CS 682 Assignment 2

Venkata Sasank Mudigonda

***Abstract*— This assignment summarizes my work on corner detection using Harris operator and multi-scale blob detection.**

I. Harris Corner Operator

A. Computing Harris Corners

For Harris corner operator, denoted by H, is defined by

$$H = \begin{bmatrix} A & B \\ B & C \end{bmatrix} = \sum_{(x,y)\in W} w_{x,y} \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

Are the features where you expected?
Most of the corners are detected at expected places with occasional failures to detect. This depends predominantly on the threshold set. The optimal threshold cannot be quantified accurately. But we can find a ballpark of how many we'd expect the Harris operator to detect. Plotting number of corners versus threshold gives a very good outlook of where our ballpark threshold must be. Fig. 1 shows the relationship between number of corners and threshold. Through visual examination, we can find that the optimal threshold occurs at about 0.2. At 0.2 threshold, we have detected 41 corners. The cubes image with corners plotted is shown in Fig. 2.

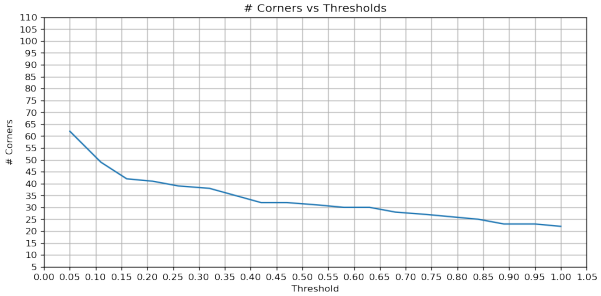

Fig. 1. Graph showing the relationship between the threshold and the number of corners detected

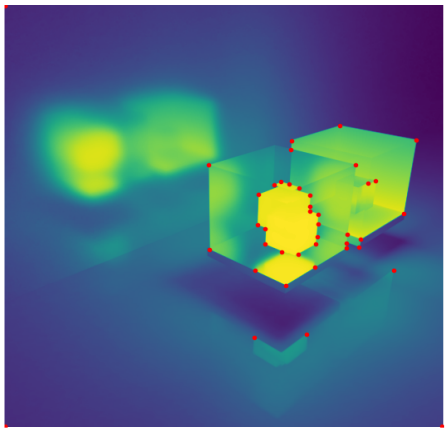

Fig. 2. Cubes image with corners marked with red dots

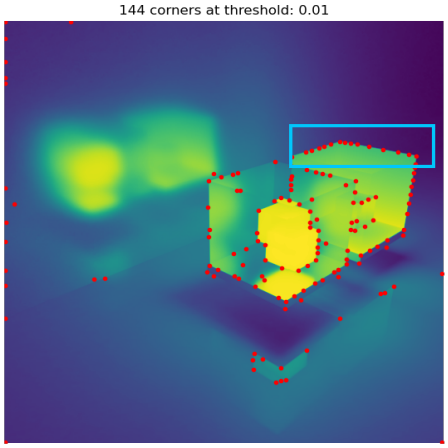

Fig. 3. Cubes image with corners marked with red dots

Are there any features in the image that you are surprised are not present? Highlight one or two regions of the image where features were detected that you did not expect (or one or two regions you thought features might exist).
Setting the threshold at 0.01, resulted in the detection of 144 corners. As highlighted in Fig. 3, there are a lot of corners detected along the edges of the cube on the right side (which are not corners). Even at this threshold, there are some corners missed on the shadow of the cubes. A lower threshold might catch them but at the cost of more false positives.

What would happen if you used a scoring function $f = trace(H) = A + C$? The trace of the Harris operators doesn't seem to be sensitive to the corners. See Fig. 4 where detected corners are plotted on the image. See Fig. 5 to find where the scoring function as been sensitive. It's mostly along the edges (both horizontal and vertical). Therefore it's not a good detector of corners.

What does this scoring function detect?
The trace of the Harris operator detects **edges** instead of corners.

