Machine Perception Assignment One

Due Date: Week 10 - Monday 5 October 2020 at 5pm

Kwaramba, Tawana (19476700)
Faculty Of Science And Engineering: School of Civil And Mechanical Engineering
Curtin University
Bentley WA, Australia
19476700@student.curtin.edu.au

***Abstract*—write at this at the end of your report :)**

***Index Terms*—Histograms, I don't know what this is tbh...**

I. Introduction

do this are the end

II. Task One: Image histogram, Harris corners and scale-invariant feature transform (SIFT) key points

The Harris corner detection algorithm, and the SIFT algorithm can be thought of algorithms which will pick the key features of an image with the pre-dominate implementations of these algorithms detect the corners of the image [2] [3]. Corners can be thought of as regions in an image with large variation of intensities in all directions [2]. Therefore, the SIFT and the Harris algorithm, are algorithms for detecting the corners in an image the key differences is the invariance and variance to certain image transformations [3]. Harris corner detection algorithm is mainly invariant to rotation, and the SIFT algorithm is mainly invariant to scaling and is also invariant to rotations due the orientation assignment stage of the SIFT algorithm [3], which is explored in greater detail in section III subsection IV-E.

A. Harris corner detection

The Harris corner detection is an algorithm whereby a window function will scan through the image to locate local maximums, these local maximums can be thought as potential candidates for key features detected by the Harris corner detection algorithm [4] [2]. This following behaviour can be modelled by the following equation:

$$E(u,v) = \sum_{(x,y)} w(x,y)[I(x+u,y+v) - I(x,y)]^2 \tag{1}$$

Whereby:

- $w(x,y)$ represents the window which is scanning over each section of the image,
- $I(x+u,y+v)$ represents the intensity neighbouring pixels, and
- $I(x,y)$ represents the intensity of the current pixel.

Thereafter, the function is through taylor expansion to form the following system of equaitons

$$E(u,v) \approx [u\ v] M \begin{bmatrix} u \\ v \end{bmatrix} \tag{2}$$

Whereby, M is represented by the following equation

$$M = \sum_{(x,y)} w(x,y) \begin{bmatrix} I_x I_x & I_x I_y \\ I_x I_y & I_y I_y \end{bmatrix} \tag{3}$$

Thereafter, a plethora of potential corners are found in the image matrices, the Harris algorithm will then go through each window function to determine if the found corner is most likely going to be an actual corner, this process is determined by the following equation [2]:

$$R = det(M) - k(trace(M))^2 \tag{4}$$

Where by $det(M)$ is the multiplication of eigenvalue values of the matrices M, and $trace(M)$ is the addition of the eigenvalue values of the matrices M [2]. Therefore, if one of the eigenvalue values is significantly greater than the other eigenvalue value those found points are most likely to be an edge, and if the eigenvalue vales are large and approximately the same value that point is a corner otherwise, that point is a flat region [2].

Harris corner detection is an algorithm which can be mainly thought of as a rotation invariant algorithm meaning, no matter the angles you rotate a given image, the algorithm is going to detect the same key-features [2] [4]. This is due to that an corner will always going to remain a corner no matter what orientation the image because a corner is just a high intensity pixel, and the algorithm only looks at a pixel and it's neighbouring pixels thus, no matter what orientation the image the pixels will still have the same neighbouring pixels [2]. Albeit, if you scale the image, a pixels neighbouring pixels will be transformed hence, the window function may determine those clusters of pixels as a non-corner when it's a corner, or as a corner when it's a non-corner. Therefore, the conducted experiments should show consistencies through the set of rotate images, and inconsistencies through-out the scaled images.

B. Harris Corner detection: Results - Diamond

1) Rotational Results:

2) Scaling results:

C. SIFT Results - Diamond

1) Rotational results:

2) Scaling results:

D. Harris Corner detection: Results - Dugong

1) Rotational Results:

2) Scaling results:

E. Discussion about produced results

To determine if an image is invariant to a certain transform we have to determine how many corners that image is detecting relative to the performed corners. Hence, the first experiment is to get the returned corners from the Harris corner detector, and the SIFT transform and find the difference relative to the original image for each transform. Therefore, for perfect results we would assume that the difference should be 0 for invariant feature as they should have picked up the same number of corners in the image. Although, to greater consolidate this claim, we must determine if the corners are been picked up in the same region hence, we calculate the histograms of the image in the same channel as the corners, and theoretically for invariant features the histograms must follow perfectly after each other as the detected pixels should lay in the same region. Finally, to determine how well the histograms follow each other we calculate the distance between each transformed histogram relative to the original histogram of the image.

Look at figure 31 we can see that through each rotated image, they is irregularities in the number of key points found in each transformed image. It's natural to conclude that this experiment dis-prove Harris corner detection invariance to rotation. Albeit, we have to consider the manner the image is rotated. As can be seen in figures **??**, 33, 34, and 35 you can see when you rotate the image they is an increase of black space around the image, and the boarders of the image may have been picked up as a corner. In figure 32 you see the boarders of the image are detected as a corner but, in the other images the boarders are not detected as a corner. Therefore, with this observation we should expect irregularities in the data produced. Furthermore, revisiting figure 31 we can see the produced histograms follow a clear sinusoidal pattern implying that although they is irregularity between the number of corners found in each transformed image but atleast they is consistency of the number of corners found relative to the angle of rotation of the image. Therefore, we can infer that the Harris corner detection algorithm would produce the same number of corners if each rotated image had a tightly bounded box around the image thus removing the corners picked up by the boarders of the image. Therefore, en-counting for the extra corners found due to the boarders of the image we can infer that the number of corners found for each transformed image will be very similar to the original transformed image. This same concepts will apply for the images found in the dugong images in section**??**.

