Cristian Mitroi
Kaela DeAngelis
Terne Thorn Jakobsen

VISION & IMAGE PROCESSING: ASSIGNMENT 2

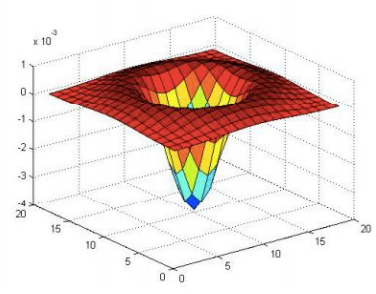
Exercise 1: Detecting Interest Points (Features)

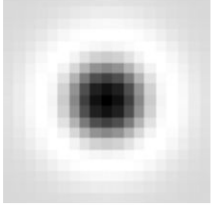
Blob detectors are a complementary technique to edge and corner detectors when processing an image; they provide a description of features within a region by use of a radius estimate rather than by curved line segments or by intersection of edges (6). The method chosen to detect features within the three target images was based on the Laplacian of a Gaussian (LoG) (3). This method combines a Gaussian smoothing kernel with a Laplacian operator prior to convolution with an image. During Gaussian smoothing, the kernel is parameterized by a scale value t , or σ, that provides a scale space representation (6):

$$g(x, y, t) = \frac{1}{2\pi t} e^{-\frac{x^2+y^2}{2t}}$$

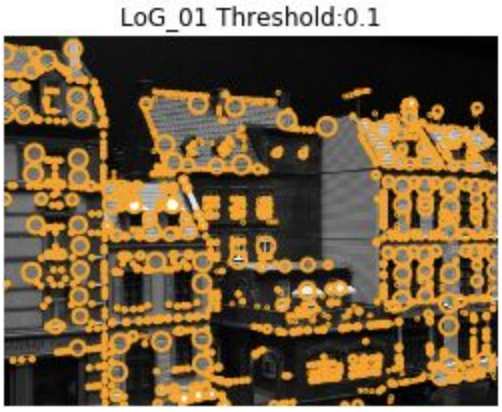
Scale space representation manipulates image data of different scales to create scale-invariant feature detection. As seen in the prior assignment, increasing the parameter σ reduces level of detail. Therefore we expect that as value t changes, the ability of the LOG filter to detect blob features will reduce. The blob detection filter obtained from Scikit-image calls for maximum and minimum σ values (8). Based on documentation, we will set a max_σ and then a num_σ parameter, while leaving the minimum σ value at default. The num_σ parameter allows the algorithm to discover blobs within the image at different sigma values between the max and the minimum σ value of 1. The output of this function is the x- and y- coordinates of the blob and the σ of the Gaussian kernel at which the blob was detected. This is an ideal option as not all blobs are a similar size. Therefore we expect to see greater variation in blob detection by use of the num_σ, accounting for fine and broad changes in brightness intensity. We can also expect variation in blob detection as the images change. This claim is supported by the angles at which all three images were taken. As the angles shift between images, brightness intensity changes due to lighting affecting size of the blobs and detection of new blobs occur because new features (e.g. larger portion of the car) are revealed and old features are lost. We do expect blob detection within areas such as the black background or the large white paneling of the leftmost building because these areas have sharp linear changes in brightness intensity (not circular) when compared to their borders.

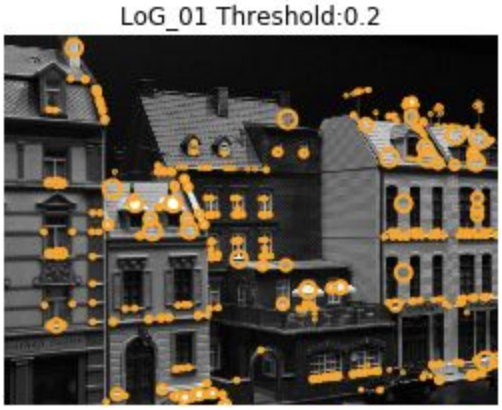
The features within a blob are considered to similar to each other, meaning the hold similar values in brightness intensity (6). Blob detection responds best to grayscale values meaning white and black spots on a white and black background. These types of areas mark a point where intensity stops increasing and starts decreasing for dark blobs and the reverse for bright blobs. Changes in brightness intensity of neighboring pixels triggers a circularly symmetric maximum response by LOG, also referred to as local extrema (see images below); the peak represents the center of the blob while the pits mark the edges (5).

