University of Burgundy

MsCV

Visual Perception Module

Report for

Computer Vision Toolbox

in

MATLAB

by

Gopikrishna Erabati

Supervisor: Dr. Abd El Rahman SHABAYEK

1. Introduction

Computer vision is the science that aims to give a similar, if not better, capability to a machine or computer. Computer vision is concerned with the automatic extraction, analysis and understanding of useful information from a single image or a sequence of images. Computer vision is an interdisciplinary field that deals with how computers can be made for gaining high-level understanding from digital images or videos. From the perspective of engineering, it seeks to automate tasks that the human visual system can do.

Computer vision tasks include methods for acquiring, processing, analyzing and understanding digital images, and extraction of high-dimensional data from the real world in order to produce numerical or symbolic information, *e.g.*, in the forms of decisions.

So, to do all this pre-processing using image processing tools and other tools is preferred. So, we aim at building a toolbox where we can do all sort of image processing and computer vision processes in one interface.

MATLAB is a powerful tool to do what we aim at. So, we explore MATLAB to build our Computer Vision (CV) toolbox. We also make use of the MATLAB's Graphical User Interface (GUI) facility to built a good user interface for our toolbox.

Note : For all the techniques, the process to run gui for each technique is explained clearly, you can find like "To do this:" (in colors) to find them easily.

2. Building ToolBox

Firstly, I divided the processing techniques into different logical clusters, so that I can place them in under one tab for easy access. But, when we make the GUI in Matlab using 'guide' (not programmatically), we cannot make use of tab widget directly. So, I faced a problem in this, and I came up with a solution to this by making use of 'visible' feature of widgets. I used radio buttons to select the tabs and placed the tabs as panels or button groups in the gui figure, where at opening function of gui we can make the panels visible 'off' and make the first panel visible 'on' and at every press of radio button to select tab, callback functions are written to make the respective tab visible and all others off, and also make use of 'getPosition' property to get the position of first tab and set the position of all other tabs according to 'getPosition' of first tab. So, by this we can make use of certain properties of GUI objects and we can make 'tabs' in GUI using 'guide'.

This building toolbox section explains how the toolbox is built and also it acts as a user manual for users to make better use of it.

2.1 The General Architecture of UI

In general, we have four types of classifications in the GUI:

a. A panel to select the required Tab.

b. A panel to show the selected tab.

c. The Images panel (input and output)

d. The control panel (to load, process, save)

The GUI can be as seen in Fig.1

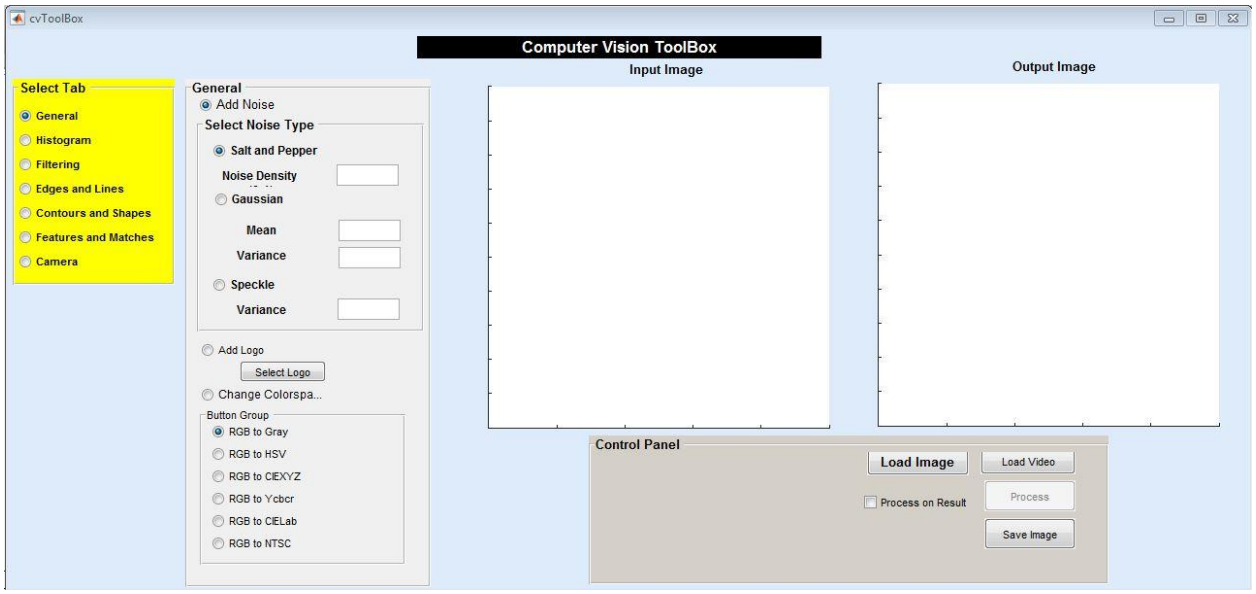

Fig. 1 The GUI

As seen in Fig.1 , the yellow color panel (Select Tab) is the panel to select the tab (a, as explained above), the panel beside the 'Select Tab' panel is the selected tab (b, as explained earlier), the two axes widgets are to show input and output data (c, as explained earlier) and below there is "control panel" to load the data, process the data and save the data (d, as explained earlier).

We can see a check box "process on result", this checkbox is for the user to decide if he/she wants to apply a processing technique on the result of previous technique or not.

2.2 The Processing Techniques on Images

Note: I had taken at most care to give the user the freedom to tune parameters for different techniques. And I also tried my best to well comment code for further reference of any user.

2.2.1 The General tab

2.2.1.1 Add noise to Images

In this tab, I implemented to add noise to the images, three types of noise 'salt and pepper', 'gaussian' and 'speckle' noise.

To do this:

1. Load image using 'load image' button.

2. Select the General tab.

3. Select 'Add noise'

4. Select noise type and enter respective parameters

5. Click process button and result will be in output image panel.

This is as shown in Fig.2

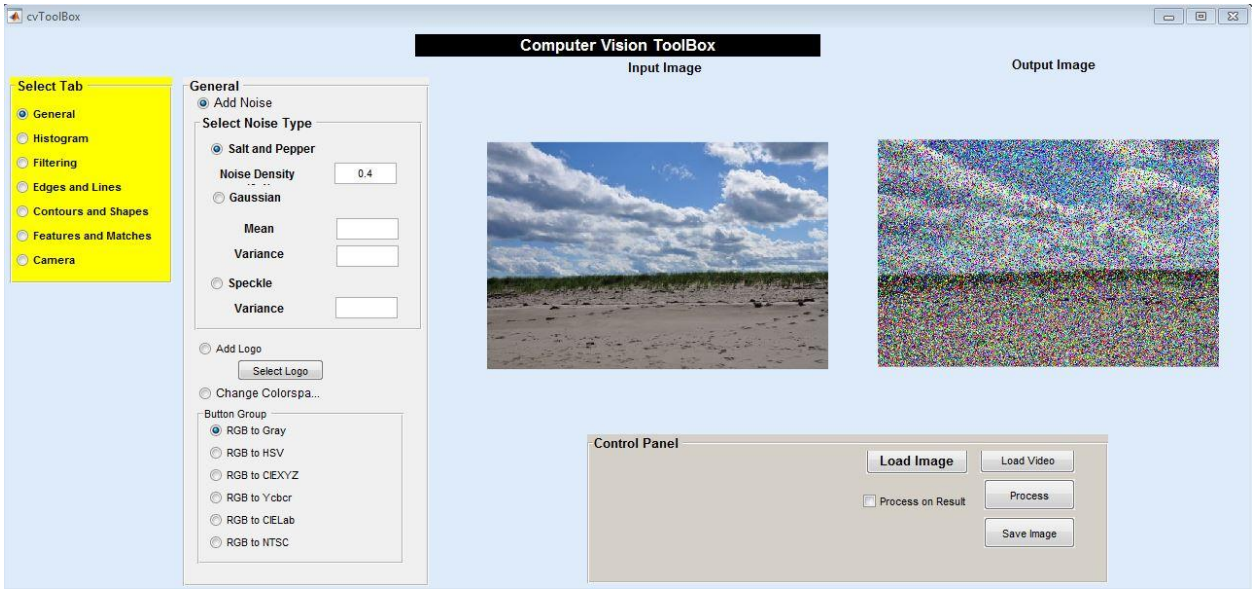

Fig. 2 Add salt and pepper noise

2.2.1.2 Add Logo

In this , module, we can add logo to our existing image.