Looking, at the scaled results of the experiment we will expect the same to hold for invariant properties for the image, and the opposite to be true for variance to a specific image transform. Therefore, we should be expecting no clear pattern in the produced histograms for the first experiment, we should see great irregularity in the produced histograms, and we should see great irregularity in the distances between the produced histograms. Therefore, as seen in figures **the key point figures in the report** they is great irregularity in the produced histograms hence implying variance to scaling transformations. The same can be said for SIFT.

En-counting only for the numbers of corners found by the Harris corner detection is not enough to prove the invariance of the algorithm. For the diamond playing card the only two visible colors are white, and red hence we can use the cards characteristic to our advantage by drawing the found corners in the green channel, the same can be said about the dugong image by drawing the found corners onto the red channel as seen in section II-B. The aim of this experiment to see if the found corners are going to be laying in the same color space as the original image hence, for the playing card we should get a spike of green pixels around the cluster of red pixels. Referring, to figure 37 we can see that the histograms overlay over each other for the mid intensities, and they is some deviation in the lower intensities. Deviation in the lower intensities of the image is expected as in some transformed images we're introducing more black pixels into the image because of the method we're using to rotate the image as seen 32, 33, 34, and 35. Therefore, as we can see the produced histograms in figure 37 support the invariance of the Harris corner detection.

Furthermore, this is further consolidated through the calculated distances between the transformed image, and the original image. As we can see in figure 31 the distances found in each transformed image is the same. Given, the introduction of greater quantity of black pixels due to the manner of rotation of the image we would expect the histograms to be a fixed distance away from each other which, is demonstrated through figure 31. Therefore, as demonstrated the Harris corner detection rotationally invariant as show through the diamond card.

III. Task Two: Image Features

an example on how to refernce other sections V

A. part i: the main steps in Liner Binary Pattern

The linear binary pattern (LBP) algorithm is an algorithm which is mainly used for feature extraction in relation to image textures [5].

The main pipeline steps of the LBP algorithm can be summarized as the following:

- Convert the image into a gray-scale image
- Calculate the LBP for each pixel in the image
- analyse the produced LBP's

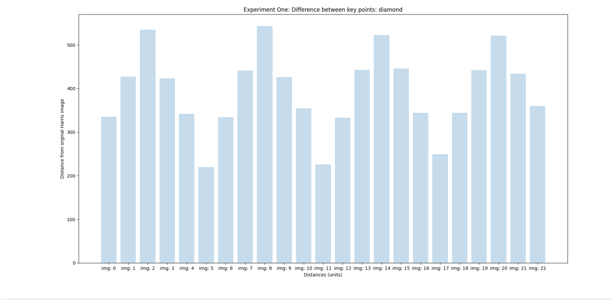

Fig. 1: Difference of keypoitns found relative to first image: Harris, Rotated, Diamond

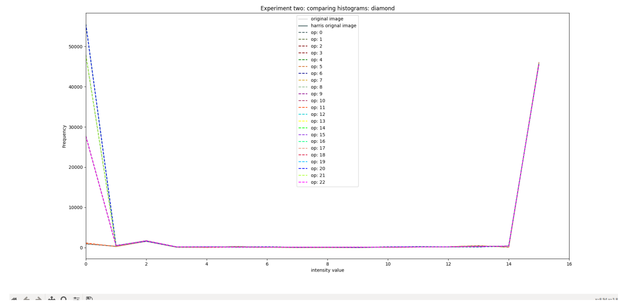

Fig. 2: Comparison of histograms for each transformed image: Harris, Roateted, Diamond

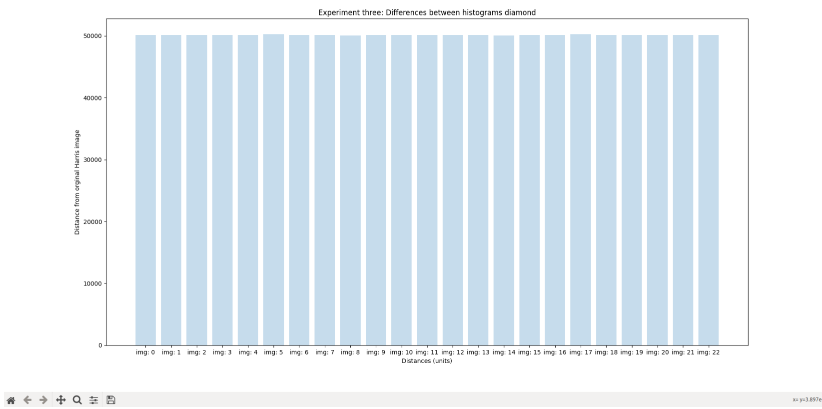

Fig. 3: Difference of distances between histograms relative to orignal: Harris, Rotated, Diamond

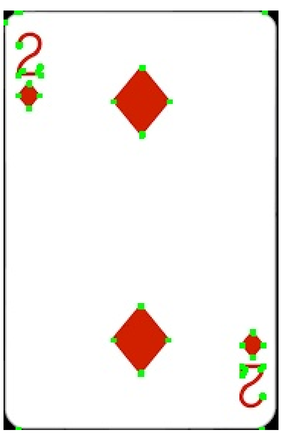

Fig. 4: Diamond harris rotated orignal produced image

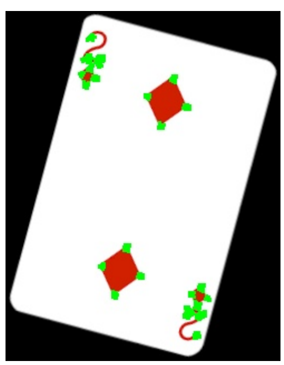

Fig. 5: Diamond harris image 1 rotated produced image

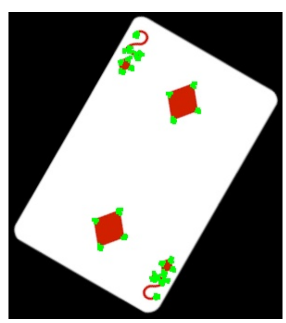

Fig. 6: Diamond harris image 2 rotated produced image

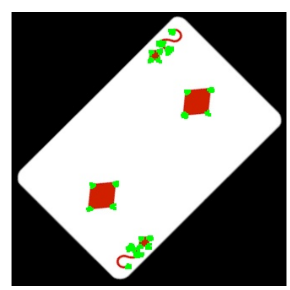

Fig. 7: Diamond harris image 3 rotated produced image

A set of 24 experimental images were produced. Only four displayed in report as displaying all 24 doesn't add great benefit to the reader. To re-view all 24 experimental images please implement the algorithms found in the appendixes.