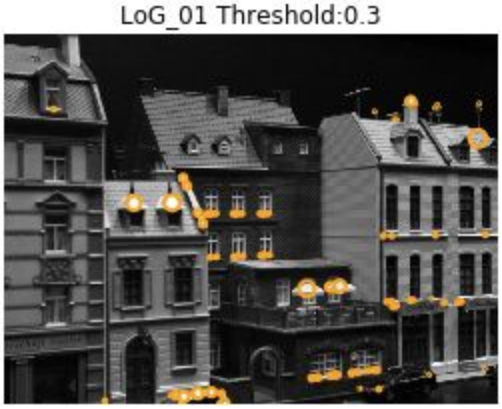




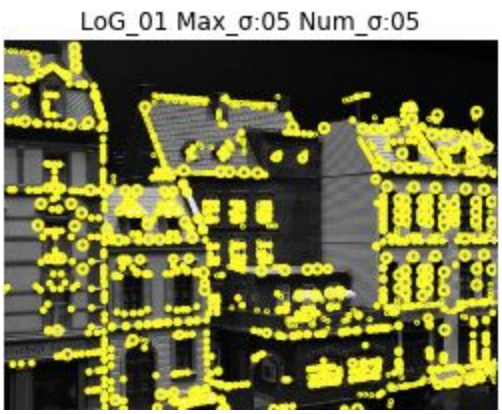
Therefore our objective was to find a set of parameters where the Laplacian operator achieves maximum response to the changes in brightness intensity with a diameter of $2\sqrt{\sigma}$. Our results demonstrated that increasing the value of the threshold for local maxima produced a reduction in blob detection for Image_01. Output images hereafter are labeled LoG followed 01, 02 or 09 and were displayed using matplotlib (4).

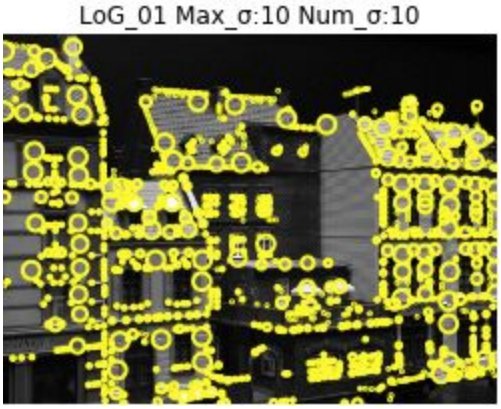


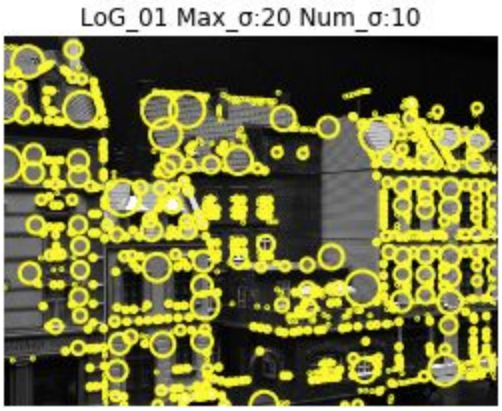




Therefore we set the threshold parameter to a fixed value of 0.1 to display the most amount of scale space maximas possible. As the reader may recall, we previously discussed how we expected the scale of the blob to be determined by the σ parameter. Therefore we compared the following values within Image 01: 1. max_σ =05, num_σ =05, 2. max_σ =10, num_σ =10, and 3. max_σ =20, num_σ =10 (see below). Based of this experiment, the image manipulated by max_σ =10, num_σ =10 presented an ideal set of parameters.

