Hierarchically Grouped 2D Local Features Applied to Edge Contour Localisation

Yuan Liu

Submitted in fulfilment of the requirements for the degree of
Doctor of Philosophy

School of Computing Science

College of Science and Engineering
University of Glasgow

May 2016

Abstract

One of the most significant research topics in computer vision is object detection. Most of the reported object detection results localise the detected object within a bounding box, but do not explicitly label the edge contours of the object. Since object contours provide a fundamental diagnostic of object shape, some researchers have initiated work on linear contour feature representations for object detection and localisation. However, linear contour feature-based localisation is highly dependent on the performance of linear contour detection within natural images, and this can be perturbed significantly by a cluttered background.

In addition, the conventional approach to achieving rotation-invariant features is to rotate the feature receptive field to align with the local dominant orientation before computing the feature representation. Grid resampling after rotation adds extra computational cost and increases the total time consumption for computing the feature descriptor. Though it is not an expensive process if using current computers, it is appreciated that if each step of the implementation is faster to compute especially when the number of local features is increasing and the application is implemented on resource limited ”smart devices”, such as mobile phones, in real-time.

Motivated by the above issues, a 2D object localisation system is proposed in this thesis that matches features of edge contour points, which is an alternative method that takes advantage of the shape information for object localisation. This is inspired by edge contour points comprising the basic components of shape contours. In addition, edge point detection is usually simpler to achieve than linear edge contour detection. Therefore, the proposed localisation system could avoid the need for linear contour detection and reduce the pathological disruption from the image background. Moreover, since natural images usually comprise many more edge contour points than interest points (i.e. corner points), we also propose new methods to generate rotation-invariant local feature descriptors without pre-rotating the feature receptive field to improve the computational efficiency of the whole system.

In detail, the 2D object localisation system is achieved by matching edge contour points features in a constrained search area based on the initial pose-estimate produced by a prior object detection process. The local feature descriptor obtains rotation invariance by making use of rotational symmetry of the hexagonal structure. Therefore, a set of local feature descriptors is proposed based on the hierarchically hexagonal grouping structure. Ultimately, the 2D object localisation system achieves a very promising performance based on matching the proposed features of edge contour points with the mean correct labelling rate of the edge contour points 0.8654 and the mean false labelling rate 0.0314 applied on the data from Amsterdam Library of Object Images (ALOI). Furthermore, the proposed descriptors are evaluated by comparing to the state-of-the-art descriptors and achieve competitive performances in terms of pose estimate with around half-pixel pose error.

Acknowledgements

Firstly, I would like to give deep thanks to my principal supervisor Dr. J. Paul Siebert. He guided me into the computer vision world, and provided endless patience and support to help me go through my PhD study.

Secondly, I would like to express my gratitude to Dr. William Paul Cockshott. As my second supervisor, he always gave me suggestions whenever I needed.

Thirdly, thanks to all the members of Computer Vision and Graphics group. Because of your company in the office, I never felt lonely during the research time. I want to give my special thanks to Dr. Gerardo Aragon-Camarasa for his great help in my research.

Last but not least, I would like to thank my husband, my parents and my elder sister. I can always feel their endless love and support, even when they are not around me. I will try my best in the rest of my life to love them.