- Calculate the histograms

Before the linear binary pattern can be performed on an image the image the image has to be converted into a gray-scale image [5]. The main reason for this step is because all the color channels are unnecessary because LBP is mainly comparing the intensity of pixels relative to a neighbourhood [5], and including all the color channels will be an extra processing over-head and algorithm complexity as it contains

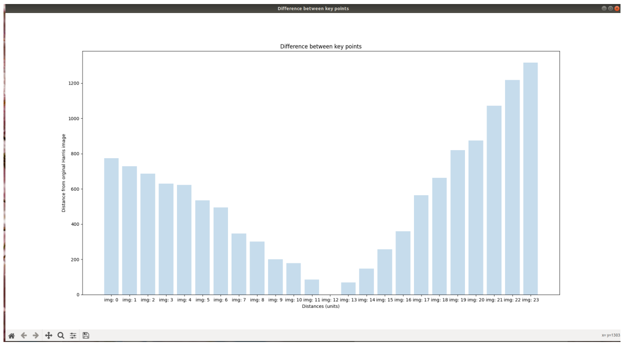

Fig. 8: Difference of keypoitns found relative to first image: Harris, Scaled, Diamond

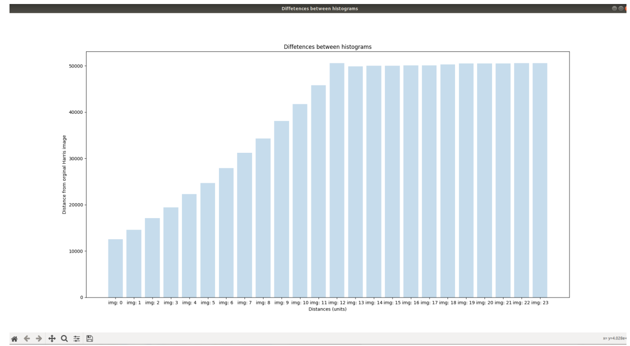

Fig. 9: Difference between the distances of histograms: Harris, scaled, Diamond

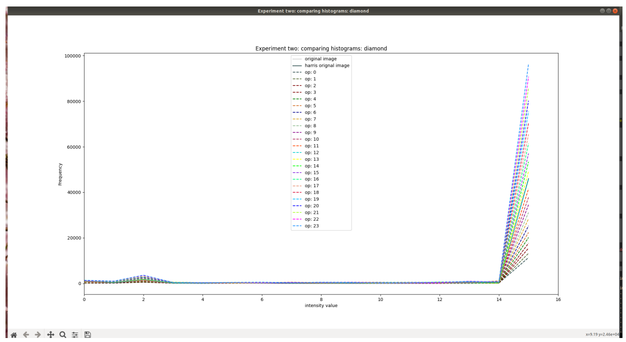

Fig. 10: All produced histgrams: Harris, Scaled, Diamond

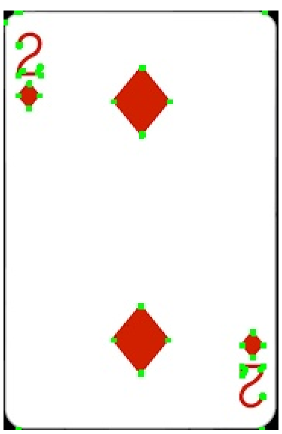

Fig. 11: Harris scaled orignal produced image: Harris, Scaled, Diamond

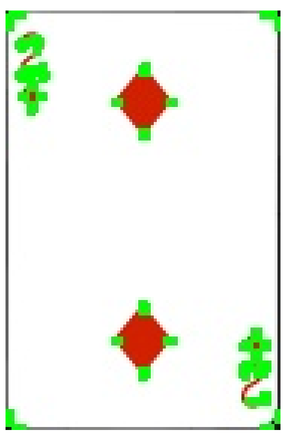

Fig. 12: Diamond arris image 1 scaled produced image

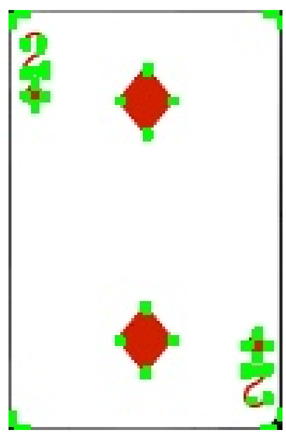

Fig. 13: Diamond harris image 2 scaled produced image

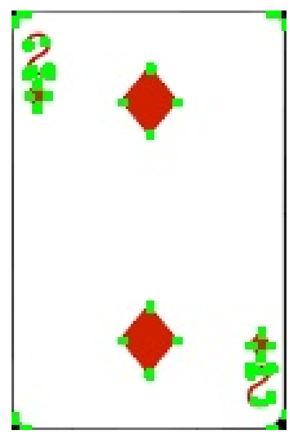

Fig. 14: Diamond harris image 3 scaled produced image

A set of 24 experimental images were produced. Only four displayed in report as displaying all 24 doesn't add great benefit to the reader. To re-view all 24 experimental images please implement the algorithms found in the appendixes.

two extra dimensions in image processing [7].

