# Image Processing TOOLBOX

Supervised by Abd El Rahman SHABAYEK

May 8, 2018

The objectives of this course work were :

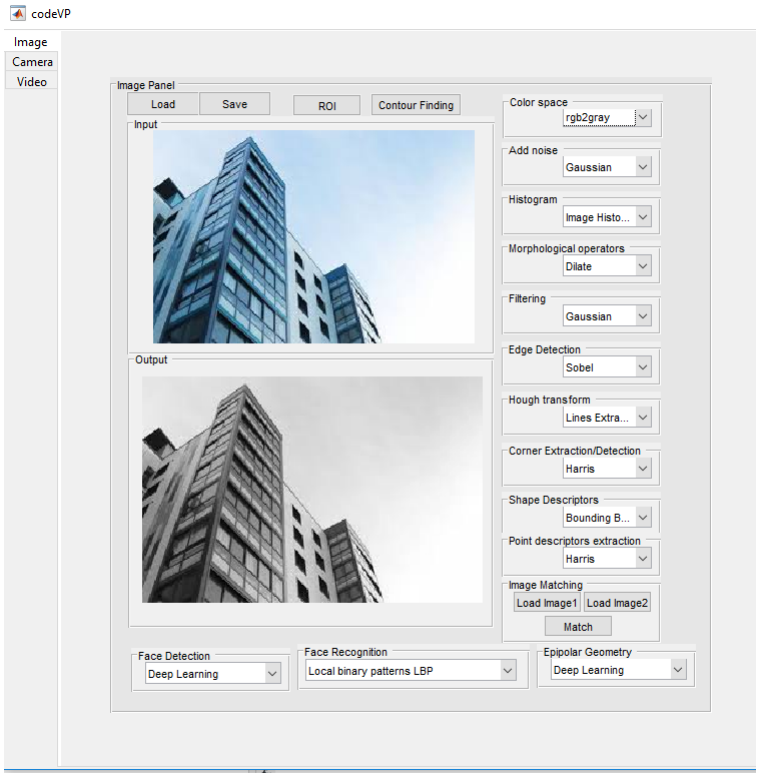
1. to learn MATLAB
2. to learn OpenCV as one of the most important computer vision open source libraries, OpenCV. 3.
3. to test and implement some important computer vision algorithms.

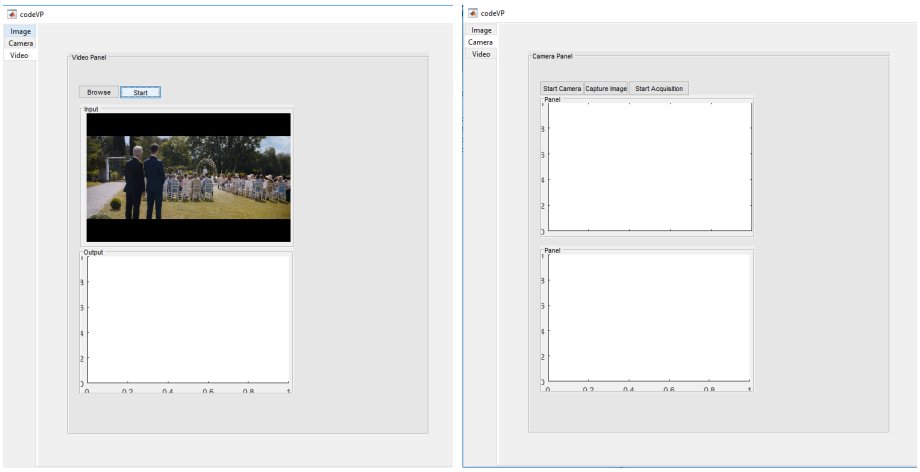
# I. MATLAB ToolBox

## 1. Architecture of the Graphical user interface

The matlab GUI is presented as a table with 3 tabs " where every input chosen has its own different tab. The GUI is shown as follow:

Figure 1: GUI image tab , video tab, camera tab (left to right)






# 2. Tasks

## 2.1. Basics

### 2.1.1.Image tab Load image

Click on the tab *Image* , then the button *Load* , then choose any image from your computer. The result will be shown in the first axes window

### 2.1.2. Load video

Click on the button *Browse* , to get the file path of the video you wan to watch . Then click on *Start* to start the video. The video can be observed in the top window

### 2.1.3. Save Image

The button *Save* is used to save any output image anywhere in the desktop.

