CMSC 426: Computer Vision

Homework-1: Image Features and Warping

1. Coding- Harris Corner Detection

The image is first denoised to eliminate the sharp edges and avoid the derivative becoming undefined at that particular point. It is then converted to gray format, since it would be in 2D now instead of 3D (RGB colorspace). The sigma and window size are assumed to be 1 and the corner threshold value to be 10,000,000 based on trial and error from the values computed in the algorithm.

To compute the image derivative, first the Prewitt Filter is applied, i.e. Gx and Gy. After that, the image is convoluted in 2D using the Gx and Gy values to get Ix and Iy in X and Y directions respectively. They are multiplied with each other to compute the product of derivatives at every pixel. Next, I(3x3) matrix is used for convoluting over the product of derivates at every pixel and we get Sx2, Sy2 and Sxy.

Now, to fetch the Matrix at every pixel (x,y), for loop using the dimensions of the grayscale image is executed and H = [Sx2(x,y), Sxy(x,y); Sy(x,y) Sy2(x,y)] values are computed. Finally, the Response of the Detector at each Pixel is calculated using R = det(H) – k*(trace(H)^2). The threshold is thereafter applied to separate the most significant corner points and those values are stored in a im(x,y) matrix.

Lastly, the Local Maxima is found out using the ordfilt2(I, 9, ones(3x3)) function to find the pixel with the maximum value out of 9 pixels inside the window. The Harris Points are found using the local maxima and corner threshold values which is further plotted on the actual image to indicate the Harris Corners. The heatmap of the im(x,y) matrix is also plotted which is displayed in black & white format because of the pixels having values either 1 (for high contrast) or 0 (for low contrast).

2. Short Answer

1. Harris Corner Detector Properties

The orientation of the image doesn't matter when it comes to detecting features as it is just a collection of pixels. The Harris corner detection algorithm works on finding the derivative of the image and it doesn't matter if it is upright or inverted. Since we would have two transformed versions of the same images, features should be detected in the corresponding locations as the corners are still the same. Also, the second moment ellipse would rotate, but its shape (i.e. eigenvalues) would remain the same. Thus, the corner location is covariant w.r.t. rotation. However, it's not the case with scaling. The points who would once be corner points inside the window, might now be falling under edges because of scaling of the pixels as well. Thus, the corner location is not covariant to scaling.

Upon incrementing the value of all pixels by a constant, there are hardly a few more Harris corners detected up to when the value of the constant is increased by 40-50 units. Above that, the number of Harris corners detected increases (along with the original ones) as the intensity of the image increases. Similar is the case when all the pixel values are multiplied by a constant. Here, a considerable rise in the number of Harris corners is observed as the intensity of the image is also multiplied. Both the mentioned cases are called Affine Intensity Change (i.e. I -> I + c or I -> k*I). Thus, it can be said that corner location is partially invariant to affine intensity change.

2. Image Warping and Invertible Transformations

In the first case, known as Forward Warping, when each pixel from the original image is computed to the nearest pixel location of the transformed point in the warped image, there are chances of the pixel landing between two pixels. At that point, the color is distributed among the neighboring pixels. This is known as splatting. While in the second case, known as Inverse Warping, the destination pixels in the warped image are looped using the transformation to identify the nearest pixel in the source image. Here, there are chances of the pixel coming in between two other pixels. At that point, the color from the nearest neighbors is interpolated. One of the primary reasons of using Inverse Warping is that it eliminates holes which is not the case for Forward. However, it requires an invertible warp function, hence it might not be possible always.

References

1. http://www.cse.psu.edu/~rtc12/CSE486/lecture06.pdf
2. https://www.mathworks.com/help/images/ref/imwarp.html
3. http://alumni.media.mit.edu/~maov/classes/vision09/lect/09_Image_Filtering_Edge_Detection_09.pdf
4. https://alliance.seas.upenn.edu/~cis581/Lectures/Fall2016/CIS581Fall16-09.pdf
5. http://www.cs.cornell.edu/courses/cs4670/2015sp/lectures/lec07_harris_web.pdf
6. https://www.mathworks.com/matlabcentral/answers/67114-what-is-the-difference-between-imwarp-and-imtransform

Dipam Patel
UID: 115809833