The next step in the algorithm is to calculate the LBP for each neighbouring cell. The algorithm will initialise a function window of nxn size which will scan through the image, and at each group of cells the linear binary pattern will compare the pixel value with the surrounding pixel values [5]. The algorithm will start at any set point inside the function window and label the pixels in a clock-wise or a counter clockwise direction [5] [8]. If the current pixel is greater than the middle pixel the pixel would be labelled 1, otherwise it will be labelled 0 the labelling is also applicable in the other way thus for greater pixels label the pixel 1 otherwise, label the pixel 0. [8] [5].

Thereafter the algorithm will determine decimal represen-

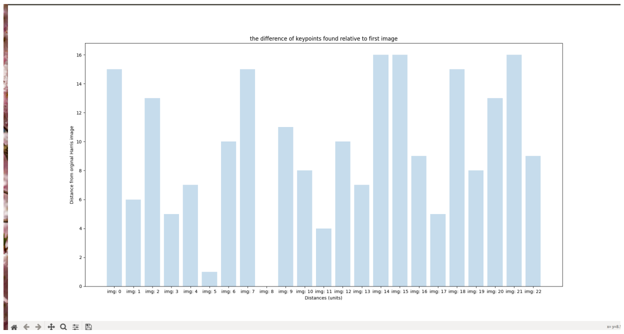

Fig. 15: Difference of keypoitns found relative to first image: SIFT, Rotated, Diamond

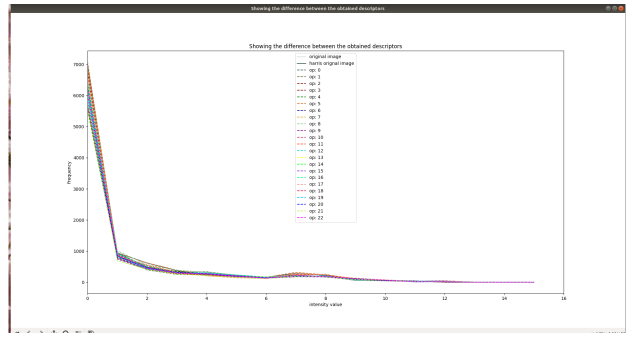

Fig. 16: Comparison of histograms for each transformed image: SIFT, Roateted, Diamond

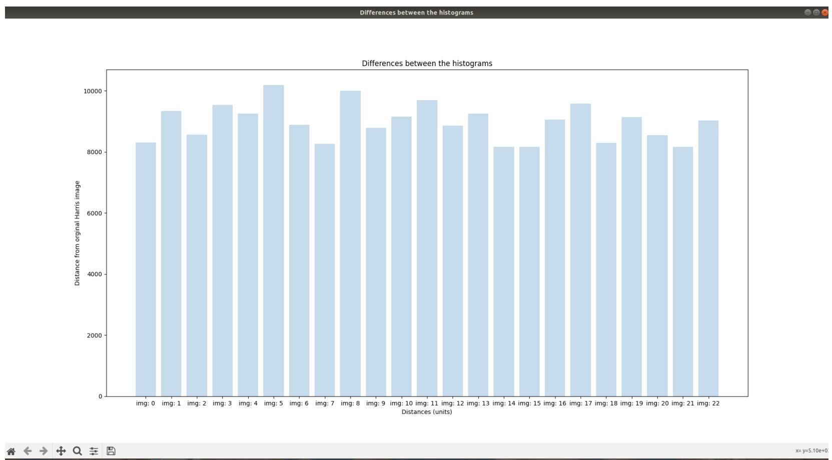

Fig. 17: Difference of distances between histograms relative to orignal: SIFT, Rotated, Diamond

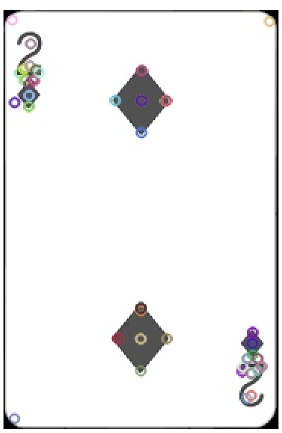

Fig. 18: Diamond SIFT rotated orignal produced image

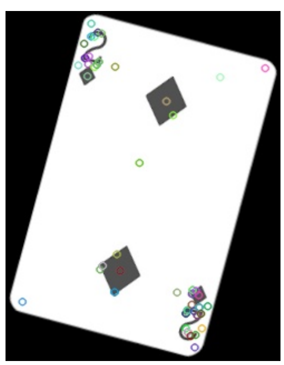

Fig. 19: Diamond SIFT image 1 rotated produced image

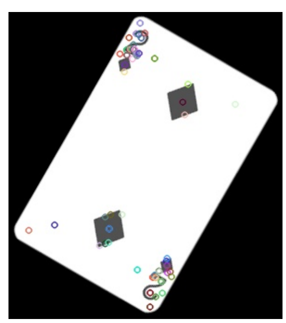

Fig. 20: Diamond SIFT image 2 rotated produced image

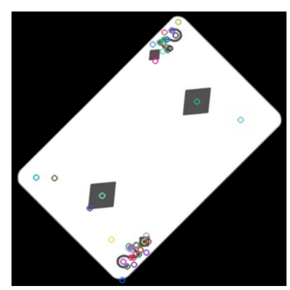

Fig. 21: Diamond SIFT image 3 rotated produced image

A set of 24 experimental images were produced. Only four displayed in report as displaying all 24 doesn't add great benefit to the reader. To re-view all 24 experimental images please implement the algorithms found in the appendixes.