### 2.1.4. Contour finding

By clicking on the button *Contour Finding*, we are able to observe drawn on the image the contours of the object.


## 2.2. Changing the colorspace

### 2.2.1. Original image

Inside the *colorspace* box , click on the arrow then on *Original* .It will show the original image in the down window as shown in figure 2.

### 2.2.2. Grayscale image

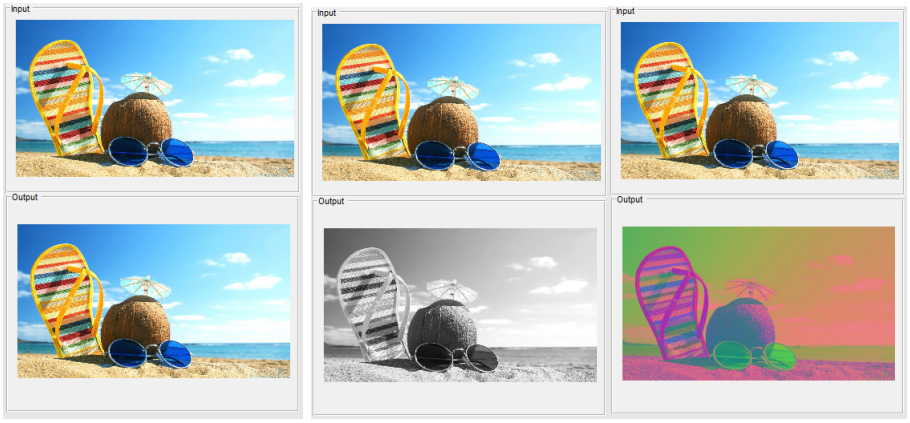
Inside the *colorspace* box , click on the arrow then on *rgb2gray* . This function will converts the truecolor image RGB into the grayscale intensity image. The result is shown in the down window as shown in figure 2.

### 2.2.3. YCBR image

Inside the *colorspace* box , click on the arrow then on *rgb2ycbrc* . This function will converts converts the RGB color space values in rgbmap to the YCbCr color space. The result is shown in the down window as shown in figure 2.

Note : Don't forget to choose the colorspace everytime you make a task

Figure 2: Original , grayscale and ycbrc image (left to right)



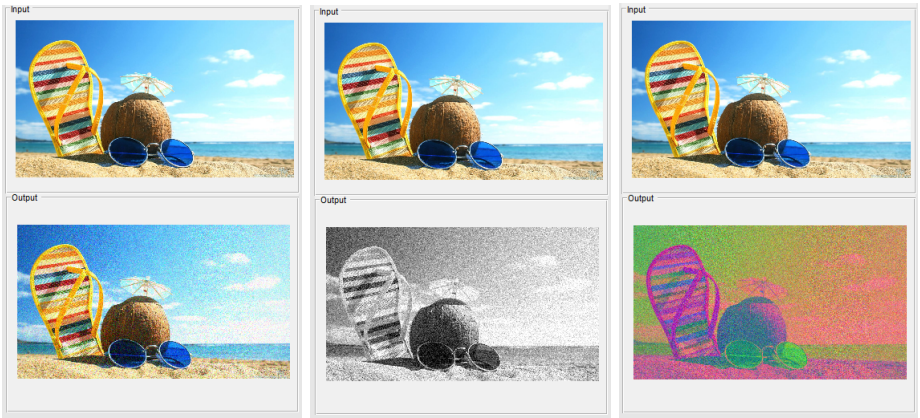

## 2.3. Add noise

Here we add different noise ( gaussian, salt and pepper , poisson and speckle) to a basic image and watch the difference. We can see as an example the following step on how to compute it.

### 2.3.1. Gaussian noise

Inside the *Add noise* box , click on the arrow then on *Gaussian* . The result is shown in the down window as shown in figure 3.