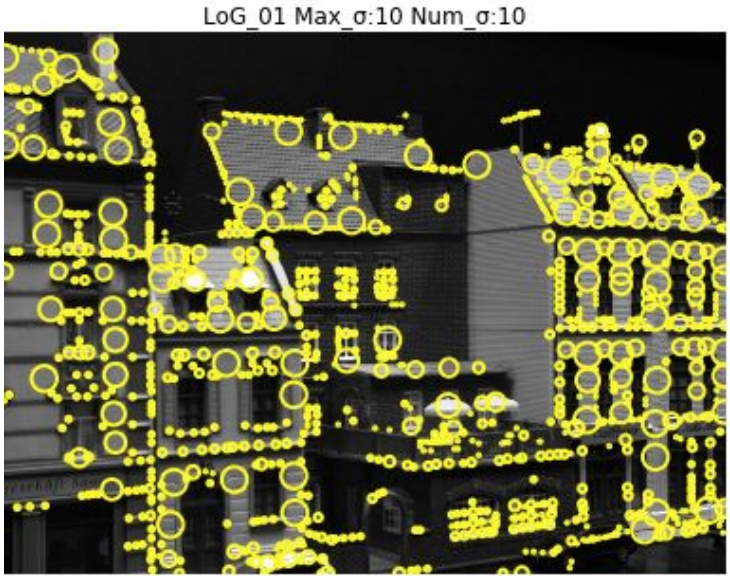


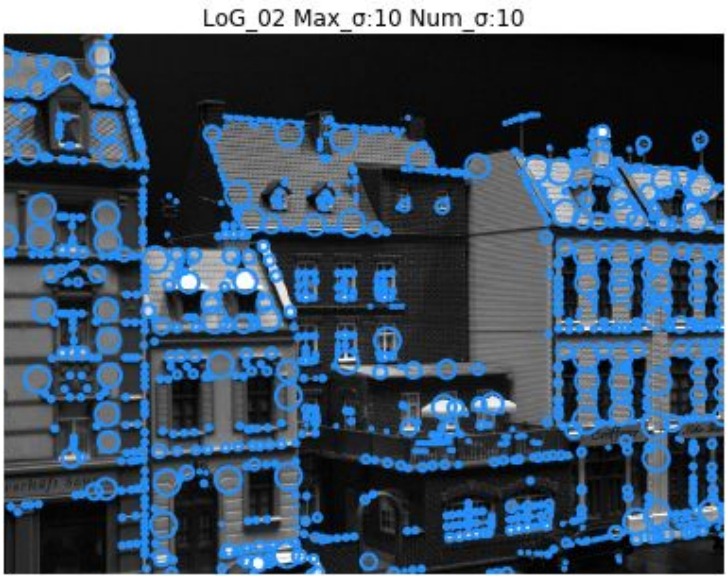


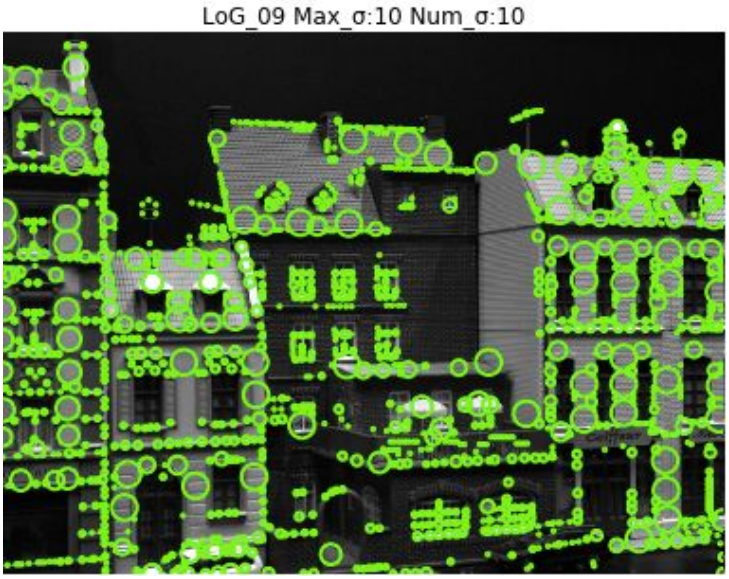


This is due to the idea that the first set of values was too small to detect blobs of larger intensities which can be seen in the leftmost building by the lack of blobs between the panels when compared to the second and third set of parameters. However, the first and second set of parameters detected blobs of very fine detail evident by clear blobs the building's windows when compared to the third set, which blobs appear sloppily overlapped inside the windows. Therefore the second set of parameters was ideal because of the

happy medium between detecting fine and broader blobs. Thereafter we applied the parameters max_σ=10, num_σ =10, threshold=0.1 to images 001 (yellow), 002 (blue), and 009 (green). As expected there was not heavy variation between the images in terms of fine and broad blob detection. However, variation in blob detection did occur where the image revealed more obvious shifts in angles, particularly in the perimeters of the images and the borders between buildings.

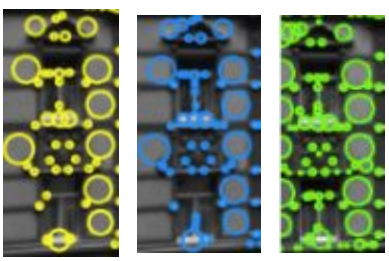




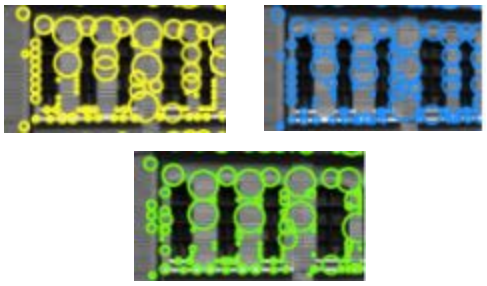


LoG_09 revealed more information about the surrounding objects than the other two because of the angle it was taken at, which was more straightforward. This shift most notably occurred in the following closeups ;

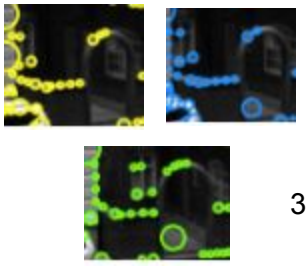