tation of the produced binary pattern for the centre pixel. The algorithm will start from the same position as the labelling had began, transverse the window in the same direction as the labelling had done [5]. This number is stored in an output LBP matrix [5]. This process is repeated indefinitely till all the pixels of the original image are mapped to an output LBP matrix [5]. Consequently, the process will form a replica image of the original image although, this outputted matrix will be

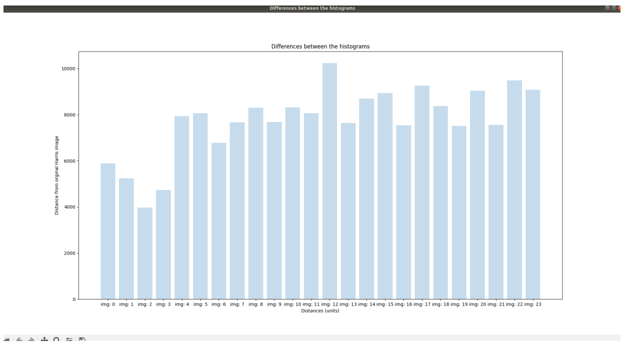

Fig. 22: Difference of keypoitns found relative to first image: SIFT, Scaled, Diamond

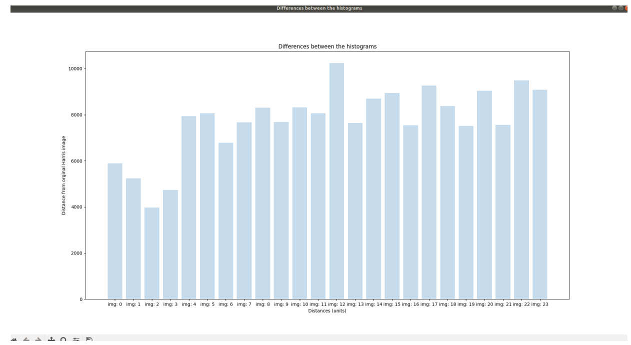

Fig. 23: Difference between the distances of histograms

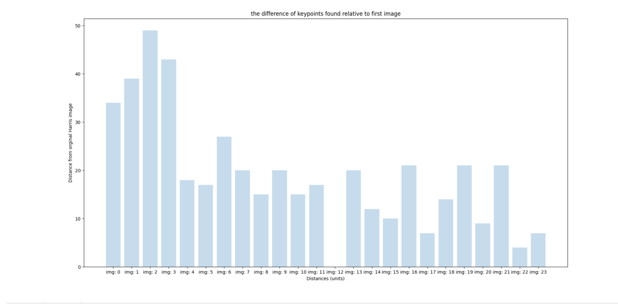

Fig. 24: All produced histgrams: SIFT, Scaled, Diamond

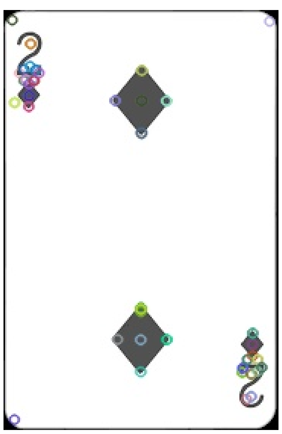

Fig. 25: Harris scaled orignal produced image: Harris, Scaled, Diamond

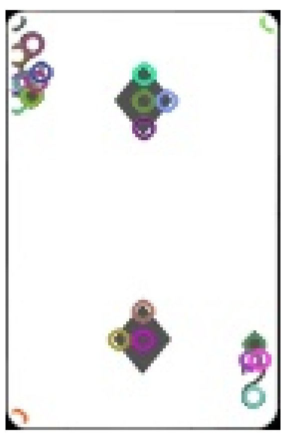

Fig. 26: Diamond SIFT image 1 scaled produced image

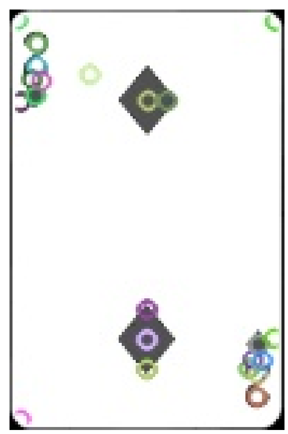

Fig. 27: Diamond SIFT image 2 scaled produced image

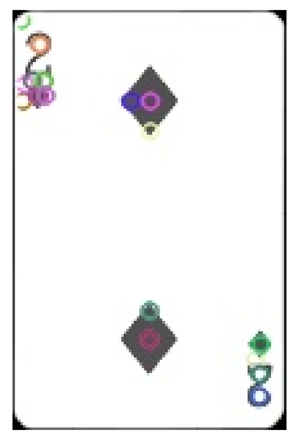

Fig. 28: Diamond SIFT image 3 scaled produced image

A set of 24 experimental images were produced. Only four displayed in report as displaying all 24 doesn't add great benefit to the reader. To re-view all 24 experimental images please implement the algorithms found in the appendixes.

more descriptive about the textures found in the image [5].