Table of Contents

1 Introduction **1**

1.1 Background 1

1.2 Motivation and Objective 3

1.3 Overview of the General Approach 4

1.4 Major Contributions 6

1.5 Thesis Organisation 6

1.6 List of Publications 6

2 Background and Literature Review **8**

2.1 Object Detection and Localisation 8

2.1.1 Object Detection 8

2.1.2 Object Localisation 11

2.2 Feature Extraction 14

2.2.1 Feature Representation 15

2.2.2 Feature Hierarchy 21

2.2.3 Orientation Assignment of the Local Features 23

2.3 Image Processing on Hexagonal Grid 25

2.4 Summary and Discussion 27

3 Initial Investigation of Hierarchical Feature Representation Based on Hexagonal Grouping Structure **29**

3.1 Motivation and Objective 29

3.2 Hierarchical Feature Extraction 30

3.2.1 Feature Detection 30

3.2.2 Feature Representation 31

3.3 Evaluation 35

3.4 Summary and Discussion 40

4 2D Object Localisation Based on Matching HexSHoG Descriptors 44

4.1 Motivation and Objective 44

4.2 Rotation-Invariant Feature Extraction 45

4.2.1 Rotation-Invariant Hexagonal Structure 45

4.2.2 HexSHoG Generation 46

4.2.3 Rotation-Invariant Performance of HexSHoG 49

4.3 2D Object Localisation 50

4.3.1 Detection and Pose Estimate 51

4.3.2 Edge Labelling for Object Localisation 53

4.4 Performance Evaluation 53

4.4.1 Pose Refinement Performance 55

4.4.2 Edge Labelling Performance 57

4.5 Summary and Discussion 58

5 2D Object Localisation Based on Matching HexBinary Descriptors 62

5.1 Motivation and Objective 62

5.2 HexBinary Feature Extraction 64

5.2.1 Orientation Assignment 64

5.2.2 First Order HexBinary 65

5.2.3 Second Order HexBinary 66

5.3 Application to 2D Object Localisation 67

5.3.1 Pose Refinement 67

5.3.2 Edge Labelling for Object Localisation 67

5.4 Performance Evaluation 69

5.4.1 Rotation-Invariant Performance of HexBinary 69

5.4.2 Pose Refinement Performance 70

5.4.3 Edge Contour Localisation Performance 74

5.5 Summary and Discussion 75

6 Learning Better HBDs: Hexagon-Based Binary Descriptors **78**

6.1 Motivation and Objective 78

6.2 Approach 79

6.2.1 Sampling Structure 80

6.2.2 Descriptor Generation 83

6.3 Performance of Rotation Invariance 84

6.4 Parameter Learning of HBD 86

6.4.1 Dataset 86

6.4.2 Parameter Learning 87

6.4.3 Performance Evaluation 89

6.5 Application to Pose Estimate 90

6.6 Summary and Discussion 92

7 Conclusion and Future Work **104**

7.1 Thesis Objectives 104

7.2 Contributions 105

7.3 Future Work 105

7.3.1 Out-of-Plane Application 105

7.3.2 2D Object Localisation Using Features of Pure Edge Contours . . . 106

7.3.3 Develop A Semantic Contour Detector 107

7.3.4 Develop A Hierarchically Grouped Descriptor Based on Pure Edge Contours 110

7.3.5 Collect A Data Set of Natural Images with Scale Changes 110

7.3.6 Reduce the Feature Dimension 111

7.3.7 Develop A Ternary Descriptor 111

Bibliography **112**

List of Tables

3.1 Mean value over the rotation degrees of both mean matching score and mean correct matching rate for each level of the hierarchical descriptor based on *Data1*. 39

3.2 Standard deviation (SD) over the rotation degrees of both mean matching score and mean correct matching rate for each level of the hierarchical descriptor based on *Data1*. 39

3.3 Mean value over the rotation degrees of both mean matching score and mean correct matching rate for each level of the hierarchical descriptor based on *Data2*. 39

3.4 Standard deviation (SD) over the rotation degrees of both mean matching score and mean correct matching rate for each level of the hierarchical descriptor based on *Data2*. 40

3.5 Computing time for a single descriptor 40

4.1 Pose estimate refinement performance as a function of pixel search range and the initial pose estimate performance by SIFT 57

5.1 Mean error of single edgel position for each level of descriptor and the corresponding reduced value Δ from the initial mean error (G_FHexB means the first order HexBinary descriptor generated from Gaussian filtered images; L_SHexB means the second order HexBinary descriptor generated from LoG filtered images). 72

5.2 Standard deviation of single edgel position error for each level descriptor and the corresponding reduced value Δ from the initial standard deviation. Abbreviations of the descriptors are defined as the same as in Table 5.1. . . 73

5.3 Number of improved images after pose refinement. Abbreviations of the descriptors are defined as the same as in Table 5.1. 73

5.4 Computing time for a single descriptor 74

5.5 The descriptor dimension length 74

6.1 The descriptor length (L) and the number of sampling fields (N) 84

6.2 The learning range of different parameters 88

6.3 The learned values of different parameters for constructing the hexagonal structure. 89

6.4 Numeric distribution of images detected within a given pose error range (pixels) and the corresponding error ranges. 91

6.5 Number of images successfully detected with a pose error of less than 5 pixels, and their corresponding pose error Mean (*Mean*) and Standard Deviation (*SD*) in pixels. 91

List of Figures

1.1 Two different hexagonal grouping structures for generating hierarchical feature descriptors. For each row, from left to right, the hexagonal structure changes from the first level to the third level of the hierarchy. 5

1.2 Overview of the 2D object localisation system 5

2.1 Pedestrian detection result: the red rectangle indicates a pedestrian detected in the region, and the green rectangle shows that no pedestrian appears in the covered area. 9

2.2 An image patch is represented by a histogram of local oriented gradients. . 16

2.3 Sampling grids: the left is the Cartesian sampling grid in SIFT, redrawn from [Mikolajczyk and Schmid, 2005], the right is the log-polar sampling grid in GLOH, redrawn from [Mikolajczyk and Schmid, 2005]. 16

2.4 DAISY sampling structure: each circle indicates a Gaussian smoothed region, and the centres of them are the sampling positions for computing DAISY, redrawn from [Tola et al., 2008]. 17

2.5 Processing stages for the generic descriptor algorithm, redrawn from [Winder and Brown, 2007]. 18

2.6 Left : BRISK sampling pattern with $N = 60$ points, reproduced from [Leutenegger et al., 2011]. Right: FREAK sampling pattern which is similar to the retinal ganglion cells distribution, reproduced from [Alahi et al., 2012] . . . 19

2.7 A generic pattern shared by different objects. 20

2.8 Schematic illustration of a semantic hierarchy, represented from [Epshtein and Ullman, 2007]. 21

2.9 Hierarchical compositional representation of object categories, represented from [Leonardis and Fidler, 2011]. 22

2.10 SURF orientation assignment [Bay et al., 2006]. The vector represents the wavelet response in x and y directions at each pixel in the neighbourhood. . 24

2.11 In the square grid, the pixel has two types of distance from the neighbours to it, while in the hexagonal grid, all the pixel neighbours have the same distance to the pixel. 26

3.1 (a) Overlapped sampling of local patches; (b) A single HoG covers the patch centred at the red key-point, where the 9 blue points are the neighbours with 3-pixel chessboard distance to the red key-point, and the circle is the Gaussian kernel size for extracting HoG descriptor in this patch. 31

3.2 Hierarchical structure of the rectangularly grouped feature descriptor RecHoG. The 9 red points are the sampling positions of the rectangular grouping. Each circle represents a gaussian weighted patch represented by HoG descriptor. 32

3.3 Hierarchical structure of the hexagonally grouped feature descriptor HexHoG. The 7 red points are the sampling positions of the hexagonal grouping. Each circle represents a gaussian weighted patch represented by HoG descriptor. 34

3.4 Used data for evaluation: The first row shows the 5 reference images in *Date1*. The second row shows the 5 reference images in *Date2*, and the third row illustrates the test images containing the corresponding objects with 10 degrees physical rotation in the real world. 36