A.

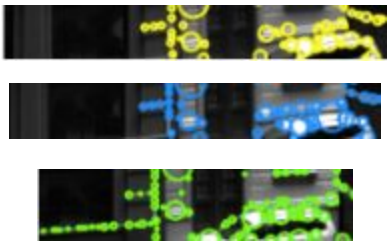


B.



C.



D.  E. 

The camera revealed more of the hallway in at the loss of the window (C), revealed a leftmost fourth window in the middle floor (D), and revealed a better view of the car (E). These shifts caused difference in blob detection, as different features were made available for filtering. Closeups A and B of all three images revealed that, while Image LoG_09 provided a more head-on view, it cropped out the fifth rightmost window (B) and cut off the left-side paneling next to the window. What these closeups revealed is that Image LoG_01 and Image LoG_02 were most similar in blob detection due to the similarity in the angles at which the images were taken. There was also noticeable overlap in detected blobs in all three images, most noticeably in Images LoG_01 and LoG_02 (see closeups B and E). This effect can also be attributed to the angles at which they were taken because lighting in the image had an affect on the threshold of brightness intensity. One last comparison was that the black background as well as the paneling between buildings in closeup D did not trigger blob detection. Despite a difference in brightness intensity between these areas, this output revealed that the algorithm was in fact working. These regions were not symmetrically changing brightness intensity but rather doing so linearly, making these areas better detected by edge and corner detectors.