Finally, the computation of histograms is done to the output matrix array in relation to the size of the window function hence if a 3 by 3 neighbourhood with 8 surrounding pixels this will yield 256 patterns ($2^8 = 256$) thus, being able to form a histogram with a minimum value of 0, and a maximum value of 255 [5]. Therefore, forming a final feature vector of the image [5]. This is the main steps of the LBP algorithm, although they exists other implementations of the LBP algorithm such as the one proposed by Ojala elt al [5]

IV. HERE

A. Benefits of LBP

- The ability to capture fine grained details in an image [5]. Thus making it suitable to accurately identifying the

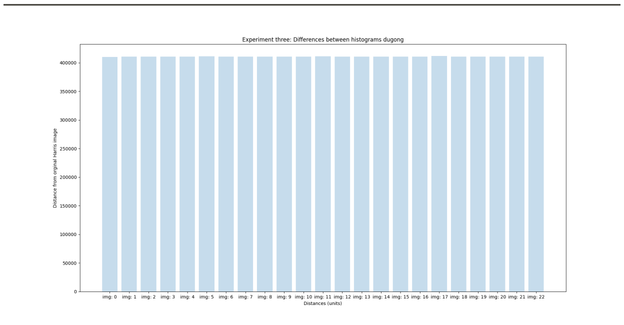

Fig. 29: Difference of keypoints found relative to first image: Harris, Rotated, Diamond

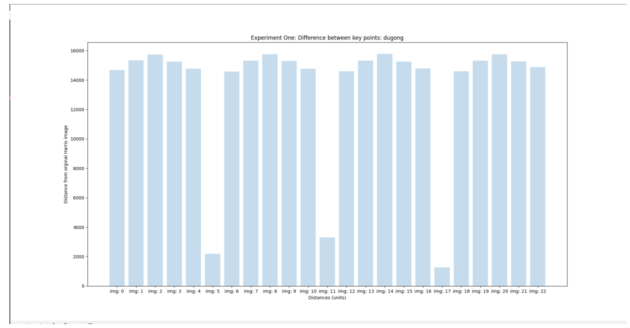

Fig. 30: Comparison of histograms for each transformed image: Harris, Roateted, Diamond

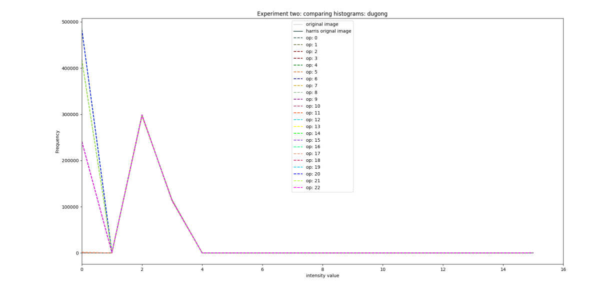

Fig. 31: Difference of distances between histograms relative to orignal: Harris, Rotated, Diamond

Fig. 32: Diamond harris rotated orignal produced image

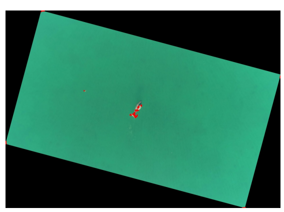

Fig. 33: Diamond harris image 1 rotated produced image

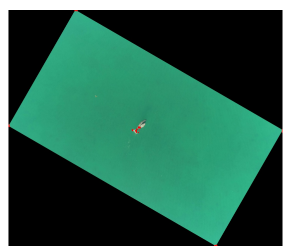

Fig. 34: Diamond harris image 2 rotated produced image

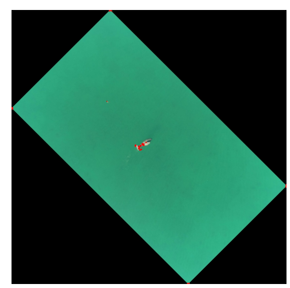

Fig. 35: Diamond harris image 3 rotated produced image

A set of 24 experimental images were produced. Only four displayed in report as displaying all 24 doesn't add great benefit to the reader. To re-view all 24 experimental images please implement the algorithms found in the appendixes.

location of a cell [6].

- Computationally fast, as it's operations are done on a gray-scale image [7] [8].
- few algorithms which gives you information about the texture of an image, as most algorithms give your information about corners and edges [8].

B. *Disadvantages of LBP*

- cannot capture fine grained details at varying scales, this is only possible for a 3 by 3 function window [5]. Although, they has been an LBP implementation algorithm proposed by Ojala et al which overcomes this problem [5].
- For the case of cell identification, it greatly identifies many regions incorrectly relative to other feature detection algorithms [6].

C. *part i: the main steps in histogram of gradients*

D. *part i: the main steps in SIFT*

E. *part ii*

V. Task Three

A. *Results for Diamond playing card*

Don't forget to include the areas found by each object extracted

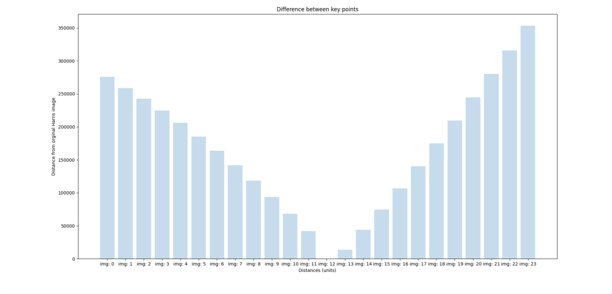

Fig. 36: Difference of keypoitns found relative to first image: Harris, Rotated, Diamond

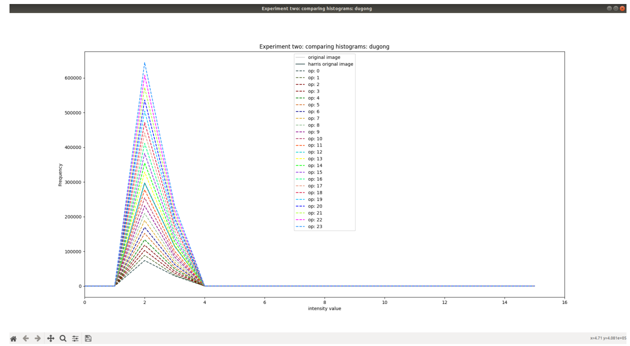

Fig. 37: Comparison of histograms for each transformed image: Harris, Roateted, Diamond