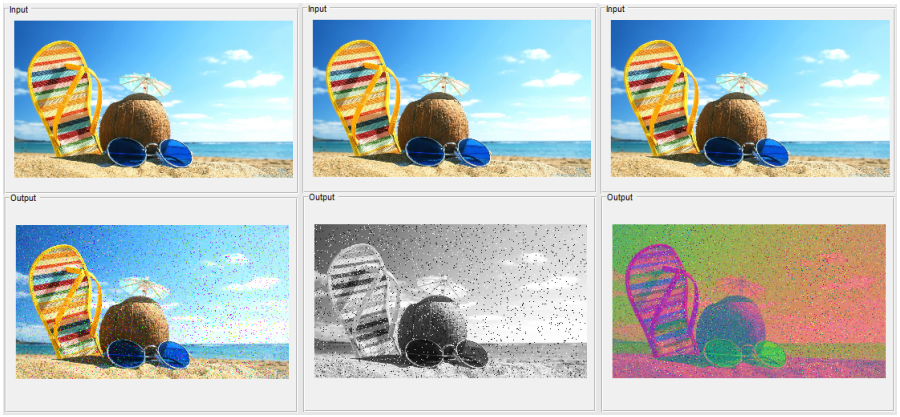
Figure 3: Gaussian noise on original , grayscale and ycbrc image (left to right)




### 2.3.2. Salt and Pepper noise

Inside the *Add noise* box , click on the arrow then on *Salt&Pepper* . The result is shown in the down window as shown in figure 4.

Figure 4: Salt&Pepper noise to original , grayscale and ycbrc (left to right)




## 2.4. Histogram

Before implementing the histogram, it is mandatory to choose a colorspace because the result will depend on it . Refer to the paragraph 2.2 to change the colorspace.

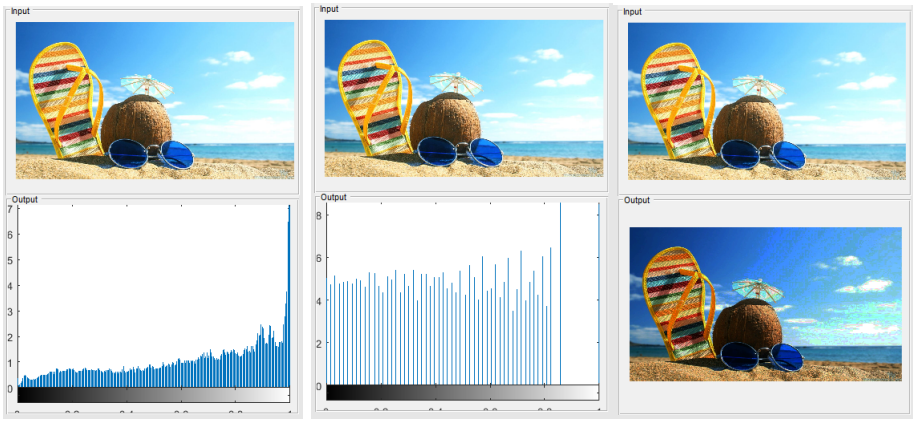
### 2.4.1. Display image Histogram

For instance, we will choose to keep the original image. Inside the *Histogram* box , click on the arrow then on *Image Histogram* . The function will plot the histogram of the image as shown in the down window as shown in figure 5.

### 2.4.2. Display image Histogram

Inside the *Histogram* box , click on the arrow then on *Histogram Equalization*. This function enhances the contrast of the original image using histogram equalization . The ploting and the final image result is shown in the down window as shown in figure 5.

Figure 5: Plotting the image histogram, the histogram equalization and show the image after histogram equalization (left to right)




## 2.5. Morphological operators

Binary images may contain numerous imperfections. In particular, the binary regions produced by simple thresholding are distorted by noise and texture. Morphological image processing pursues the goals of removing these imperfections by accounting for the form and structure of the image. So here we will applying different morphological operators to observe their different effect to the image.

To do that we will use the grayscale image (color space) then inside the *Morphological operators* box , click on the arrow and choose the operator you want to apply to the inage . The results are shown in the figure 6.

Figure 6: Dilatation, Erosion , Opening, Closing (left to right)

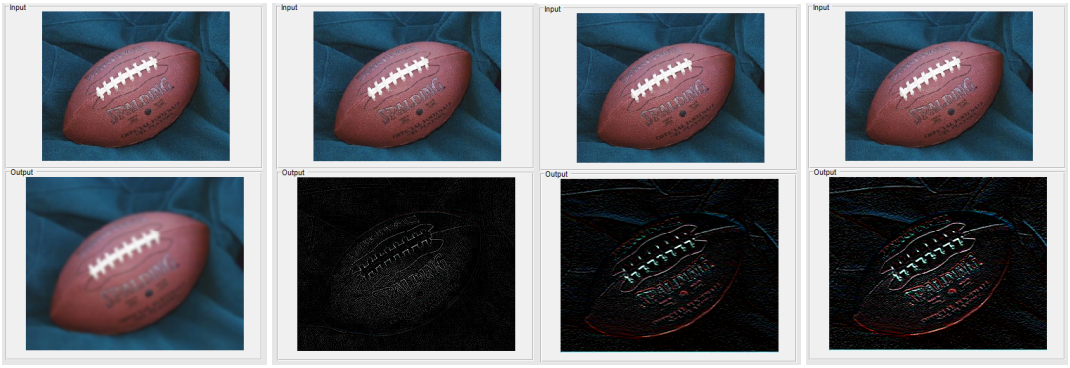

## 2.6. Filtering

In this part we want to blur , to smooth the image by applyind different filter on it.