Exercise 2: Simple Matching of Features

The second part of the assignment deals with the "correspondence problem" and feature matching between two (or more) separate images of the same 3D scene. Briefly, the "correspondence problem" tackles the challenge of identifying which objects and/or points in one image match objects and/or points in the second image (or multiple other images). This is called *stereo fusion*. The challenges faced are usually ones of (1):

- camera having been repositioned
- objects in the view having been repositioned
- the passing of time
- occlusion due to differing point of views

Panorama creation (or "image stitching") is one typical scenario where this problem is encountered. In this case the process needs to identify which sections of the photographs overlap, so that they can be joined together at those particular points.

There are two types of algorithms that address this problem:

- Correlation-based algorithms
    - Yield a "DENSE set of correspondences" (2) by searching for the best match of a specific "patch" of an image (7).

- Feature-based algorithms
    - Yield a "SPARSE set of correspondences" (2) by searching for features in the images and verifying whether the arrangement is similar between the images (7).

In this exercise, we use a correlation-based method for establishing feature correspondences between two images. Features given by the Laplacian of Gaussian blob detection in exercise 1 are used for matching the images taken at slightly different angles. We match the three images, in turn, with each other, using four different patch sizes of length N = 5, 7, 9 and 13. By calculating the sum of squared intensity differences we find the patch in the other image that matches the current patch closest (9).

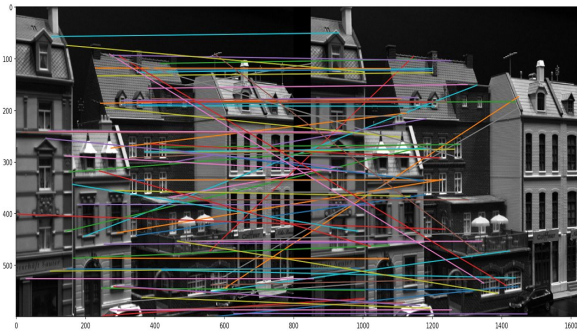
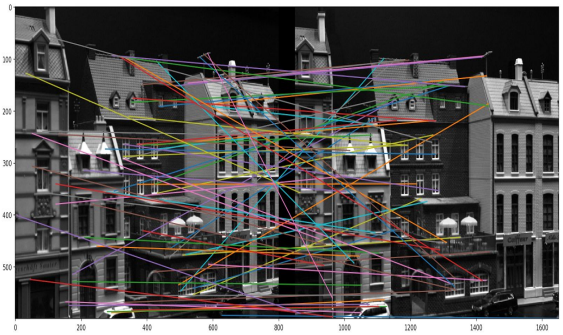
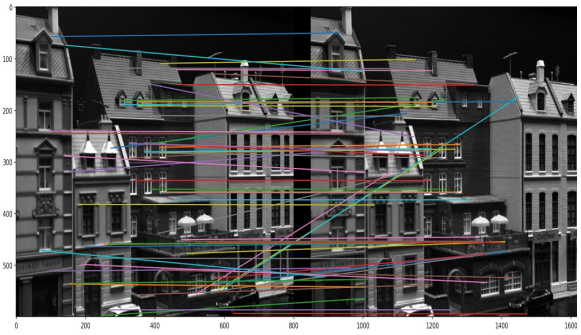
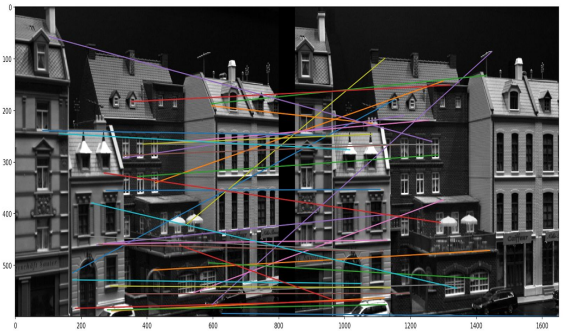
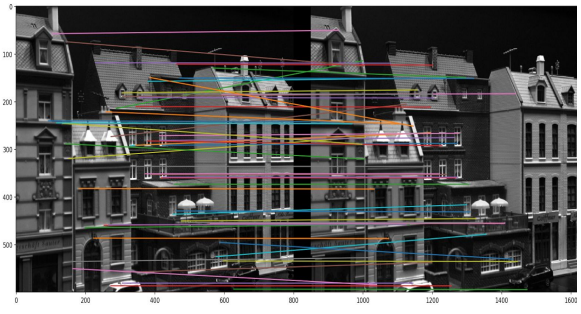
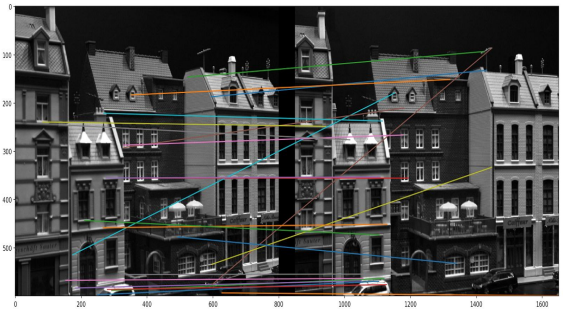
In regards to the relevant statistics of results, the original keypoints (1052 for img1, 1094 for img2, 1276 for img9) detected in exercise 1 had to be refined in order to retrieve more accurate results (see *Table of acceptance criteria*). Therefore we chose to implement an algorithm that checks the ratio between the smallest and the second smallest dissimilarities found for a given keypoint patch, setting the threshold to 0.7. This means that keypoints between Image_001 and Image_002 and also Image_001 and Image_009 were not only matched via their closest match, but also by identifying the second best match. If any of the resulting ratios fell above 0.7, they were considered false matches and were subsequently eliminated. After implementation, nearly half of the keypoints were eliminated from all three images. These output numbers slightly increased as patch size increased from 5 to 7 to 9 and to 13. Relatedly, left-right and right-left correspondence removed nearly half of the keypoints between Image_001 and Image_002, and Image_001 and Image_009 thereafter. This number decreased as patch sized increased.