3.5 The first row shows the results of mean matching score (rotation invariance) for different levels of RecHoG and HexHoG as a function of rotation degree based on *Data1*. The second row shows the results of mean correct matching rate (distinctiveness) for different levels of RecHoG and HexHoG as a function of rotation degree based on *Data1*. Each subfigure illustrates the comparison among four different levels of the hierarchical descriptor. . . . 37

3.6 The first row shows the results of mean matching score (rotation invariance) for different levels of RecHoG and HexHoG as a function of rotation degree based on *Data2*. The second row shows the results of mean correct matching rate (distinctiveness) for different levels of RecHoG and HexHoG as a function of rotation degree based on *Data2*. Each subfigure illustrates the comparison among four different levels of the hierarchical descriptor. . . . 38

3.7 The comparison between RecHoG and HexHoG in each single level based on *Data1*. The first four subfigures illustrate results of the mean matching rate (rotation invariance). The last four subfigures show results of the mean correct matching rate (distinctiveness). 42

3.8 The comparison between RecHoG and HexHoG in each single level based on *Data2*. The first four subfigures illustrate results of the mean matching rate (rotation invariance). The last four subfigures show results of the mean correct matching rate (distinctiveness). 43

4.1 The red arrow is the dominant orientation of the area covered by the red dot circle. In (a), the figure shows the conventional method to achieve rotation invariance of the local descriptor, which is achieved by first rotating the local area to make the dominant orientation aligned with the X-coordinate then sampling two of the hexagonal vertexes along the horizontal line. In (b), there is no rotation of the local area, but the hexagonal vertexes are sampled directly according to the dominant orientation with two vertexes aligning with the dominant orientation. 46

4.2 Local patch represented by HoG and SHoG. The upper histogram is the standard HoG, and the lower histogram is SHoG. SHoG is achieved by shifting the peak value of HoG to the head position of the histogram. 47

4.3 The left figure illustrates the second level structure of HexSHoG . The right figure shows the structure of the red dot covered region in the left figure. . . 48

4.4 The results of mean matching score (rotation invariance) as a function of rotation degree. The left figure shows the comparison among HoG, DHoG and SHoG. The right figure illustrates the comparison among four different levels of HexSHoG. SHoG denotes level 0 of HexSHoG. The scale of Y axis is different between the two figures. 50

4.6 Each test image is directly matched to the corresponding reference image for object detection and pose estimate. 51

4.5 The brief summary of the 2D object localisation system. 51

4.7 Pose refinement and edge contour labelling processes. The two graphs show the same test image with its edge contours labelled in blue, and the red edges are the projected edges of the reference object according to the estimated pose information. (a) is the local searching process for pose refinement after initial pose estimate; (b) is the local searching process for edge contour labelling after pose refinement. 52

4.8 Examples of the data used in the application experiment: The top row shows the reference images from ALOI. The middle row shows the corresponding in-plane rotated versions. The bottom row shows the test images generated by embedding the rotated images into different backgrounds. 54

4.9 Displacement error of local HexSHoG matching as a function of rotation degree. 55

4.10 Failed examples of detection by SIFT: The first column shows the reference objects. The remaining columns show the test objects with backgrounds. . . 56

4.11 The mean pose error results for the refined pose estimation are illustrated as a function of pixel search range. The initial pose error given by SIFT with no search process is also displayed in this figure for clear comparison. The error bar shows 95% confidence interval for the mean pose error. 56

4.12 Edge projection from the reference objects into the test images according to the initial and refined pose estimate, shown in left and right columns, respectively. The first two rows show the examples with improvement after pose estimate refinement. The last two rows show the examples that failed to achieve improvement after pose estimate refinement. 60

4.13 Object edge contour labelling results: from the first column to the third column, edge labelling results by using *HexSHoG1*, *HexSHoG2*, *HexSHoG3*, are displayed respectively. 61

5.1 The hierarchically hexagonal structure for HexSHoG: (a) The first level sampling pattern of the hexagonal structure (the red arrow shows the dominant orientation of the red dotted region); (b) The second level sampling pattern of the hexagonal structure; (c) The third level sampling pattern of the hexagonal structure. Each black circle represents the Gaussian kernel size which could be freely parameterised. 63

5.2 The original first level sampling pattern of the hexagonal structure. Two of the hexagonal vertexes are sampled on the direction of X-coordinate. Each black circle represents the Gaussian kernel size. 65

5.3 The matching results for different HexBinary descriptors as a function of rotation degree. The difference between two binary descriptors is computed by Hamming Distance. 70

5.4 Pose refinement performance with 95% confidence interval for the mean pose error: (a) The level one performance of different hierarchical descriptors; (b) The level two performance of different hierarchical descriptors; (c) The level three performance of different hierarchical descriptors. 71

5.5 (a) ROC curve illustration. (b) Distribution of pixel-level localisation: each red dot represents an edge labelling ROC point, *RP* vs *RN*. 75

5.6 Pose estimate and edge contour localisation examples: The first column shows reference contour projection based on initial pose estimate by SIFT. The second column displays reference contour projection based on refined pose estimate by $G\_SHexBinary3$. The third column illustrates directly labelled edge contour points of the test object. 77

6.1 The first level hexagonal structure for generating HexBinary descriptors. The black circle G is the Gaussian window for smoothing, and the red edge r is the edge length of the basic hexagon. 79