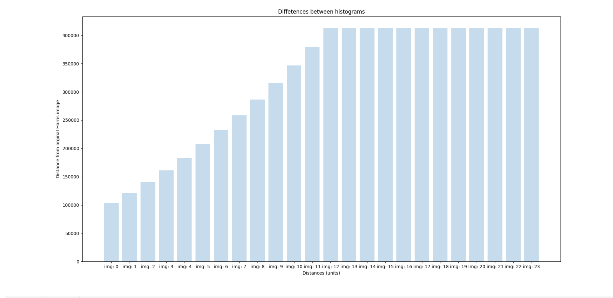

Fig. 38: Difference of keypoitns found relative to first image: Harris, Rotated, Diamond

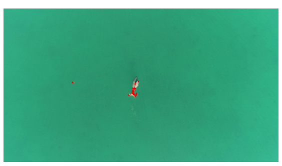

Fig. 39: Diamond harris scaled orignal produced image

Fig. 40: Diamond harris image 1 scaled produced image

Fig. 41: Diamond harris image 2 scaled produced image

Fig. 42: Diamond harris image 3 scaled produced image

A set of 24 experimental images were produced. Only four displayed in report as displaying all 24 doesn't add great benefit to the reader. To re-view all 24 experimental images please implement the algorithms found in the appendixes.

Don't forget to insert the tables for the area found for each object in here as well

B. Results for Dugong

don't forget to insert the results for the area found by the algorithm

don't forget to insert to insert the resutls for the area found by the algorithm here