Case	Removed by second lowest/lowest ratio - left	Removed by second lowest/lowest ratio - right	Removed by left-right / right-left matching
Result_001-to-002-patch-size-5	499	541	473
Result_001-to-002-patch-size-7	563	594	425
Result_001-to-002-patch-size-9	581	616	411
Result_001-to-002-patch-size-13	596	615	391
Result_001-to-009-patch-size-5	511	669	467
Result_001-to-009-patch-size-7	588	733	420
Result_001-to-009-patch-size-9	626	789	394
Result_001-to-009-patch-size-13	614	799	395

Table of acceptance criteria

Higher patch sizes of N = 13 greatly improved matching between img001 and img002 (See *Table of results* below and images generated by our script). This makes sense since more pixels are then

included in the patch, giving the algorithm more information to work with. However, when matching img001 to img009, increasing patch size to N = 13 makes matching worse. The reason for this may be that the two images, 1 and 9, are more different – i.e. taken from more different point of views. Matching img001 and img002 seems to provide better results because the images lie on the same epipolar plane while img009 is taken from a higher angle, meaning that the epipolar lines are not aligned. This problem is due to the epipolar constraint (10). Typically, you will not need to search the whole image to find matches but can conveniently limit the search along the epipolar lines. But when the epipolar plane is not aligned, the matches are visibly more skewed/error prone. To get more accurate results, it could be useful to do *image rectification* where image coordinates are transformed in order to have a common image plane (11). Also, having taken the images from different angles changes the light exposure, which affects the features being detected, meaning that the features of the images are essentially quite different.