To do that we will use the grayscale image (color space) then inside the *Filtering* box , click on the arrow and choose the filter you want to apply to the inage between *Gaussian*, *Laplacian*, *Prewitt* and *Sobel*.

In this example we will keep the original image color space. The results are shown in the figure 7.

Figure 7: Gaussian , Laplacian , Prewitt, Sobel Bluring (left to right)

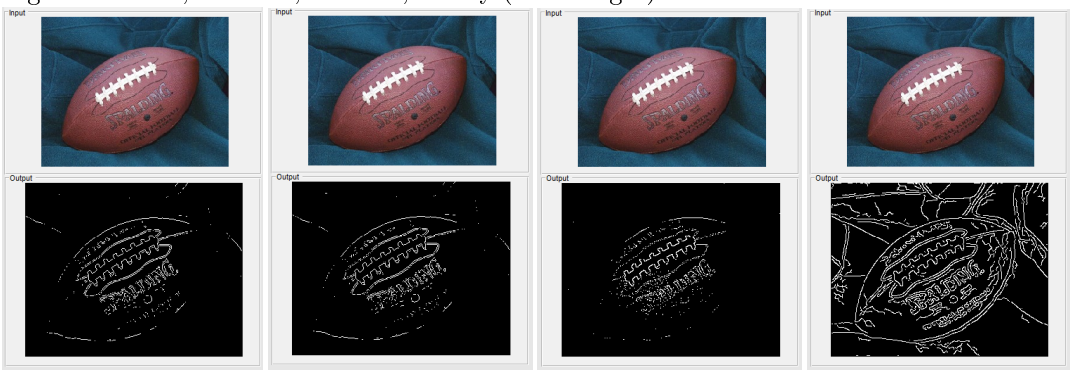


## 2.7. Edge detection

Edge detection is a fundamental tool in image processing, machine vision and computer vision, particularly in the areas of feature detection and feature extraction. In the ideal case, the result of applying an edge detector to an image may lead to a set of connected curves that indicate the boundaries of objects, the boundaries of surface.

To apply the edge detector operators on the image you must specify *rgb2gray* as colorspace then inside the *Edge detection* box , click on the arrow then choose the operator you want to apply to the inage between *Sobel*. *Prewitt*, *Roberts* and *Canny*. The results are shown in the figure 8.

Figure 8: Sobel, Prewitt , Roberts, Canny (left to right)



## 2.8. Hough Transform

The Hough transform is a feature extraction technique used in image analysis, computer vision, and digital image processing.The purpose of the technique is to find imperfect instances of objects within a certain class of shapes by a voting procedure. The classical Hough transform was concerned with the identification of lines in the image, but later the Hough transform has been extended to identifying positions of arbitrary shapes, most commonly circles or ellipses. Here we specify $rgb2gray$ as colorspace.

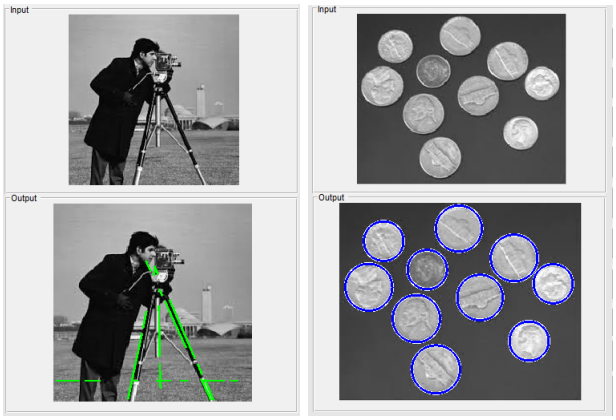
### 2.8.1. Lines extraction using Hough Transform

To extract lines using Hough transform, click on the arrow then choose $Lines\,extraction$ inside the $HoughTransform$ box. See the result in figure 9