To do this:

1. Load image using 'load image' button.

2. Select the General tab.

3. Select 'Add Logo

4. Select logo by 'selectlogo' button.

5. Click process button and result will be in output image panel.

The result will be as shown in Fig.3

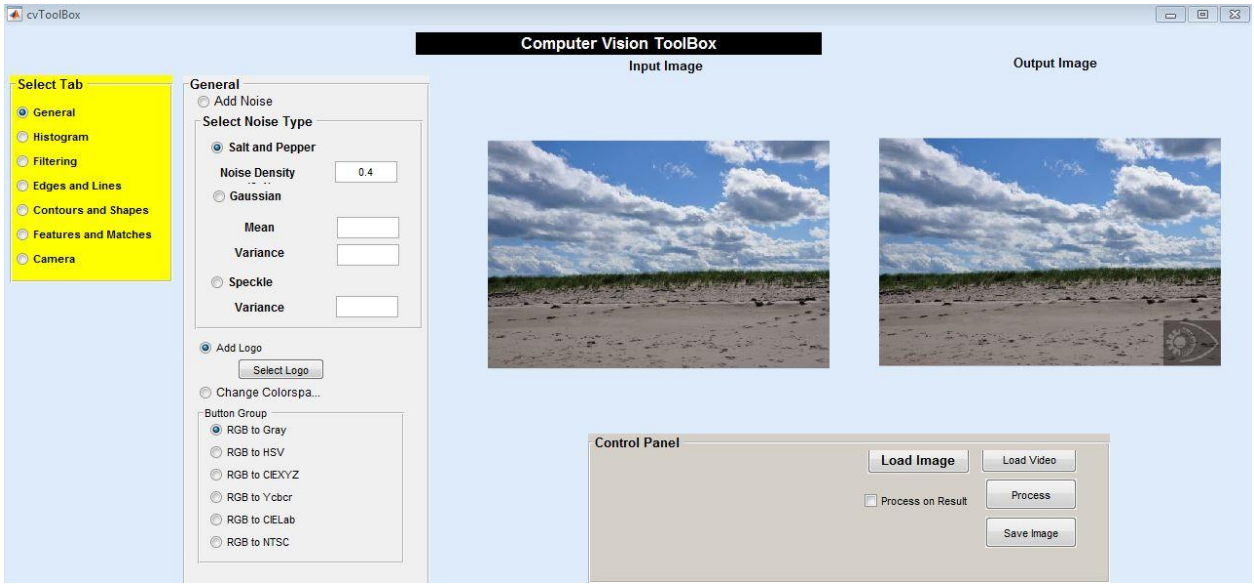

Fig. 3 Add logo

2.2.1.3 Change Colorspace

In this module, I implemented to change image into 6 different color spaces.

To do this:

1. Load image using 'load image' button.

2. Select the General tab.

3. Select Change Colorspace

4. Select color space to change to.

5. Click process button and result will be in output image panel.

The result will be as shown in Fig.4

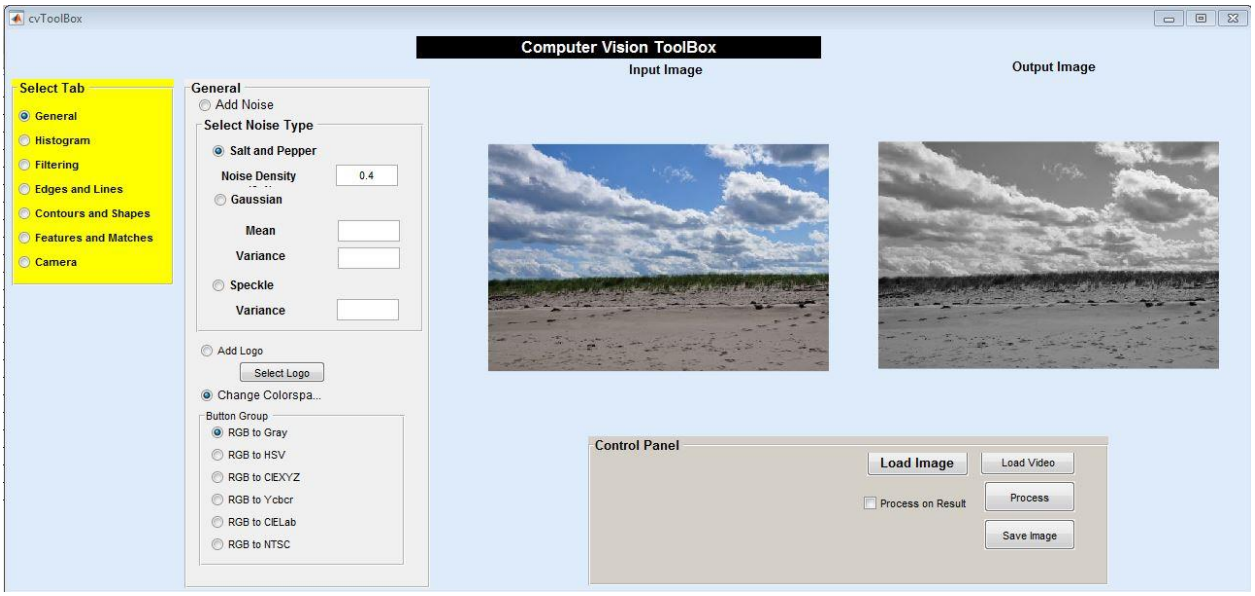

Fig. 4 RGB2GRAY

2.2.2.1 Histogram

In this module, we can plot histograms of both gray and color images. the color histograms will be R G B histograms.

To do this:

1. Load image using 'load image' button.

2. Select the Histogram & Morph. tab.

3. Select Histogram and plot histogram

4. Click process button and result will be in output image panel.

The result will be as shown in Fig.5

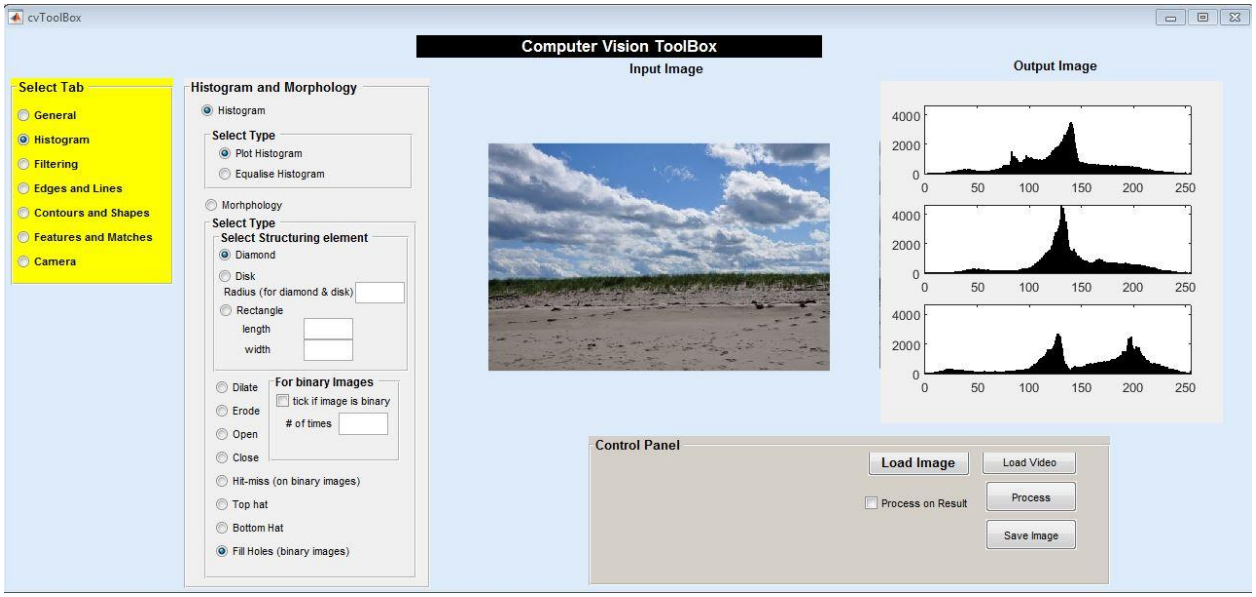

Fig. 5 histogram of RGB image

There is an option to equalise histogram

To do this:

1. Load image using 'load image' button.

2. Select the Histogram & Morph. tab.

3. Select Histogram and equalise histogram

4. Click process button and result will be in output image panel.

The result will be as shown in Fig.6

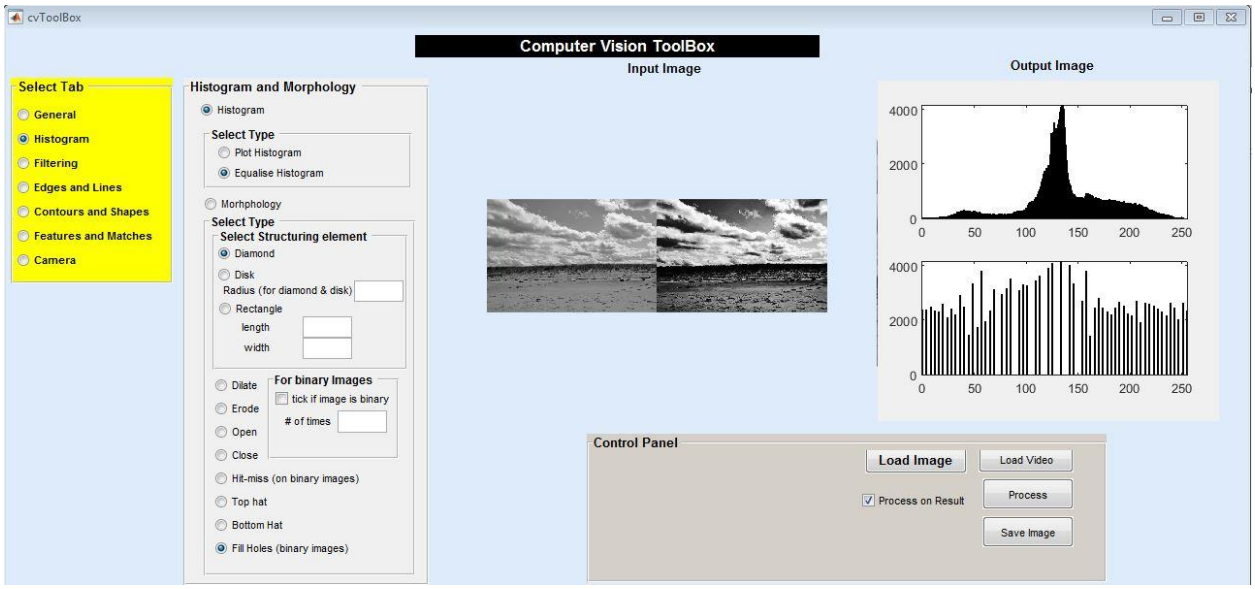

Fig. 6 Equalise histogram

2.2.2.2 Morphoogy

In this module, we can apply morphological operations (8 operations) with three types of structuring elements both on gray scale and binary images.

To do this:

1. Load image using 'load image' button.

2. Select the Histogram & Morph. tab.

3. Select Morphology

4. Select the morphological operation to perform and select the structuring element and respective parameters.

5. Click process button and result will be in output image panel.

The result will be as shown in Fig.7

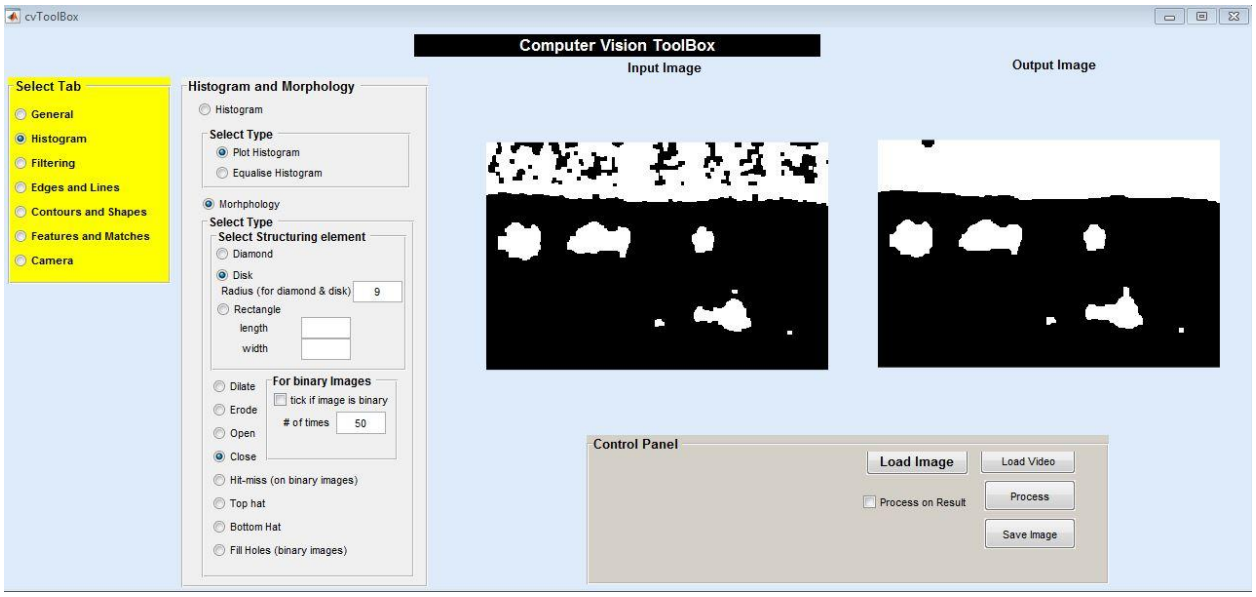

Fig. 7 Morphological Opening

2.2.3 Filtering (LPF & HPF)

2.2.3.1 LPF

I tried to implement LPF which blurs the image. the filters implemented are

Average, Gaussian, Bilateral, Median

To do this:

1. Load image using 'load image' button.

2. Select the Filtering tab.

3. Select LPF

4. Select type of LPF and enter parameters

5. Click process button and result will be in output image panel.

The result will be as shown in Fig.8

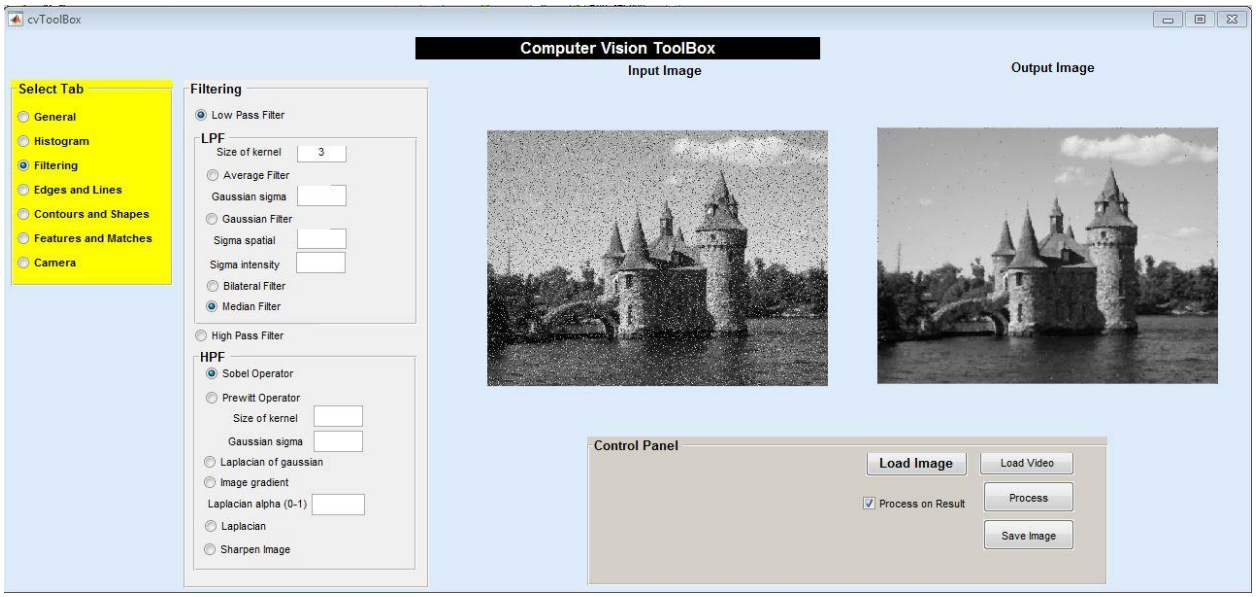

Fig. 8 Median filter on salt and pepper noise image

(Here, we can use checkbox (process on result) , to first apply salt and pepper noise on an image and then apply median filter on result)

2.2.3.2 HPF

Here, I tried to implemented all types of high pass filters possible like:

Sobel, Prewitt, Laplacian of gaussian, Laplacian, Image gradient and Sharpen.

To do this:

1. Load image using 'load image' button.

2. Select the Filtering tab.

3. Select HPF and

4. Select filter type and enter parameters

5. Click process button and result will be in output image panel.

The result will be as shown in Fig.9,10.