6.2 (a) is the second level structure of generating HexBinary. (b) is the new proposed structure of the third level hexagonal descriptor. The red star point represents the key-point position where to sample the local descriptor. The arrow illustrates the dominant orientation of the local region around the key-point. The first level hexagonal grouping structure comprises the basic hexagon centred at the red $\star$, and the sampling positions for generating binary strings are taken from the hexagon centre and the vertexes. The second level hexagonal structure now covers more image area around the key-point since six more hexagon centres are sampled as shown in blue $+$, which are sampled at different positions in (a) and (b). For the third level descriptor, another 12 more hexagon centres as shown in yellow $\triangle$ in (b) are computed together with the previous 7 hexagon centres around the key-point, while in (a), 7 second level structures will be constructed centred on the red $\star$ and blue $+$. All the black points indicate the sampling positions according to the corresponding hexagonal centre positions. 80

6.3 The first level structure of the hexagon-based hierarchical descriptor. The arrow indicates the local dominant orientation. The two sampling vertexes of the hexagonal structure are aligned with the local dominant orientation. . 83

6.4 $RecognitionRates$ of different level comparisons among *SHexBinary*, *HexIDB* and *HexLDB*. The comparison of each single level is illustrated from (a) to (c), respectively. 85

6.5 Image sequences: each sequence has 6 images, and only the first and the last images are illustrated. From the second to the sixth image, the difficulty in matching to the first image increases progressively. 87

6.6 Pose estimate performance with error bars showing 95% confidence interval for the mean pose error based on the results given in Table 6.5. 92

6.7 $RecognitionRate$ performance of the third level descriptors: *HexIDB3* and *HexLDB3*, with the changes of the standard deviation of the Gaussian smoothing kernel. Each image pair has 11 colour bars to represent the correct matching rates with 11 different sigmas. The matching difficulty of the pairs is increasing from left to right of each figure. The left column shows the performances with *HexIDB3*. The right column shows the performances with *HexLDB3*. 94

6.8 $RecognitionRate$ performance of the third level descriptors: *HexIDB3* and *HexLDB3*, with the changes of the standard deviation of the Gaussian smoothing kernel. Each image pair has 11 colour bars to represent the correct matching rates with 11 different sigmas. The matching difficulty of the pairs is increasing from left to right of each figure. The left column shows the performances with *HexIDB3*. The right column shows the performances with *HexLDB3*. 95

6.9 $RecognitionRate$ performance with Gaussian sigma=3.4. For each image pair, the performances of 3 different levels of *HexIDB* and *HexLDB* descriptors are illustrated. 96

6.10 $RecognitionRate$ performance of the third level descriptors: *HexIDB3* and *HexLDB3*, with $\sigma = 3.4$ and the kernel size varying. Each image pair has 10 colour bars to represent the correct matching rates with 10 different kernel sizes. The matching difficulty of the pairs is increasing from left to right of each figure. The left column shows the performances with *HexIDB3*. The right column shows the performances with *HexLDB3*. 97

6.11 $RecognitionRate$ performance of the third level descriptors: *HexIDB3* and *HexLDB3*, with $\sigma = 3.4$ and the kernel size varying. Each image pair has 10 colour bars to represent the correct matching rates with 10 different kernel sizes. The matching difficulty of the pairs is increasing from left to right of each figure. The left column shows the performances with *HexIDB3*. The right column shows the performances with *HexLDB3*. 98

6.12 $RecognitionRate$ performance with Gaussian sigma=3.4, kernel size=17 $\times$ 17. For each image pair, the performances of 3 different levels of *HexIDB* and *HexLDB* descriptors are illustrated. 99

6.13 $RecognitionRate$ performance of the third level descriptors: *HexIDB3* and *HexLDB3*, with $\sigma = 3.4$ and the kernel size $= 17 \times 17$. There are 8 different colour bars representing the hexagon edge length changing in the range of [1,8]. The matching difficulty of the pairs is increasing from left to right of each figure. The left column shows the performances with *HexIDB3*. The right column shows the performances with *HexLDB3*. 100

6.14 $RecognitionRate$ performance of the third level descriptors: *HexIDB3* and *HexLDB3*, with $\sigma = 3.4$ and the kernel size $= 17 \times 17$. There are 8 different colour bars representing the hexagon edge length changing in the range of [1,8]. The matching difficulty of the pairs is increasing from left to right of each figure. The left column shows the performances with *HexIDB3*. The right column shows the performances with *HexLDB3*. 101

6.15 $RecognitionRate$ performance with Gaussian sigma=3.4, kernel size=17×17, hexagon edge length=3. For each image pair, the performances of 3 different levels of *HexIDB* and *HexLDB* descriptors are illustrated. 102

6.16 The $RecognitionRate$ performance for each image pair with utilising different local feature descriptors. 103

7.1 Edge contour localisation for the out-of-plane application: (a) a robot gripper under 3° of out-of-plane rotation ; (b) a robot gripper with 5° of out-of-plane rotation. 106

7.2 Compute a shape context: pick out all the edge points of a contour, and then for the edge point p on the contour, compute its relative distance and angle to all the other points to generate a histogram, which is defined as the shape context of point p. 107

7.3 An example of object localisation based on *shape context* feature matching: (a) is the reference image. (b) is the reference edge map projected into the test image coordinate according to the pose estimate result. (c) is the test image, (d) is the test edge map, where the red points are from (b). (e) and (f) are the labelling results shown in the test edge map and the test image, respectively, where the red points are the labelled edge points of the target object. 109

7.4 A simple example of edge contour grouping: the grouping of the red, blue, and green contours can make a more distinctive feature descriptor than any of these contours considered individually. 110

Chapter 1

Introduction

Detecting objects in unknown images while simultaneously localising the edge contours (both internal and bounding) defining these objects is a challenging task in computer vision. A straightforward method to localise the object is matching edge contour points of the object. Moreover, the features used to represent the edge contour points must be robust to geometric transformations, such as rotations. To these ends, the main work of this thesis is twofold: Firstly, an investigation into 2D object localisation based on matching edge contour points. Secondly, the exploration of hierarchical feature extraction based on hexagonal structures that have rotational symmetry.