If we want to detect corners, why might we not want to use this scoring function?
Since the trace of the Harris operator detects edges (both horizontal and vertical), one might argue it can be sensitive to corners, as it is clearly evident from the plot of the trace of the Harris operator. But when we apply local maxima, we

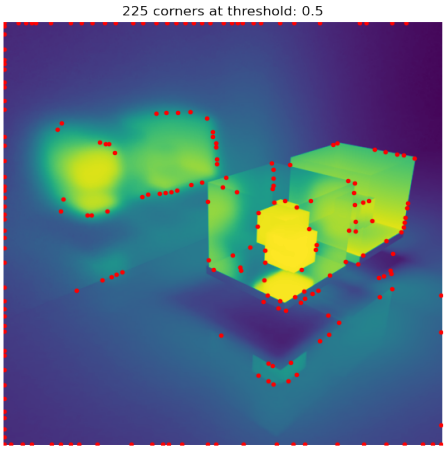

Fig. 4. Corners plotted with trace of the Harris matrix as the scoring function

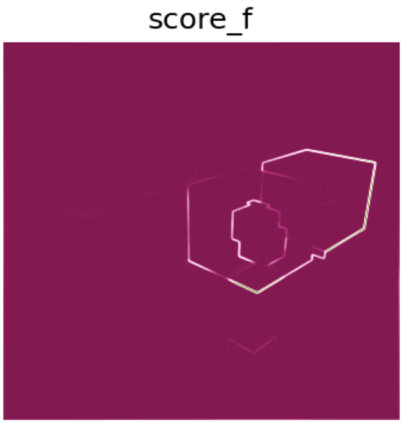

Fig. 5. Plot of the output of scoring function

might lose corners in the process. Hence it's not an effective way of detecting corners.

B. Varying the Weight Matrix

Discuss the differences between these four weight functions. In particular, what happens when the filter width (or σ) is very large? Increasing the filter size of the normal weight matrix hurt the number of edges detected from 41 corners at 5x5 to just 11 corners at 25x25. The increase in the size of the weight matrix suppressed the outputs of the scoring function drastically.

On the other hand, Gaussian exploded the number of corners detected at σ = 5 from 41 to 383. And, these corners are very close by to one another. This is because, the size of the Gaussian filter increased from comparable weight matrix of size 5x5 to 17x17 (size > 3σ), there by increasing the spread of the outputs resulting in an explosion of the detected corners. You can observe the outputs of the scoring function in Fig. 6. These experiments are done with a constant threshold of 0.2. With σ = 50, the filter size would go to at least 150 x 150. This tremendously increases the spread of the outputs of the scoring function. This makes it difficult to pinpoint the locations of the corners in the image.

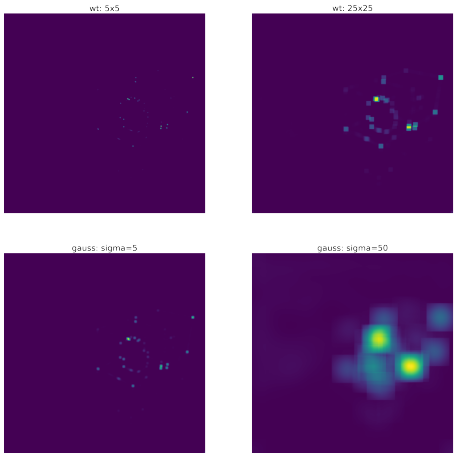

Fig. 6. Plot of the output of scoring function

What happens if we were to use a 1x1 weight matrix w=1 ? Why does this occur?
Using weight matrix of [1], it will not take any effect on the scoring function because it's an identity matrix of 1x1.

II. Multi-scale Blob Detection

A. Scale-Normalized Filter Response

What is the relationship between the peak σ and the circle's radius
The mathematical relationship between σ and radius r of a blob is given by

$$\sigma = r/\sqrt{2}$$

This is clearly seen from the plot in Fig. 7 and Fig. 8. In both the figures, an image with a blob of radius 8 is taken and LoG with σ = 1 and σ = 5 are applied. It's clearly visible in Fig. 8 that filter produced the peak value at σ = 5, implying that the blob has radius r 7.07 pixels.

In Fig. 9, the graph of maximum intensity values after applying LoG filter at various σ values is shown. There is a discrepancy between the graph the filter plots of the image. The filter plot clearly shows that the maximum intensity of the detected blob occurs at σ = 5. The graphs showed it occured at σ = 1. This could be because of the lack of normalization across the features outputs at various σ values.