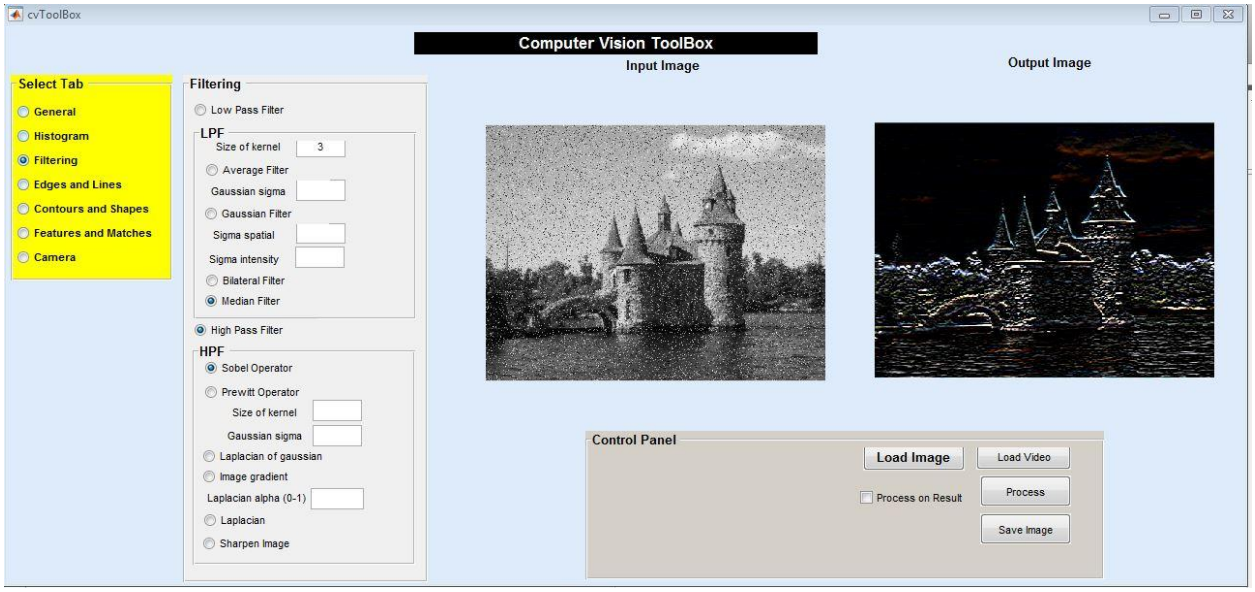

Fig. 9 Sobel operator

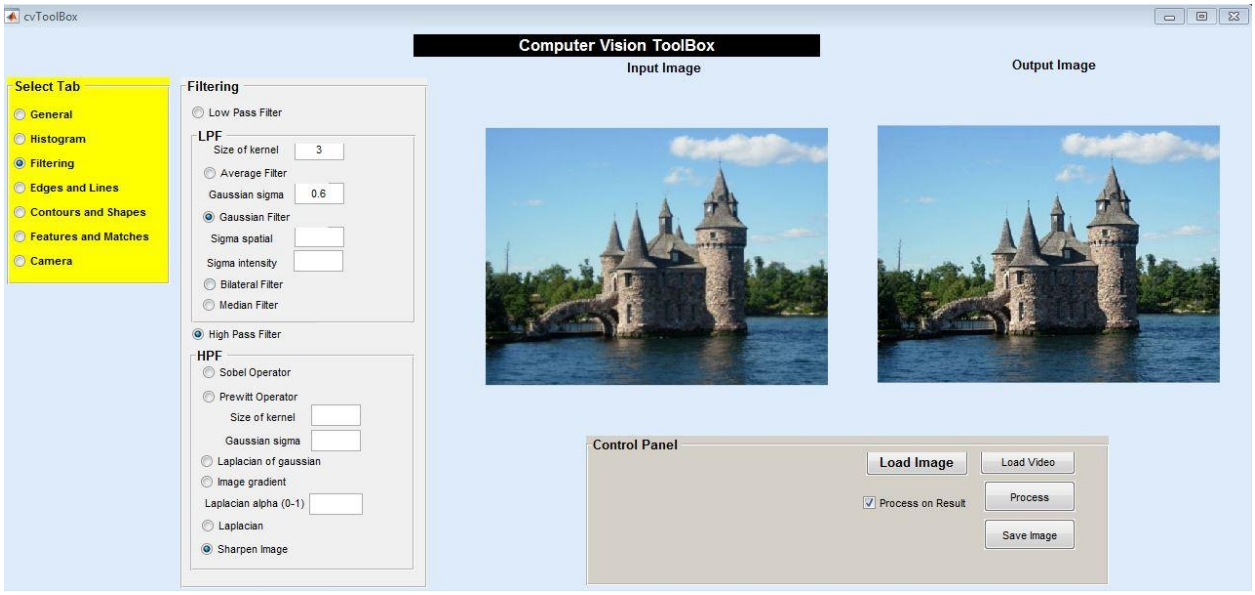

Fig. 10 Sharpen image

2.2.4 Edges, Lines and Circles

2.2.4.1 Edges

Here, I implemented six type of edge detectors like:

Canny, Lalpacian of Gaussian, zero crossings.

To do this:

1. Load image using 'load image' button.

2. Select the Edges and Lines tab.

3. Select Edges

4. Select edge type and enter parameters

5. Click process button and result will be in output image panel.

The result will be as shown in Fig.11

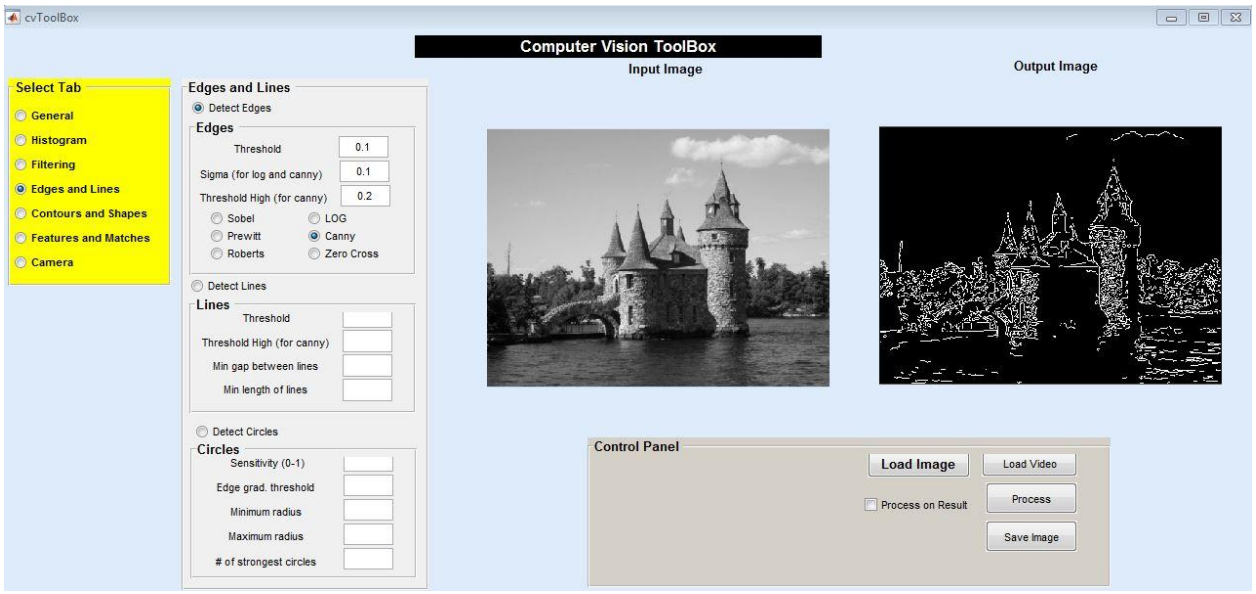

Fig. 11 Canny edge detector

2.2.4.2 Lines

We detect lines using Hough transform.

To do this:

1. Load image using 'load image' button.

2. Select the Edges and Lines tab.

3. Select Lines

4. Enter parameters

5. Click process button and result will be in output image panel.

The result will be as shown in Fig.12

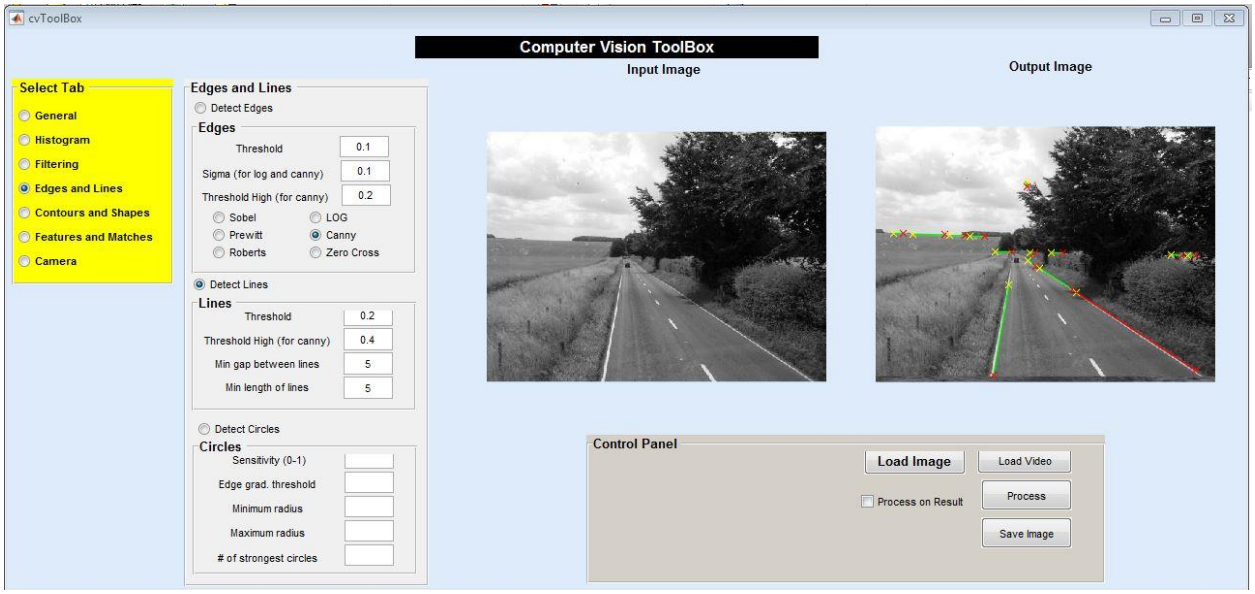

Fig. 12 Hough Lines

2.2.4.3 Circles

Circles are also detected using hough trasnform.

The result is as shown in Fig. 13.

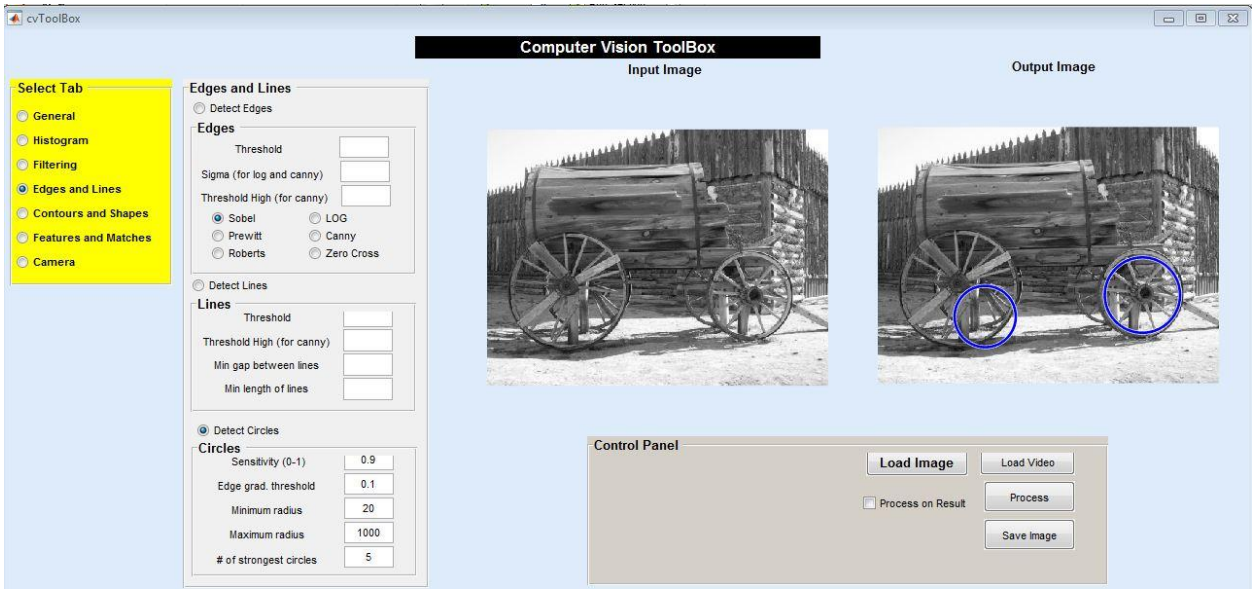

Fig. 13 Hough Circles

2.2.5 Contours and shapes

2.2.5.1 Contours

The input to this module should be a binary image.

The contours are drawn around the boundary of objects.

To do this:

1. Load image using 'load image' button.

2. Select the Contours and Shapes tab.

3. Select 'draw contours'

4. Click process button and result will be in output image panel.

The result will be as shown in Fig.14

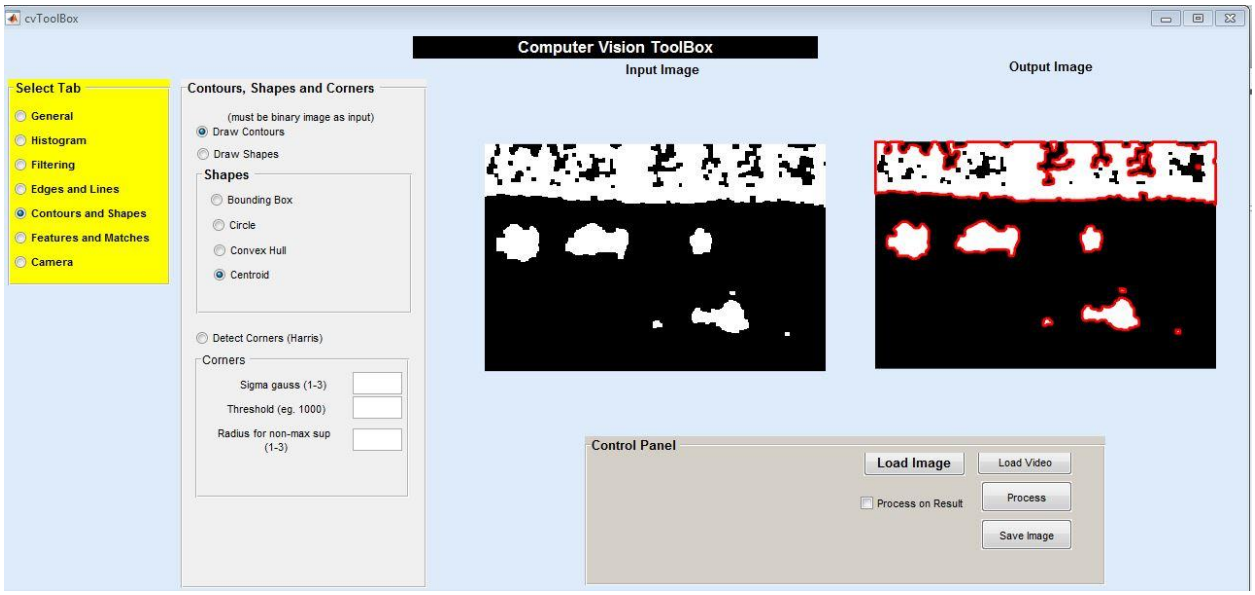

Fig. 14 Contours

2.2.5.2 Shapes

We can also draw basic shapes around the objects.

the shapes I implemented are : bounding box, minimum enclosing circle, convex hull, centoird of shape.

To do this:

1. Load image using 'load image' button.

2. Select the Contours and Shapes tab.

3. Select draw shapes

4. Select shape type.

5. Click process button and result will be in output image panel.

The result will be as shown in Fig.15, 16

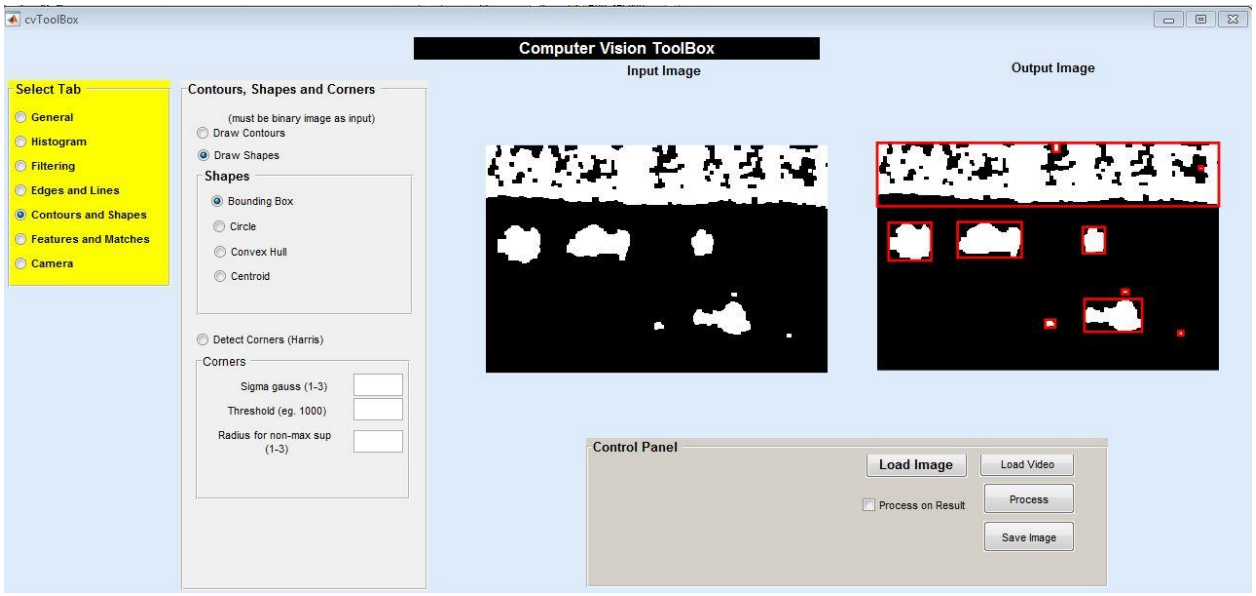

Fig.15 bounding box

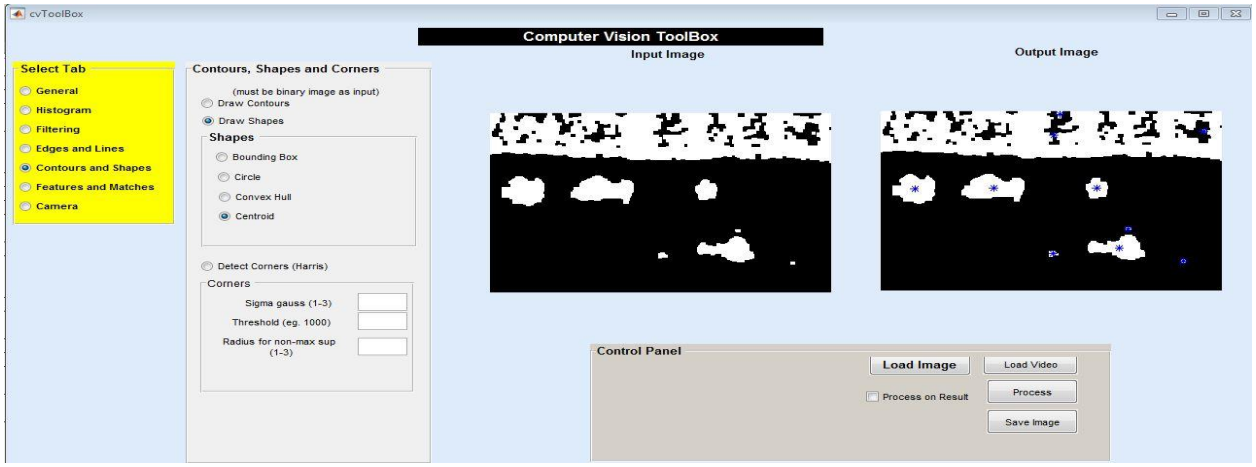

Fig. 16 Centroid of shapes

2.2.5.3 Detect Corners

We can detect corners in images using harris corner detector.

To do this:

1. Load image using 'load image' button.

2. Select the Contours and shapes tab.

3. Select Detect Corners (Harris)

4. Enter parameters

5. Click process button and result will be in output image panel.

The result will be as shown in Fig.17

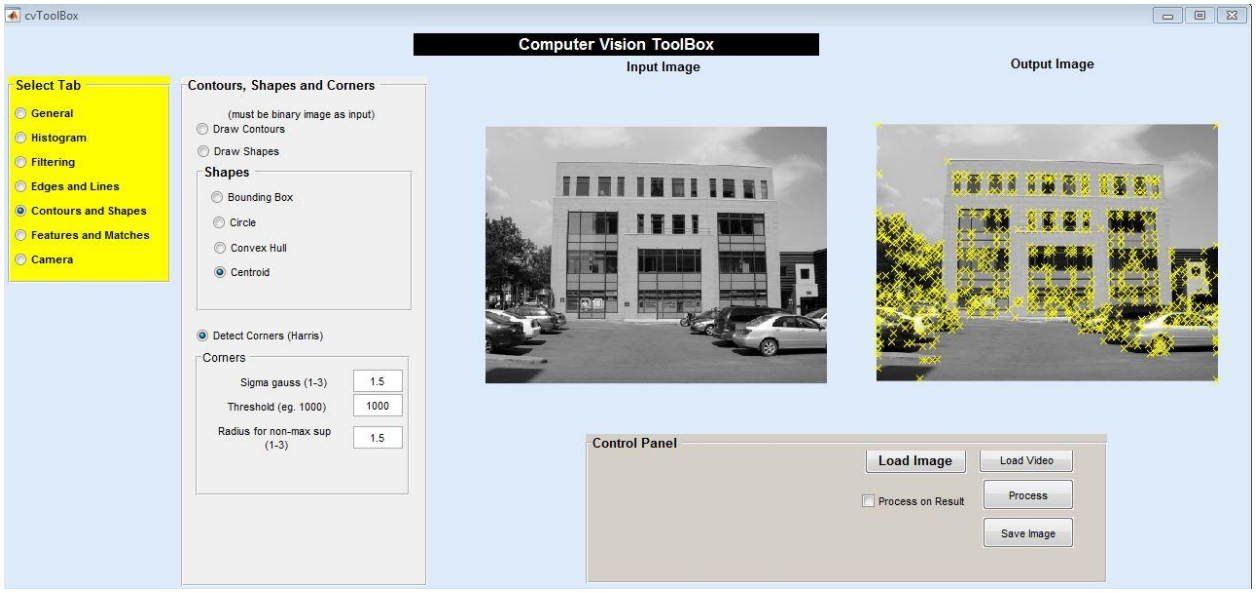

Fig. 17 Harris Corners

2.2.6 Features and Matching

We can detect features in an image using FAST, SURF, SIFT and HOG (Histogram of Gradient).

To do this:

1. Load image using 'load image' button.

2. Select the Features and Matching tab.

3. Select feature type (FAST, SURF, SIFT, HOG)

4. Enter parameters

5. Click process button and result will be in output image panel.

The result will be as shown in Fig.18, 19

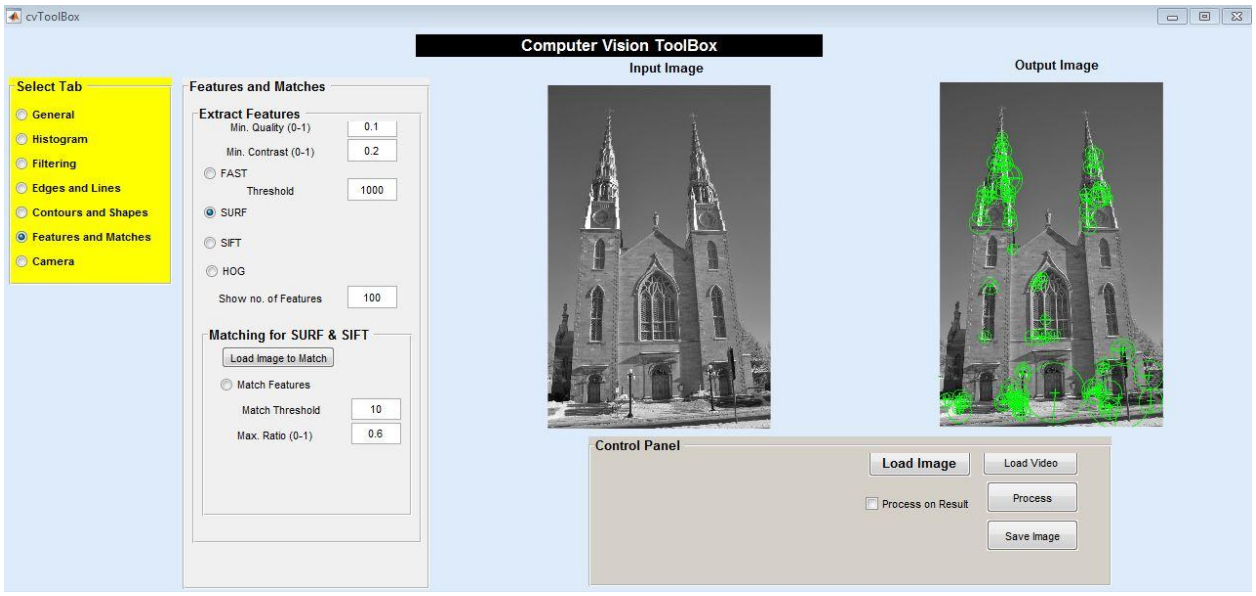

Fig. 18 SURF features

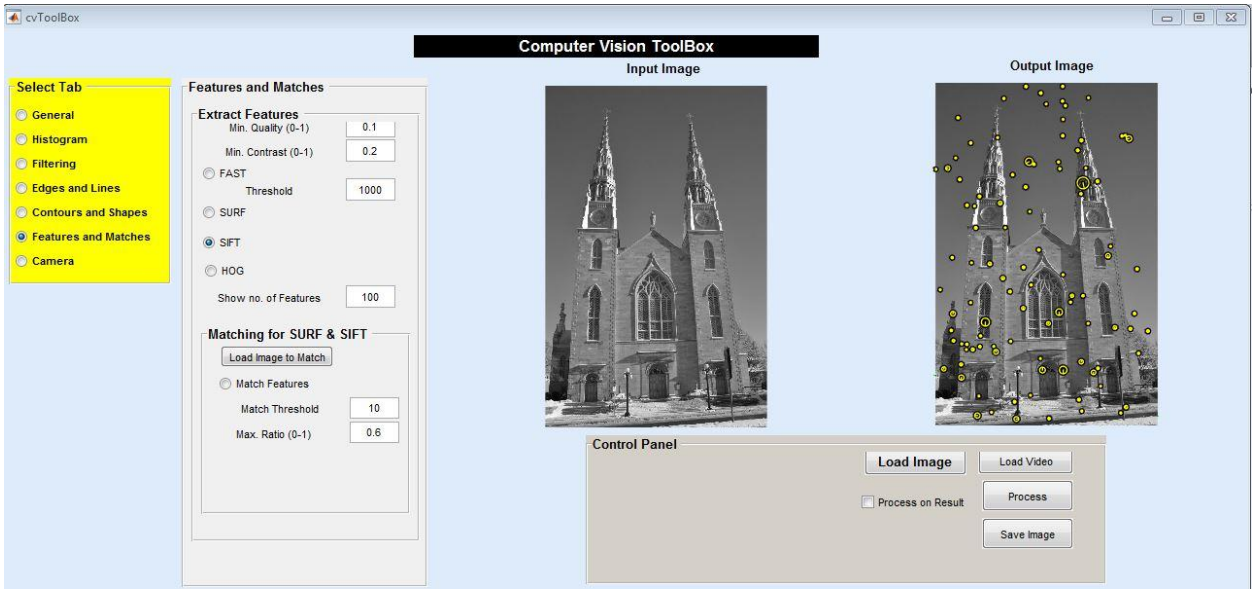

Fig. 19 SIFT Features

We can also perform matching using SURF and SIFT features between two images.

To do this:

1. Load image using 'load image' button.

2. Select the Features and Matching tab.

3. Select FAST or SURF feature.

4. Press load other image pushbutton to load second image to match.

5. Select Match Features radio button.

6. Enter matching parameters

7. Click process button to get matching features and matching lines as shown in Fig. 20.

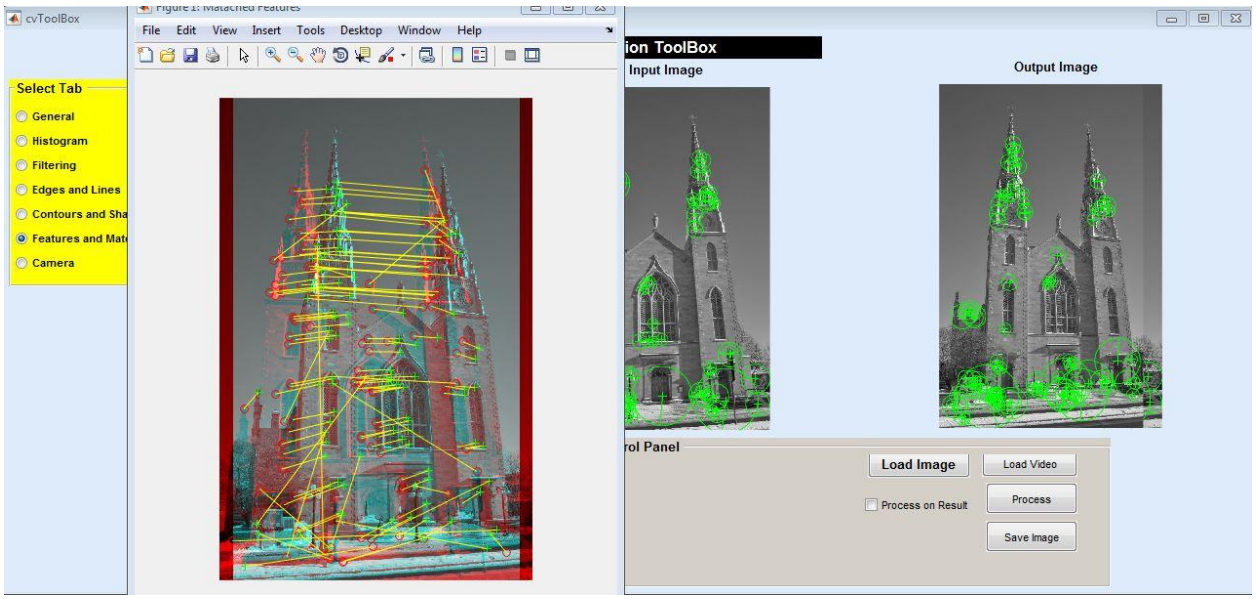

Fig. 20 SURF Matching

2.2.7 Camera Calibration and Hmography

2.2.7.1 Camera Calibration

We can calibrate camera given some calibration pattern images. By calibrating a camera we can also undistort any distortions in image.

To do this:

1. Load calibration images using 'load image to calibrate' button

2. Click process button and undistorted result will be in output image panel.

The result will be as shown in Fig. 21

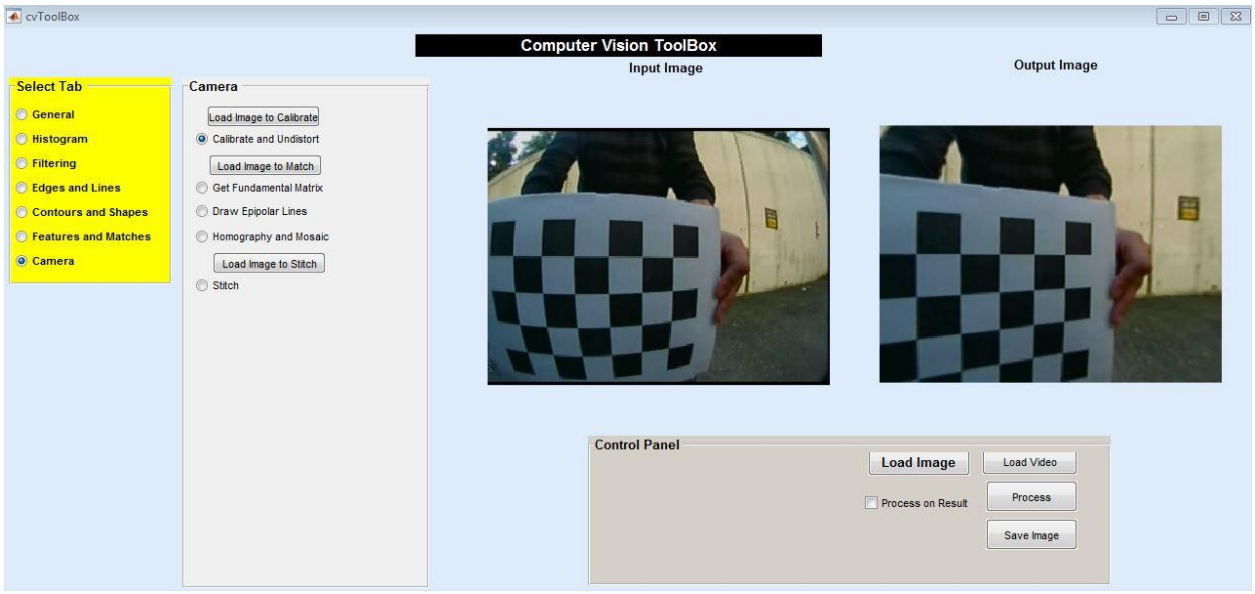

Fig. 21 Undistorted image after camera calibration

We can also visualize the extrinsic, like where the calibration pattern are according to camera.

This is as shown in Fig. 22.