1.1 Background

In the field of computer vision, interpretation of the environment by machine is arguably one of the most important tasks required to achieve fully autonomous systems. Intelligent machines are now required to be competent in visual interpretation tasks, such as object recognition, categorisation, object tracking or navigation through the environment. Though their vision systems are still far from being as competent as that of the human visual system, computer vision researchers have been contributing to the endeavour of devising and constructing artificial vision systems for many decades and have achieved significant progress.

One of the most significant tasks in computer vision is object detection. Based on matching distinctive features extracted from model images and unknown images, the object of interest will be defined as whether it appears in the unknown image. The preferred detection result is not only indicating the existence of the object, but also localising the edge contour positions of the detected object in the image. Therefore, the object edge contours need to be labelled to illustrate the existence and the pose of the object in the image. However, it is a challenging problem in computer vision. Many object detection results only indicate the object of interest in a bounding box without giving the localisation of the object boundary contours [Viola and Jones, 2004; Dollar et al., 2012; Shao et al., 2012]. Previously published research work on object detection with localisation has reported integrating together a number of different image processing techniques, such as contour detection, model learning [Ferrari et al., 2010], and segmentation [Gu et al., 2009]. These processes could greatly improve the system capability of dealing with intra-class variations and image transformations, and help to localise the object contours more accurately, though they are very expensive to compute in terms of time consumption.

Feature extraction is one of the essential steps for visual interpretation of the environment. It serves as a means of dimensionality reduction for complex data and is expected to represent the data well and facilitate the visual interpretation tasks listed above. Therefore, many research investigations have been conducted into different feature extraction methods. However, no single feature descriptor appears to be sufficiently generic to be applicable to all types of image. Accordingly, the feature extraction process must always be tailored appropriately to represent the critically distinctive properties of the image. Current research investigations have focussed on capturing and representing the appearance information and shape information exhibited by the object in an image and accordingly, these two dominant features have been explored widely.

Appearance information is generally sampled by means of a local image patch which can be centred at *interest points*, or extracted systematically at *fixed* locations on a grid, or even extracted at *random* locations. The feature of the appearance information is usually a vector descriptor characterising the statistical information of the region with certain robustness and distinctiveness. Many of the state-of-the-art local feature descriptors are computed based on the appearance information by distributing the oriented gradients in the local area into histograms, such as the work in [Lowe, 2004; Mikolajczyk and Schmid, 2005; Dalal and Triggs, 2005; Tola et al., 2008]. With the requirement of real-time application, binary strings are introduced to use the appearance intensity comparison to generate local features achieving faster computation and less storage requirement without decreasing the feature distinctiveness [Calonder et al., 2010; Leutenegger et al., 2011].

Shape, as a significant feature to distinguish different objects, has also been researched extensively [Yang et al., 2008]. It could be generally divided into two categories to learn the shape description: one is contour-based, and the other is region-based. However, many shape features are developed by learning from the pure shape models. When it comes to natural image processing, a well-performed edge contour detector is always needed to first outline the object shape in the image. Inevitably, the cluttered background information and noise will disturb the detection of the object shape structure.

1.2 Motivation and Objective

Localising the position of an object accurately implies knowing the location of the object's bounding contours. Matching shape contour features between the model and test images is a straightforward method to label the object contours. However, an object's contours could be distorted due to the interaction with edge points emanating from the background (or indeed become fused with background edges), when the object is embedded in a cluttered environment. Therefore, matching edge contour points rather than shape contours might be an alternative method to localise the object successfully.

Many successful feature extraction approaches are based on sampling *interest points*, such as corners, to allow features to be localised in two dimensions and also to reduce the computational burden of representing key local patterns that are diagnostic for the object being represented. A disadvantage of this approach however, is that many types of object contain few or even no high-contrast corners defining characteristic appearance information specific to the object or object class in question. Images of many man-made objects such as plates, cups, or types of tool, fall into this category. In addition, to localise the object based on matching edge contour points, there will necessarily be many more edges detected than corners, therefore the computational efficiency of the descriptor used to sample at edge locations must be minimised for the approach to be computationally tractable. As a consequence, developing a robust general purpose feature descriptor that can cope with these situations could be particularly significant in robot vision, or manufacturing context.

Accordingly, to advance the ability to represent the objects having few corners, this work seeks to devise feature descriptors capable of capturing appearance information localised at arbitrary edge contour pixels as opposed to only being located at high curvature edge contours, i.e., corners. Moreover, in order to maintain computational efficiency, we also explore rotationally symmetric descriptors sampled on the vertices and centre of a hexagon (somewhat similar to DAISY descriptor), as opposed to traditional rectilinearly sampled descriptors. The use of hexagonal grids has been reported to implement low-level image processing operations, such as edge detection [Vidya et al., 2009] and image alignment [Shima et al., 2010]. It has been demonstrated that, compared to the image processing on a square grid, processing on a hexagonal grid requires less computation and yields smaller quantisation error [Kamgar-Parsi et al., 1989]. Therefore, based on the related work of image processing on hexagonal grid and the intrinsic advantages of a hexagon with sampling efficiency and consistent connectivity, there would appear to be the potential to exploit the use of hexagonal structures when devising local feature descriptors which are both distinctive and computationally efficient.