### 2.8.2. Circles extraction using Hough Transform

To extract lines using Hough transform, click on the arrow then choose $Circles\,extraction$ inside the $HoughTransform$ box. See the result in figure 9

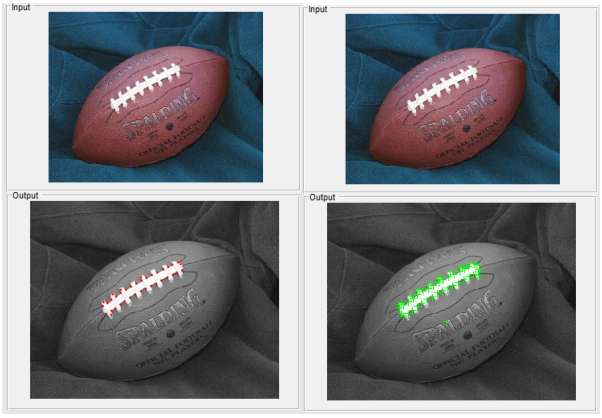
Figure 9: Lines extraction and Circles extraction using Hough Transform




## 2.9. Corner extraction

You must specify $rgb2gray$ as colorspace then inside the $Corner\,extraction$ box, then click on the arrow then choose $Harris$ or $FAST$ . box. See the result in figure 10

Figure 10: Harris and FAST corner extraction

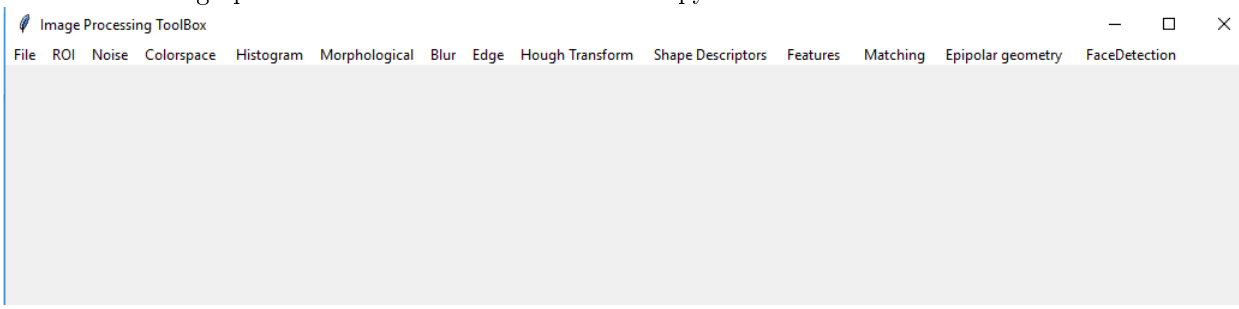


# II. Python using ToolBox

## 1. Basics

### 1.1. GUI user interface - Menu task bar

Shown below is the graphical user interface of the toolbox in python.




### 1.2. Input

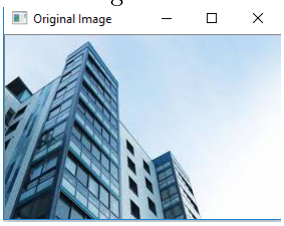
We can choose one on this following input :