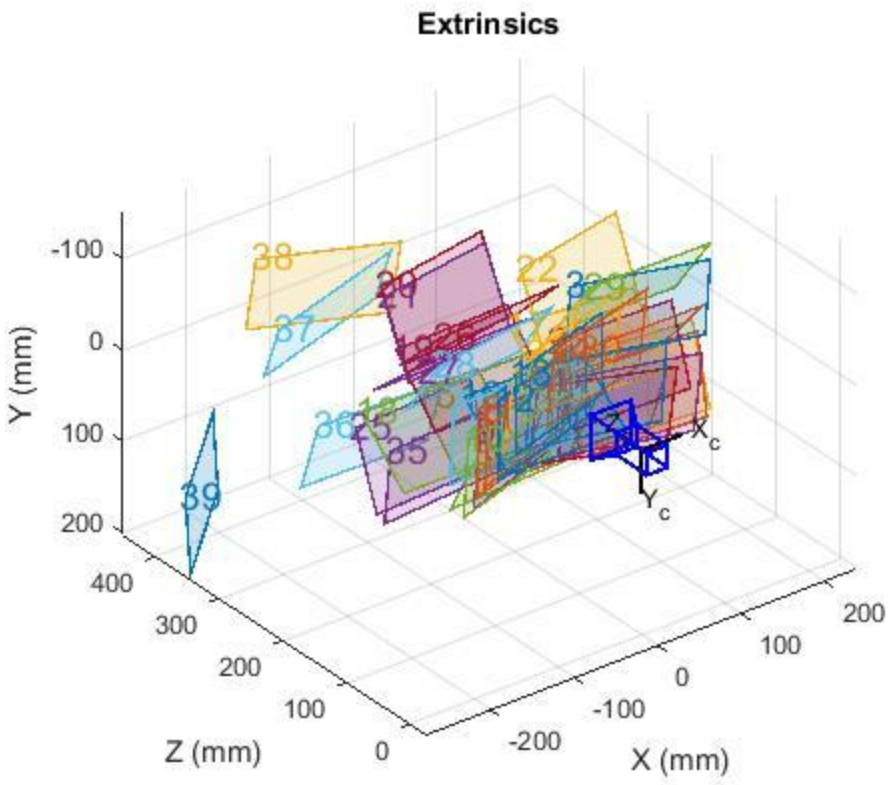

Fig. 22 Extrinsic (calibration pattern) with respect to camera

2.2.7.2 Fundamental matrix

We can get the fundamental matrix using 7-point, 8-point or RANSAC method from corresponding points. So we initially need corresponding points, which we can get from feature matching as described above in 2.2.6.

To do this:

1. Load image using 'load image' button.

2. Select the Camera tab.

3. Select Fundamental matrix radio button.

4. Click load image to match' to load the other image to find corresponding points and there by fundamental matrix.

5. Click process button and result will be shown in Control panel as a 3 x 3 matrix.

2.2.7.3 Draw Epipolar lines

When we have two images taken from camera (by moving camera) or by two cameras of a same scene, we can have matching points between both and by epipolar geometry the matching point of left image will be on epipolar line (of point in left image) in right image. So, at times , this epipolar constraint can help us to find good matching.

To do this:

1. Load image using 'load image' button.

2. Select the Camera tab.

3. Select 'draw epipolar lines' radio button.

4. Click load image to match' to load the other image to find corresponding points and there by draw epipolar lines.

5. Click process button to draw epipolar lines.

The result will be as shown in Fig. 23.

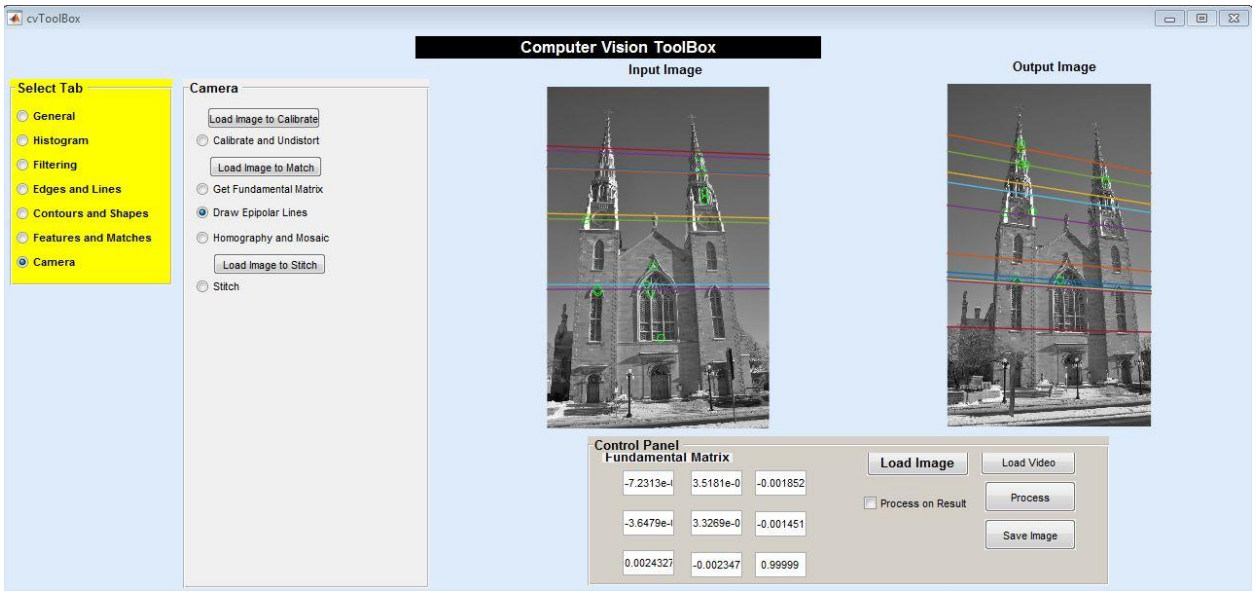

Fig. 23 Epiolar Lines

2.2.7.4 Homography and Mosaic

We can calculate homography between two images and use that homography to stitch two images (mosaic).

To do this:

1. Load image using 'load image' button.

2. Select the Camera tab.

3. Select Homography and Mosaic radio button.

4. Click load image to match' to load the other image to find corresponding points and there homography.

5. Click process button to show the homography matrix in control panel and warping of images to form mosaic.

This result is as shown in Fig. 24.

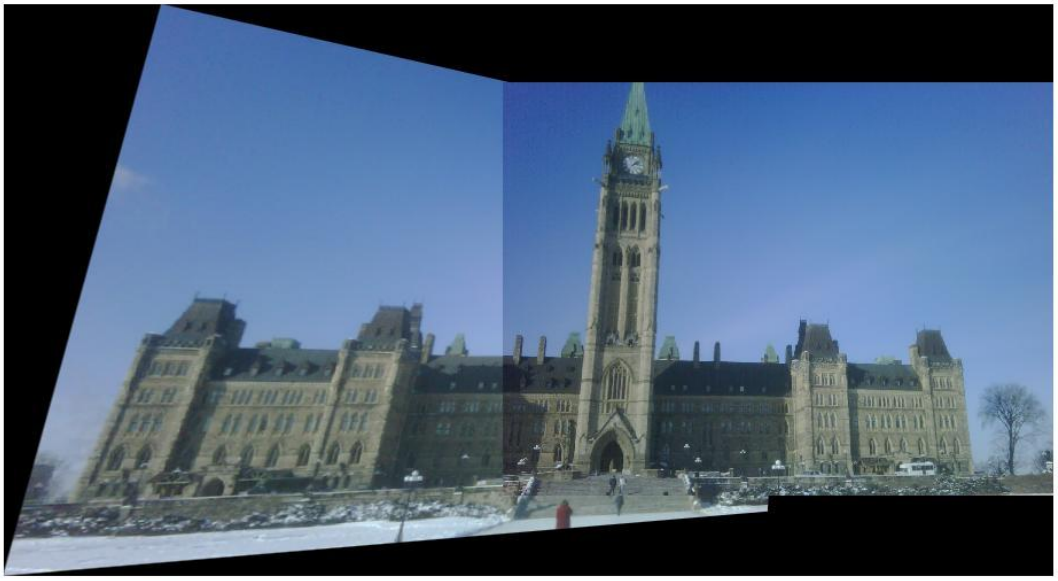

Fig. 24 Image warping after finding Homography

2.2.7.5 Stitching

We can stitch different frames together to form a panorama by making use of homography and warpingrecursively.

2.3 The Processing techniques on videos

I tried to implement some of the techniques on video sequences. Due to time constraints I could not do on all techniques but the main idea is same for any technique.

The techniques for which video would work are:

Add noise, Change Color spaces, Filter (both LPF and HPF).

In this module, I load the video and store them in frames and apply technique on frames and show the input video and output video as frames in input and output panel.

The demonstration can be viewed from gui.

To do this:

1. load the video using 'load video' button.

2. Select the processing technique (only the techniques said above).

3. Click process button.

4. We can view the input video in input panel and output video in output panel.

3. Conclusion

This project has given me a great exposure of the MATLAB as a powerful computer vision and image processing tool. This has also given me a great experience to learn many new things regarding GUI and epipolar geometry.

Future Work

Due to lack of time, the other techniques using video could not be implemented at the date of submission, but I would work on this toolbox further to develop it more sophisticated way and including more features like a history of processes and undo and redo a process like we do in commercialtoolboxes.

References

[1] Matlab documentation

[2] Mathworks Central

[3] Stackoverflow forums