N	Img001 to Img002	Img001 to Img009
5		
7		
9		

13

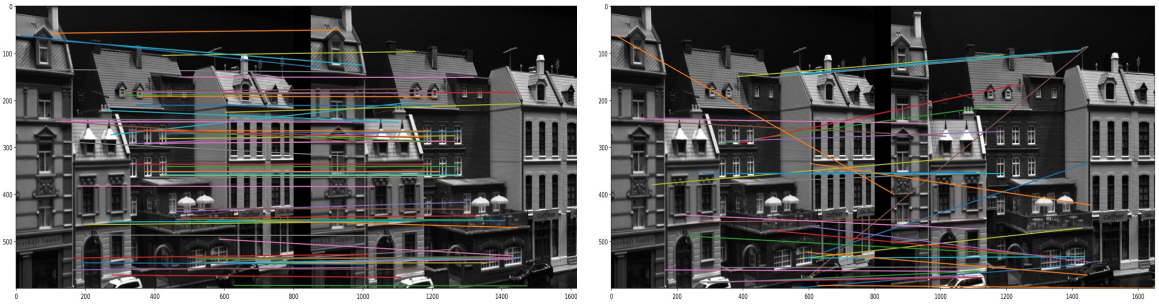


Table of results

Table of dissimilarities below gives an overview of the mean and standard deviations of the dissimilarity measure for the accepted matches. There is evidently very large standard deviations, meaning large variation between the dissimilarities. Unsurprisingly, the standard deviation is greater when patch size is increased and the standard deviation is specifically high when comparing img001 to img009 – the images that are most different in respects to angle/point of view.

	Mean of dissimilarity	Standard deviation of dissimilarity
001-to-002-patch-size-5	345	335.852311059
001-to-002-patch-size-7	866	596.546138943
001-to-002-patch-size-9	1597	890.387364562
001-to-002-patch-size-13	4297	2691.29129756
001-to-009-patch-size-5	374	278.032163756
001-to-009-patch-size-7	1115	851.138977596
001-to-009-patch-size-9	1981	1688.84114808
001-to-009-patch-size-13	4276	3336.05265199

Table of dissimilarities

Citations

1. Bebis, G. (2004) The Correspondence Problem. University of Nevada, Reno. Retrieved from https://www.cse.unr.edu/~bebis/CS791E/Notes/StereoCorrespondenceProblem.pdf
2. Collins, R. (2007) Lecture 07: Correspondence Matching. Penn State University. Retrieved from http://www.cse.psu.edu/~rtc12/CSE486/lecture07.pdf
3. Collins, R. (2007) Lecture 11: LoG and DoG Filters. Penn State University. Retrieved from http://www.cse.psu.edu/~rtc12/CSE486/lecture11_6pp.pdf
4. Hunter, J.D., et al. (2007) Pyplot: matplotlib.pyplot. (Matplotlib 1.2.1 documentation) [code documentation] Retrieved from https://matplotlib.org/api/pyplot_api.html
5. Lazebnik, S. (2011) Lecture 8: Blob Detection. University of North Carolina at Chapel Hill Retrieved from http://www.cs.unc.edu/~lazebnik/spring11/lec08_blob.pdf
6. Bob Detection (2017) Wikipedia, The Free Encyclopedia. Retrieved from https://en.wikipedia.org/wiki/Blob_detection
7. Correspondence Problem (2015) Wikipedia, The Free Encyclopedia. Retrieved from https://en.wikipedia.org/w/index.php?title=Correspondence_problem&oldid=661209488
8. van der Walt, S., et al. (2014) Blob Detection. (Scikit-lmage 0.14 development) [code tutorial]. Retrieved from http://scikit-image.org/docs/dev/api/skimage.feature.html#skimage.feature.blob_log
9. Forsyth, D.A., & Ponce, J. (2003) Chapter 7: Stereopsis. *Computer Vision: a Modern Approach* (pp. 207) Upper Saddle River, Nj, USA: Prentice Hall.
10. OpenCV – Python Tutorials – Camera Calibration and 3D reconstruction Retrieved from https://docs.opencv.org/3.2.0/da/de9/tutorial_py_epipolar_geometry.html. Visited on 12/12/2017.
11. Epipolar geometry. (2017, December 10). In Wikipedia, The Free Encyclopedia. Retrieved 13:04, December 12, 2017, from https://en.wikipedia.org/w/index.php?title=Epipolar_geometry&oldid=814772533