1. Load image
2. Load video
3. Load Live camera

## 2. Tasks

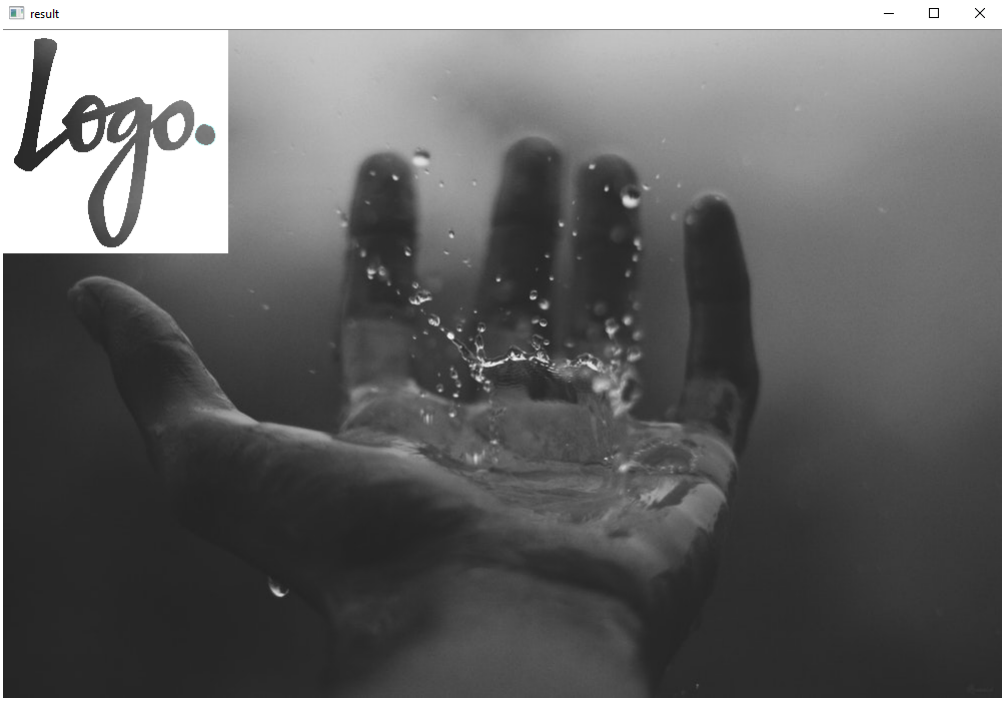
### 2.1. File

Load an image



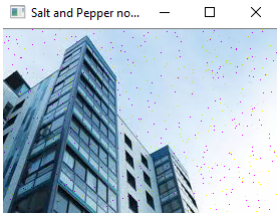

### 2.2. ROI

Here we choose a logo and add it to our image.



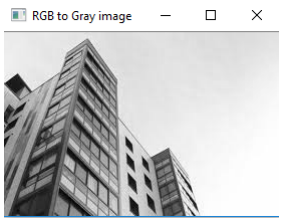

## 2.3. Noise

Add salt and Pepper noise

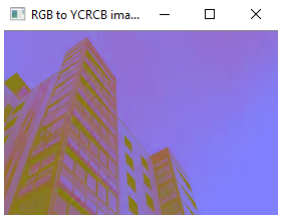



## 2.4. Color Space

- Change the colorspace into Gray



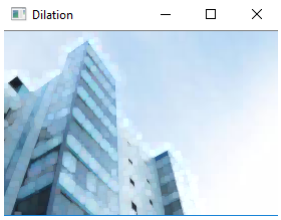

- Change the Rgb image to YCBRC image



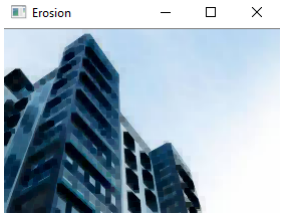

## 2.5. Histogram

## 2.6. Morphological operators

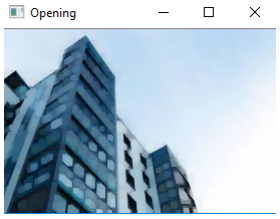
- Dilatation



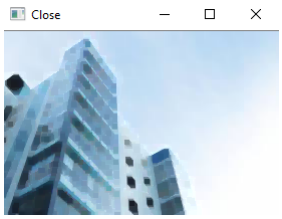

- Erosion




- Opening

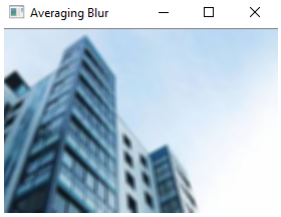



- Closing

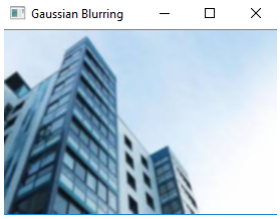


## 2.7. Blur

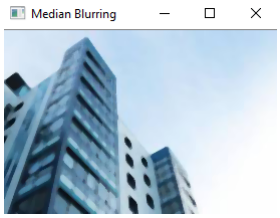
- Averaging




- Gaussian Blur

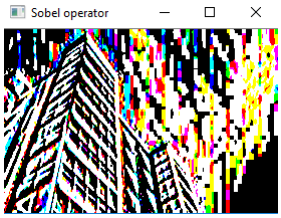



- Median Blur

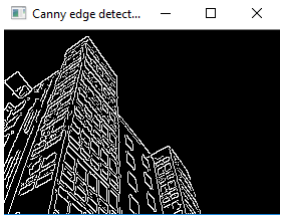



## 2.8. Edge detection

- Sobel




- Canny




## 2.9. Hough Transform

## 2.10. Shape Descriptors

- Finding contour