Inspired by the hierarchical nature of the primate visual cortex, hierarchical object representations have been subject to deep investigation and have achieved impressive classification performance in recent decades [Perrett and Oram, 1993; Riesenhuber and Poggio, 1999; Serre et al., 2005; Bo et al., 2011]. A hierarchical structure could better capture the diagnostic features of an object by representing the relatively larger parts of the object, which are less likely to appear in other objects, and therefore be of more diagnostic value. A hierarchical structure might also capture the spatial relationships between local, smaller, neighbouring parts, which potentially comprise generic patterns appearing in many different objects and therefore afford little diagnostic value. Representing an object by means of a hierarchically grouped local feature mechanism allows the region size supporting the extracted feature to be determined dynamically, which might potentially achieve better feature distinctiveness than hand-crafted features extracted from regions with fixed size.

To these ends, a 2D object localisation system based on matching edge contour points is investigated in this thesis, motivated by the research of detecting and localising objects through learning shape features. In addition, driven by the potential benefits that hexagonal structure and hierarchical mechanism might potentially bring, this thesis also focuses on investigating new local feature descriptors employing a hexagonal structure and a hierarchical grouping mechanism. The hypothesis that the proposed descriptors could afford improved stability and geometric invariance in visual processing is evaluated in the proposed 2D object localisation system.

1.3 Overview of the General Approach

In this thesis, two different hexagonal grouping structures are introduced to generate the feature hierarchy with up to 3 levels, and they are illustrated in Figure 1.1. The main reason for adopting the hexagonal structure is to take advantage of its sampling efficiency and consistent connectivity. The symmetry of a hexagon in three directions can afford a better opportunity to extract rotation-invariant features than that of a rectangle, which can only afford symmetry in two directions. Consequently, a set of hierarchical feature descriptors is proposed in this thesis based on the two hexagonal grouping structures with rotation-invariant properties.

Figure 1.2 illustrates an overview of the 2D object localisation system proposed in this thesis. It can be briefly divided into two sections: pose estimate and edge labelling. The initial pose estimate is obtained by using Random Sample Consensus (RANSAC) [Torr and Murray, 1997] after the object detection process implemented by means of Generalised Hough Transform (GHT) [Ballard, 1981]. In this case, any appropriate feature can be employed for pose estimate, such as SIFT [Lowe, 2004]. Thereafter, the proposed local feature descriptors extracted at edge contour points are employed to match and refine the pose estimate, based on which, the edge features are utilised again to perform edge contour point matching and finally label the object edge contours.

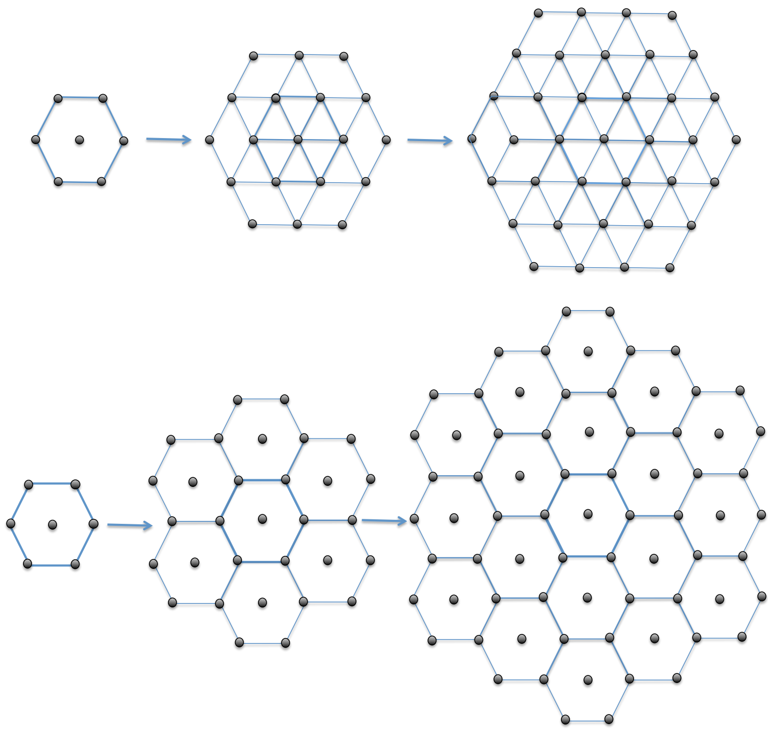


Figure 1.1: Two different hexagonal grouping structures for generating hierarchical feature descriptors. For each row, from left to right, the hexagonal structure changes from the first level to the third level of the hierarchy.

