ASSIGNMENT 2 NAME: SHREYA BISWAS STUDENT ID: 40018190

The goal of this assignment is to find feature points in an image and match the feature points in two similar images. The three main steps involved is:

* Feature Detection
* Feature Description
* Feature Matching

FEATURE DETECTION

I have used Harris Corner detector to compute the feature points in an image. Harris corner detector takes the gradient of each pixel in a window and computes the Harris Matrix. We can compute the corner strength a pixel using the formula:

Corner strength = det(H)/trace(H)

I have used a threshold of 60% to get the feature points with highest corner strength. I have added non maximum suppression to ensure that each feature point is a local maxima in its 3X3 neighborhood.

FEATURE DESCRIPTION

Once we find the feature points we compute a feature vector for each feature point to describe its neighborhood. I have used SIFT to compute feature descriptor for a feature point. SIFT takes a feature point and gets the 16x16 neighborhood around it. It divides the 16x16 neighborhood into 4x4 grids. For each 4x4 grid, we find the orientation of the gradient and compute a histogram of 8 bins. We then normalize the 1x8 dimensional vector so that each bin in the histogram has a height <=0.2. If a bin in the histogram has a height>0.2, we clip it to exactly 0.2. We normalize the final vector again. This ensures that our feature descriptor is contrast invariant. So finally, we get a 16\*8 = 128 dimensional vector for each feature point.

FEATURE MATCHING

We generate feature descriptors for each feature point in the two images, then find the best match for each feature point in image1 and image2. For each feature vector in image1, we calculate the SSD distance to all feature points in image2. We pick the pair with the least distance and match them.

METHODS USED:

harris\_corner(srcImage): Detects corner with higher corner strength

non\_maxima\_suppression(cornerArray, window): Checks whether each keypoint is local maxima within a window, if not then it is discarded

adaptive\_max\_suppression(harrisPoints): Computes distance between each keypoint and sorts them in descending order of distance and takes only the ones with highest distance between them. This method is used to achieve adaptive maximum suppression.

sift\_desc(image, features): Computes the 128 dimensional vector to describe each keypoint and its neighboring area

rot\_invariance(grid\_angle): Computes the angle of the keypoint and rotates its neighboring pixels to its angle. This is used to achieve rotation invariance

get\_histogram(angle, magnitudes): Caluclates the 8 bin histogram for a 4x4 grid around a pixel

normalize(histogram): This method normalizes the histogram to 0.2 that is if any of the bins in the histogram has a value greater than 0.2, it is clipped to 0.2

neighbors(radius, rowNumber, columnNumber, arr): This method returns the neighboring pixels of a pixel in the image at a certain radius

get\_featureMatch(f\_desc1, f\_desc2): This method is used to match the features between to given images

HOW TO RUN THE CODE:

1. Extract zip file
2. Open FeatureMatching.ipynb in Jupyter Notebook
3. Enter the proper paths to the images in the main() function
4. Press ctrl+Enter
5. The Keypoints for feature detection of images are saved in feature\_image1.png and feature\_image2.png for image1 and image2 respectively
6. The feature matches of the pair of images is saved in output.png

REFERENCES:

* Github https://github.com/cuitianyuan/CV-FeatureCreationAndMatching
* Github https://github.com/saitejaprasadam/Image-KeyPoint-Matching
* https://muthu.co/harris-corner-detector-implementation-in-python/
* https://medium.com/@lerner98/implementing-sift-in-python-36c619df7945
* Lecture Slides
* Multi-Scale Oriented Patches, Matthew Brown, Rick Szeliski, Simon Winder
* A combined corner and edge detector, Chris Harris & Mike Stephens
* Object Recognition from Local Scale-Invariant Features, David G Lowe