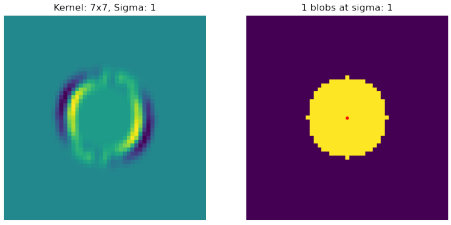

Fig. 7. LoG with σ = 1

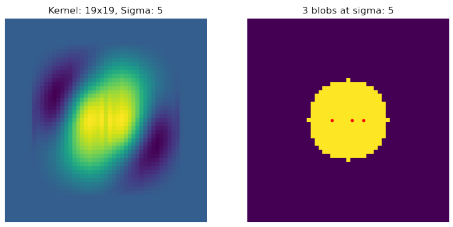

Fig. 8. LoG with σ = 5

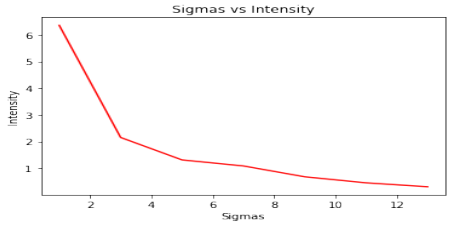

Fig. 9. Intensity vs σ plot

B. Annotating an Image with Multi-Scale Detections

Below are some samples of blob detection (Fig. 10, Fig. 11, Fig. 12). The white circles determine the blobs in the images with appropriate radii. Although limited sigmas have been captured, we could apply scale-normalized blob detection to detect more blobs. *In Fig. 7 and Fig. 8, the locations of the extrema of the LoG filter responses have been plotted on the original image.*

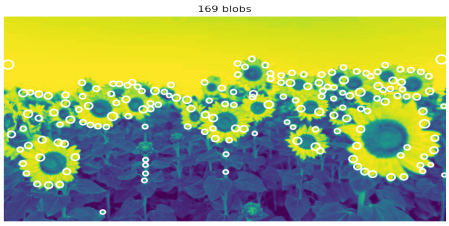

Fig. 10. 169 blobs detected, σ=[3, 5, 7, 9, 11]

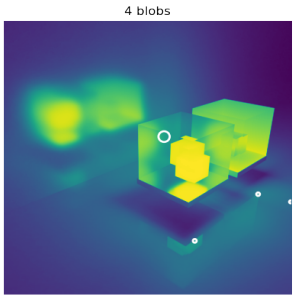

Fig. 11. 4 blobs detected, σ=[3, 5, 7, 9, 11, 23, 33]

III. Image Warping

How do the changes in these vectors correspond to changes in the warped image?

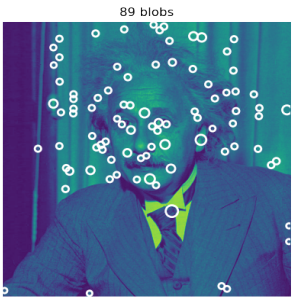

Fig. 12. 89 blobs detected, σ=[3, 5, 7, 9, 11]

For planar transformation, we can use the the matrix of the form given below:

$$\begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix}$$

The planar transformations have 6 degrees of freedom (translation along horizontal and vertical axes, rotation, scaling, sheer). The identity matrix I does not affect the image. The translation and rotation matrices look like this:

$$T(t_x, t_y) = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \quad R(\theta) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

If we want rotation followed by a translation, then we multiply R and T in the same order.

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = R \times T \times \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

The scale transformation matrix is given:

$$S = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The transformation given by the below matric is shown in Fig. 13. The transformation seems to have rotated the image by 90° degrees given by $R(90)$ with origin as midpoint of the y-axis.

$$\begin{bmatrix} 1 & -1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

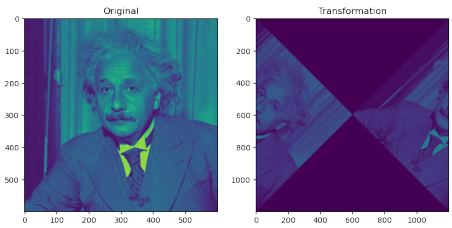

Fig. 13. 89 blobs detected, σ=[3, 5, 7, 9, 11]

How do the two "bottom row" parameters control how the image is warped?

The bottom row gives 2 additional degrees of freedom, out of plane rotation across both horizontal and vertical axes. The matrices show below are responsible for the transformations in Fig. 14 and Fig. 15.

$$H_{hori} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 50 \\ 0 & 0.001 & 1 \end{bmatrix} \quad H_{vert} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 40 \\ 0.001 & 0 & 1 \end{bmatrix}$$