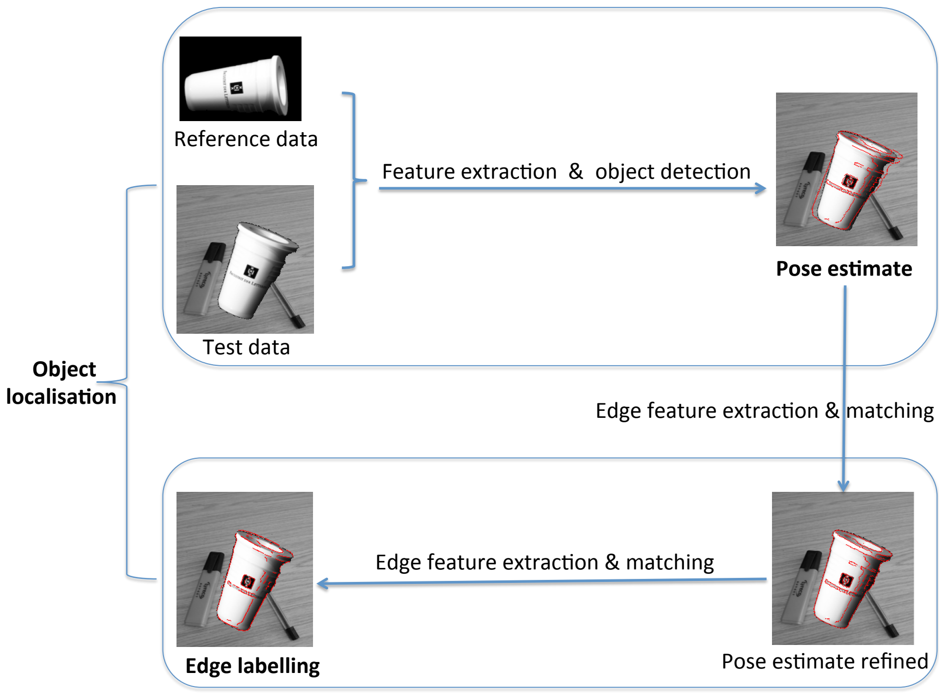

Reference data
Test data
Feature extraction & object detection
Pose estimate
Object localisation
Edge feature extraction & matching
Edge feature extraction & matching
Edge labelling
Pose estimate refined

Figure 1.2: Overview of the 2D object localisation system

1.4 Major Contributions

The major contributions of this thesis are summarised as follows:

- Two different kinds of hexagon-based hierarchically grouped structure (HHGS) are introduced for local feature extraction, which demonstrates the potential benefits of utilising hexagonal structures to improve machine cognitive abilities;
- A set of new hierarchical descriptors is generated based on the two HHGSs, which can be used to represent local features extracted at both edges and corners;
- A new method to achieve rotation invariance for local feature descriptors is proposed by utilising the rotational symmetry of the hexagonal structure, which can reduce the total time consumption of computing rotation-invariant feature descriptors;
- An alternative method to utilise the shape information for 2D object localisation is introduced to avoid the dependence on shape contour detection that can be perturbed significantly by a cluttered background, by matching edge contour points within a constrained search mechanism.

1.5 Thesis Organisation

The remainder of this thesis is organised as follows: Chapter 2 provides a comprehensive review of the related work of object detection and localisation, feature extraction, and image processing on hexagonal grid. Chapter 3 describes an initial investigation of the hierarchical feature extraction process based on both rectangular and hexagonal structures. Chapter 4 proposes a new rotation-invariant hierarchical descriptor HexSHoG based on the hexagonal grouping structure and also introduces a 2D object localisation system. To improve the time efficiency of computing the hexagon-based hierarchical feature descriptor, a set of HexBinary descriptors is proposed based on the binary coding method in Chapter 5. Furthermore, Chapter 6 introduces an investigation into optimising the configuration of the feature sampling structure and the parameters of the hexagon-based hierarchical descriptor. Finally, Chapter 7 draws an overall conclusion along with the future work regarding the work presented in this thesis.

1.6 List of Publications

The work described in this thesis has been presented in the following publications:

- Liu, Y. and Siebert, J. P. (2014). Contour localisation based on matching dense HexHoG descriptors. In *International Conference on Computer Vision Theory and Applications (VISAPP 2014).*
- Liu, Y., Aragon-Camarasa, G., and Siebert, J. P. (2014). Object edge contour localisation based on HexBinary feature matching. In *International Conference on Robotics and Biomimetics (ROBIO 2014)*, IEEE.
- Liu, Y. and Siebert, J. P. (2016). HBP: Hexagon-Based Binary Descriptors. In *International Conference on Computer Vision Theory and Applications (VISAPP 2016).*

Chapter 2

Background and Literature Review

This chapter represents work related to object detection and localisation which has been explored extensively in recent decades, and still remains an active topic in the field of computer vision. The majority of the object detection systems reported in the literature can only indicate the approximate position of objects within an image, each object localised within a "bounding box". Recently reported research can localise objects more precisely in terms of pixel labels, and is able to label the bounding edge pixels defining an object. As one of the crucial steps in object detection and localisation, local feature extraction techniques, and critically, feature representation methods, are also overviewed in this chapter. In addition, the development of image processing techniques based on hexagonal sampling tessellations is also discussed, since hexagonal geometry has inspired the construction of the new feature descriptors proposed in this thesis.

2.1 Object Detection and Localisation

2.1.1 Object Detection

Object detection has been one of the most significant and challenging topics in computer vision community. It is a crucial step for machines to understand the images and then lead to further vision tasks. The main difficulty of recognising an object in the image is to deal with the variation of the object appearance under different viewpoints, translations, rotations or scales. It will be even more difficult when the object appears in a cluttered background, and part of it is occluded. A large number of object detection approaches have been developed in recent decades. The dominant approaches of them could be generally divided in two categories: one is the sliding window method which was first introduced in [Rowley et al., 1995] and always needs to train object models first; the other is the feature-based voting system by using Generalised Hough Transform (GHT) [Ballard, 1981].

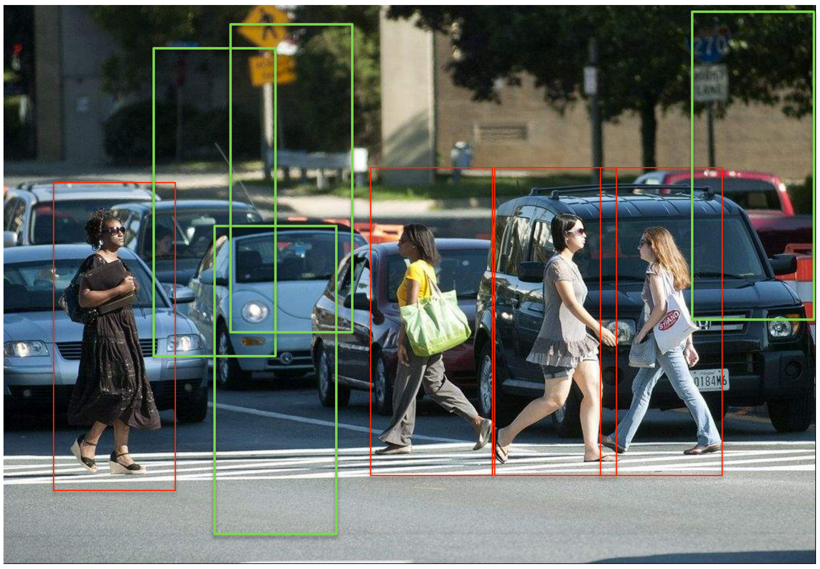


Figure 2.1: Pedestrian detection result: the red rectangle indicates a pedestrian detected in the region, and the green rectangle shows that no pedestrian appears in the covered area.