VI. Task Four: Image Segmentation with K-means

A. Diamond: Image Segmentation: with HSV color space

B. Diamond: Image Segmentation: with BGR color space

C. Diamond: Image Segmentation: with contouring

D. Dugong: Image Segmentation: with HSV color space

E. Dugong: Image Segmentation: with BGR color space

F. Dugong: Image Segmentation: with contouring

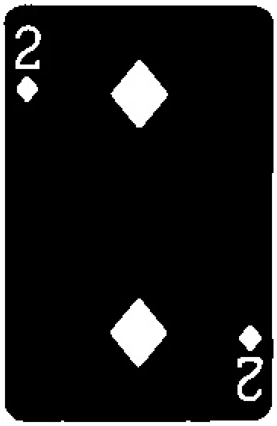

Fig. 43: Diamond: Seperated fore-ground and background

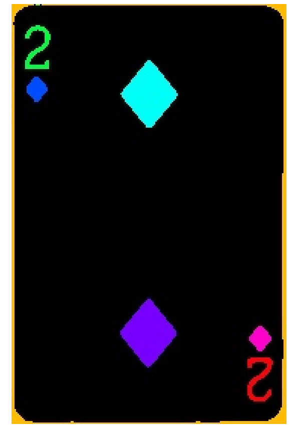

Fig. 44: Diamond: the labelling of each object found in the image

Fig. 45: Dugong: Seperated fore-ground and background

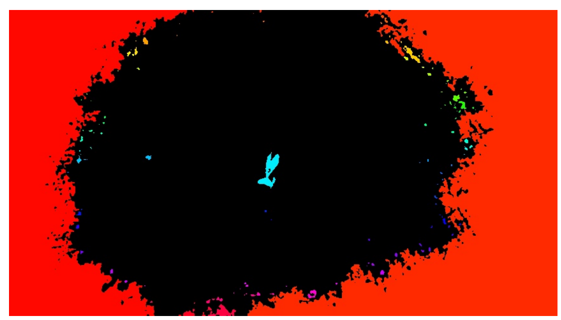

Fig. 46: Dugong: the labelling of each object found in the image

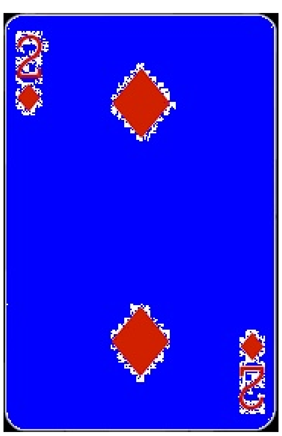

Fig. 47: Diamond: 1st cluster found by k-means

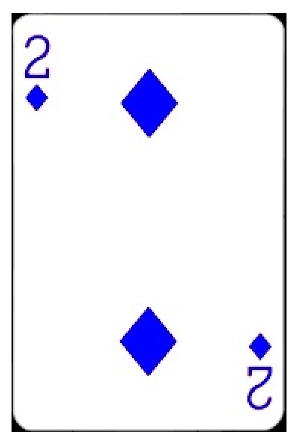

Fig. 48: Diamond: 2nd cluster found by k-means

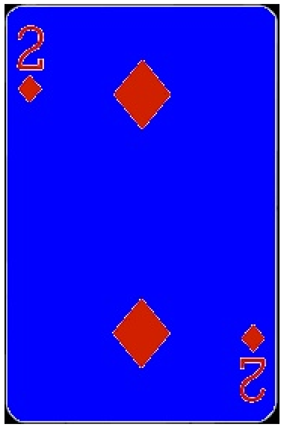

Fig. 49: Cluster found by the BGR color space

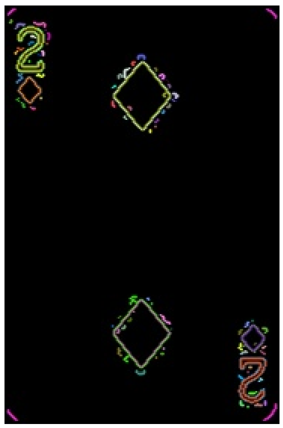

Fig. 50: Image segmentation done by contouring

Fig. 51: Dugong: 1st cluster found by k-means

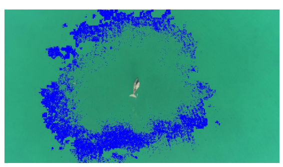

Fig. 52: Dugong: 2nd cluster found by k-means

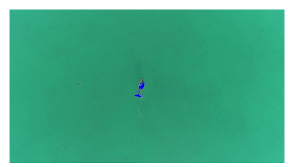

Fig. 53: Dugong: 3rd cluster found by k-means

Fig. 54: Dugong: 1st cluster found by k-means

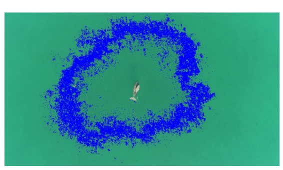

Fig. 55: Dugong: 2nd cluster found by k-means

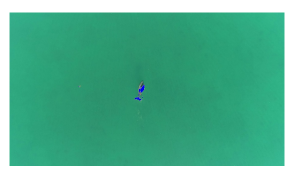

Fig. 56: Dugong: 3rd cluster found by k-means

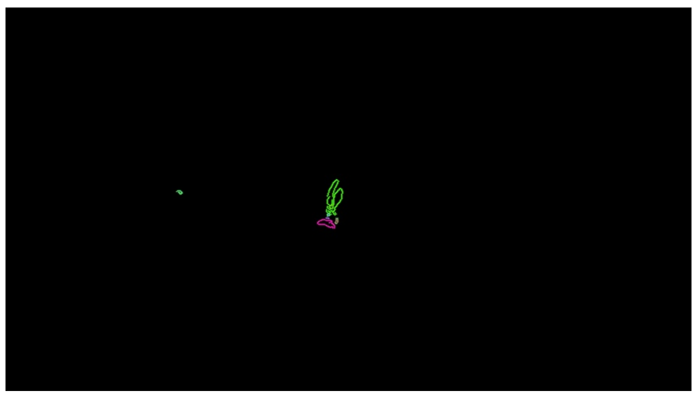
Fig. 57: Dugong: Image segmentation done by contouring

REFERENCES

[1] Dr. S. An. 2020. Machine Perception Lecture 03 [PowerPoint slides] Available:https://learn-ap-southeast-2-prod-fleet01-xythos.s3.ap-southeast-2.amazonaws.com/5dc3e34515a0e/4348643?response-cache-control=private%2C%20max-age%3D21600&response-content-disposition=inline%3B%20filename%2A%3DUTF-8%27%27lecture03_feature_detection.pdf&response-content-type=application%2Fpdf&X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Date=20200927T060000Z&X-Amz-SignedHeaders=host&X-Amz-Expires=21600&X-Amz-Credential=AKIAYDKQORRYZBCCQFY5%2F20200927%2Fap-southeast-2%2Fs3%2Faws4_request&X-Amz-Signature=bcf107d5e76759efe45687b516ac725158189de0af558d0a7a798df1723f4299

[2] A. Mordvinstev and K. Abid. ”Harris Corner detection”. OpenCV-Python Tutorials. https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_feature2d/py_features_harris/py_feature (retrieved Sept. 27, 2020).

[3] OpenCV. ”Introduction to SIFT (Scale-Invariant Feature Transform)”. OpenCV-Open source Computer vision. https://docs.opencv.org/3.4/da/df5/tutorial_py_sift_intro.html

[4] C. Harris and M. Stephens. 1988. A Combined Corner And Edge Detector. Plessey Reasearch Roke Manor, UK. [Online]. Available: https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.434.4816&rep=rep1&type=p

[5] A. Rosebrock. December, 7, 2015. ”Local Binary Patterns with Python & OpenCV”. pyimagesearch. https://www.pyimagesearch.com/2015/12/07/local-binary-patterns-with-python-opencv/

[6] S. Ozturk, and B.Akdemir. ”Comparison of HOG, MSER, SIFT, FAST, LBP and CANNY features for cell detection in histopathological images ” in Helix the scientfic Explorer. Feb 2018. [Online]. Available: http://helix.dnares.in/wp-content/uploads/2018/05/3321-3325.pdf

[7] Kanan C, Cottrell GW (2012) Color-to-Grayscale: Does the Method Matter in Image Recognition? PLoS ONE 7(1): e29740. doi:10.1371/journal.pone.0029740

[8] Dr. S. An. 2020. Machine Perception Lecture 04 [PowerPoint slides] Available: https://learn-ap-southeast-2-prod-fleet01-xythos.s3.ap-southeast-2.amazonaws.com/5dc3e34515a0e/6301343?response-cache-control=private%2C%20max-age%3D21600&response-content-disposition=inline%3B%20filename%2A%3DUTF-8%27%27lecture04_feature_extraction_2019%25281%2529.pdf&response-content-type=application%2Fpdf&X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Date=20201005T060000Z&X-Amz-SignedHeaders=host&X-Amz-Expires=21600&X-Amz-Credential=AKIAYDKQORRYZBCCQFY5%2F20201005%2Fap-southeast-2%2Fs3%2Faws4_request&X-Amz-Signature=c3d82f1e880c0787db8eb858336ae4e82df4d25ad29ea5a9563494a6f57cc6d4

[9] C. H. Chan. ”Multi-scale Local Binary Pattern Histogram for Face Recognition”. The Universtiy of Surrey. Surrey, Guildford, U.K. gu27xh. September 2008. [Online]. Available: https://www.researchgate.net/profile/Chi-Ho_Chan/publication/224936218_Multi-scale_Local_Binary_Pattern_Histograms_for_Face_Recognition/links/00b495225c9818 scale-Local-Binary-Pattern-Histograms-for-Face-Recognition.pdf

[10] . Z. Tang, Y. SU, M. J. Er, F. Qi, L. Zhang, and J. Zhou. ”A local Binary pattern based texture descriptors for classification of tea leaves.” in Neurocomputing. December 2014. [Online]. Available: https://reader.elsevier.com/reader/sd/pii/S0925231215006475?token=53A42C9FF9C438

APPENDIX

APPENDIX A
TASK 1 CODE PRINT OUT

APPENDIX B
TASK 2 CODE PRINT OUT