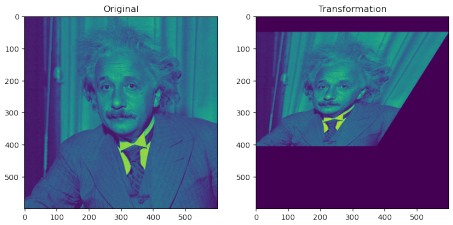

Fig. 14. Tilted horizontally (along the x-axis) out of the plane

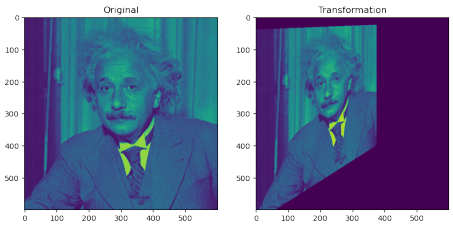

Fig. 15. Tilted vertically (along the y-axis) like a door, out of the plane

IV. Implementing Some Simple Feature Descriptors

The plots in Fig. 16, Fig. 17, Fig. 18 show the ability of the feature descriptors (raw pixels, binary descriptor, histogram descriptor) to match features across transformations (no transformation, contrast, high resolution, and transpose).

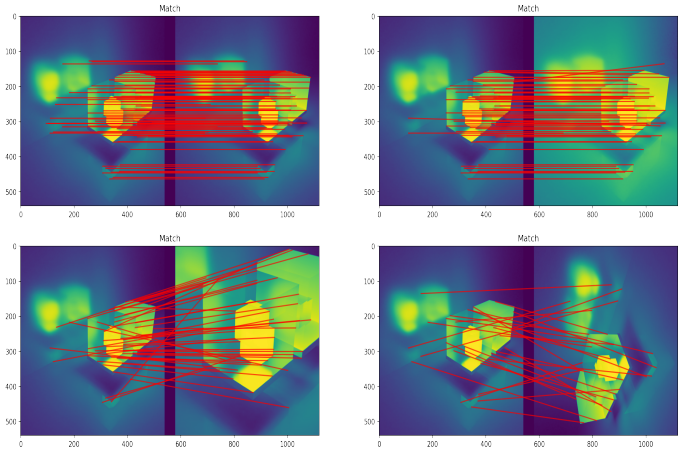

Fig. 16. Raw pixel values as feature descriptor

Which feature descriptor performs poorly on the image contrast? Explain why this descriptor performs worse than the others.

Histogram feature descriptor performs poorly on contrast. Intensity Histograms depend on distribution of the absolute intensity values in an image and thereby sensitive to the changes in those intensity values.

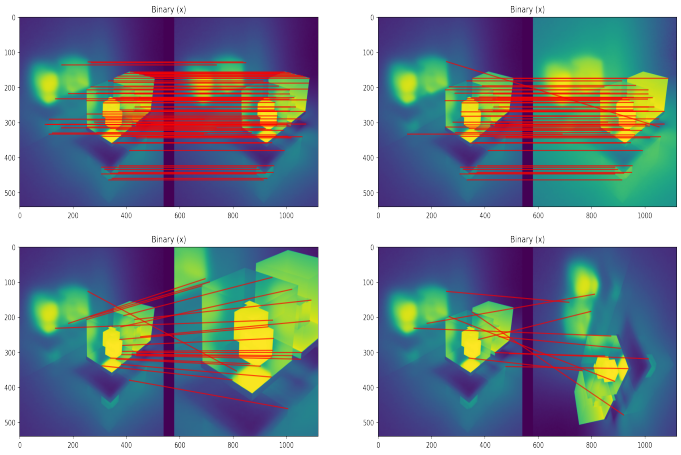

Fig. 17. Binary feature descriptor

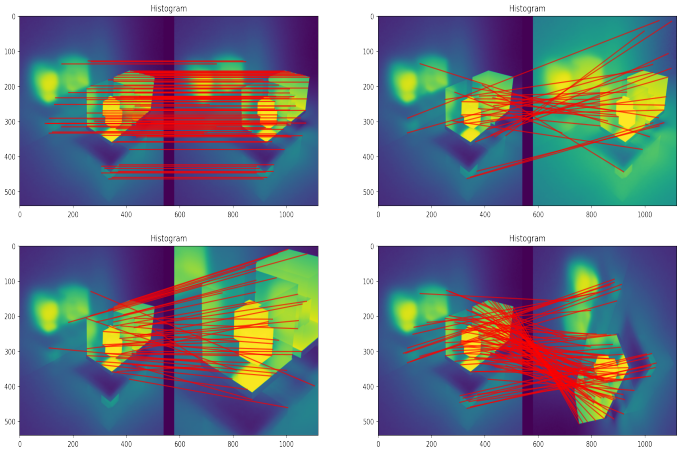

Fig. 18. Histogram feature descriptor

Which feature descriptor performs best on the image transpose? Explain why this descriptor performs better than the others.

Raw pixel values and binary descriptors perform poorly for match image features (corners) on transpose operation. These operations are ***not rotation invariant***.