Layer 2, RGB/8) *
Navigator
Background copy
Layer 2
Layer 1
Background copy 2
Background
©FACEBOOK.COM/RAFYA88

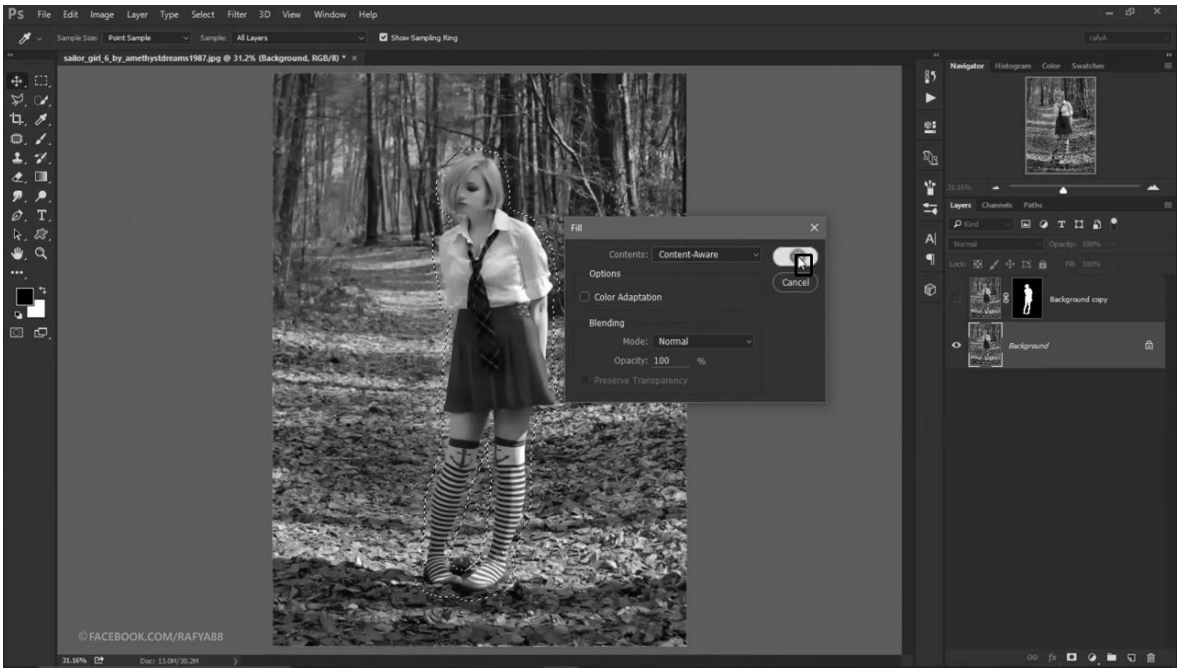

File Edit Image Layer Type Select Filter 3D View Window Help
Gaussian Blur Ctrl+F
Convert for Smart Filters
Filter Gallery...
Adaptive Wide Angle... Alt+Shift+Ctrl+A
Camera Raw Filter... Shift+Ctrl+A
Lens Correction... Shift+Ctrl+R
Liquify... Shift+Ctrl+X
Vanishing Point... Alt+Ctrl+V
3D
Blur
Blur Gallery
Distort
Noise
Pixelate
Render
Sharpen
Stylize
Video
Other
Imagenomic
Nik Collection
Browse Filters Online...
Field Blur...
Iris Blur...
Tilt-Shift...
Path Blur...
Spin Blur...
Navigator Histogram Color Swatches
Layers Channels Paths
Background copy
Background copy 2
Background
go to Filter + Blur Gallery + Tilt Shift... and do like so
© FACEBOOK.COM/RAFYA88


File Edit Image Layer Type Select Filter 3D View Window Help
bottom-of-the-sea-16913.jpg @ 51.4% (bubbles, RGB/8#) *
Color Swatches
Adjustments Styles
Add an adjustment
Layers Channels Paths
Overlay
Opacity: 100%
Fill: 100%
bubbles
org
Background copy
51.42%
Doc: 6.59M/13.7M
ENG
03:54 PM


File Edit Image Layer Type Select Filter 3D View Window Help
sailor_girl_6_by_amethystdreams1987.psd @ 31.9% (Layer 4, RGB/8) *
Color Picker (Foreground Color)
OK
Cancel
Add to Swatches
Color Libraries
new
current
Only Web Colors
© FACEBOOK.COM/RAFYA88
Layer 1
Background copy 2
Background


File Edit Image Layer Type Select Filter 3D View Window Help
bottom-of-the-sea-16913.jpg @ 51.4% (org, RGB/8#) *
Color Swatches
Adjustments Styles
Add an adjustment
Layers Channels Paths
org
Background copy
ENG 03:53 PM





code

```
import cv2
import numpy as np
import glob
import imutils
images = [cv2.imread(file,0) for file in
glob.glob("/home/hariprasath/Desktop/ub/computer_vision_image_processing/task3/*.jpg")]
template =
cv2.imread("/home/hariprasath/Desktop/ub/computer_vision_image_processing/task3/template
/template.png",0)

blur = [cv2.GaussianBlur(image,(3,3),1) for image in images]
```

```
laplacian = [cv2.Laplacian(image,cv2.CV_32F) for image in blur]
w, h = template.shape[::-1]

for i,image in enumerate(laplacian):
    current_max = 0
    gmin_loc=(0,0)
    gv =0
    gh=0
    for scale in np.linspace(0.6, 1.0, 20)[::-1]:
        resized = imutils.resize(template, width = int(template.shape[1] * scale))

        template_gaussian = cv2.GaussianBlur(resized,(3,3),1)
        template_laplacian = cv2.Laplacian(template_gaussian,cv2.CV_32F)

        r = template_laplacian.shape[1] / float(resized.shape[1])
        result=cv2.matchTemplate(image,template_laplacian,cv2.TM_CCOEFF_NORMED)
        min_val, max_val, min_loc, max_loc = cv2.minMaxLoc(result)
        if(current_max < max_val and max_val > 0.45):
            current_max =max_val
            gmin_loc = max_loc
            gv,gh =resized.shape[::-1]
            #cv2.rectangle(images[i], gmin_loc, (gmin_loc[0] + gv, gmin_loc[1] + gh), (255,0,0), 2

    cv2.rectangle(images[i], gmin_loc, (gmin_loc[0] + gv, gmin_loc[1] + gh), (0,0,0), 2)
    cv2.imshow("image",images[i])
    cv2.waitKey(0)
```