Sliding Window Detector

Sliding window detector has been widely used in many computer vision tasks, such as face detection [Rowley et al., 1998; Viola and Jones, 2001, 2004; Zhang and Zhang, 2010], pedestrian detection [Papageorgiou and Poggio, 2000; Mohan et al., 2001; Dalal and Triggs, 2005; Dollar et al., 2012] and car detection [Schneiderman and Kanade, 2000; Grabner et al., 2008; Shao et al., 2012]. It considers all the sub windows of the image and makes a decision of whether the sub windows contain the object through a classifier. The whole pipeline of using sliding window detector could be summarised as the following steps:

- *Training*: A large set of $m \times n$ sub-windows is sampled from images which include positive examples containing the object and negative examples without the object. Features are computed in all the sampled sub-windows and a classifier is trained to distinguish the positive examples and the negative examples;

- *Detecting*: A $m \times n$ sub-window slides across the whole unknown image, and each generated sub-window is applied by the classifier to compute a score S of defining whether the object appears in the sub-window;

- *Defining*: If S is bigger than the threshold T, the corresponding sub-regions in the unknown image are the candidates to be defined as containing the object of interest. Non-maximum suppression is employed to delete those heavily overlapped candidates and keep the one having the highest score.

Sliding window method indicates the approximate location of the object in a bounding box as shown in Figure 2.1. Since objects could appear in the image with different scales, the sliding window needs to search over different scales of the image. The object appearance may also vary in the image because of the viewpoint changes, non-rigid deformations or the intra-class variations. Therefore, deformable models of the objects are trained to deal with the object variations in appearance, such as deformable template model [Cootes et al., 2001], constellation model [Fergus et al., 2003] and pictorial structure model [Felzenszwalb et al., 2010]. However, training models and classifying all regions from different scales are both expensively computational processes.

Voting-based Detector

The initial Hough Transform was introduced to detect analytically defined shapes, such as lines, circles [VC, 1962]. Then it was developed as Generalised Hough Transform (GHT) to be able to detect arbitrary shapes using the principle of template matching [Ballard, 1981]. This is achieved by creating a R-table that records the information from the reference point and the edge pixels. Each edge pixel casts a probabilistic vote for the object reference position in the test image. Then an accumulation matrix is built to show the vote distribution. The peak in the matrix could be considered as the existence of the reference point of the object in the test image. In recent decades, Hough-based detection has been widely used to detect arbitrary objects. The voting elements are no longer restricted to edge pixels. Boundary fragments [Opelt et al., 2006], distinctive points [Leibe et al., 2008], local patches [Leibe et al., 2004; Okada, 2009] and regions [Gu et al., 2009] could also contribute to define the object existence by GHT.

In Lowe's paper [Lowe, 1999], Hough Transform is employed to recognise the object through matching local features from the test image to the pre-stored features of the object models. A four dimensional Hough space, which includes the parameters (x, y, σ, θ), is built to vote for the object pose. (x, y) shows the feature's location, and σ, θ denote the scale and orientation of the feature, respectively. The affine transform between the matched points from the image model to the test object could be written as:

$$\begin{bmatrix} p \\ q \end{bmatrix} = \begin{bmatrix} m & -n \\ n & m \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}, \tag{2.1}$$

$$m = S\ cos\theta, \tag{2.2}$$

$$n = S\ sin\theta, \tag{2.3}$$

where (x, y), (p, q) denote the point positions from the model image and test image, respectively; (t_x, t_y) is the image translation; S is the image scaling, and θ is the image rotation.

The Equation.2.1 could be written in an alternative way as:

$$\begin{bmatrix} p \\ q \\ \vdots \end{bmatrix} = \begin{bmatrix} x & -y & 1 & 0 \\ y & x & 0 & 1 \\ & \ldots & & \end{bmatrix} \begin{bmatrix} m \\ n \\ t_x \\ t_y \end{bmatrix}. \tag{2.4}$$

This equation can be used to describe all the matched points, and it can be conciser as:

$$B = Aw, \tag{2.5}$$

where *B* and *A* are the point sets of matches from the test and model images, respectively. In this equation, *w* is determined by the least-square solution that minimises the sum of squares of the distance between the projected model locations and the corresponding test locations. By using GHT, the work in [Lowe, 2004] pointed out that even occluded objects could also be identified by only 3 matched features. Random Sample Consensus (RANSAC) [Fischler and Bolles, 1981], as a well-known robust fitting method, can then be applied after GHT to discard outlier features and use all the inlier features to define a more precise w.

One of the main objectives in this thesis is to localise the object with edge contours labeled, which is based on the previously achieved detection result. The test dataset is comprised of 2D images including objects without intra-class variation and scale changes. Therefore, to better evaluate the sensitivity to rotation changes for the proposed feature descriptors, no training process is required, and the voting-based detection method is employed directly to produce the initial detection result in this thesis.

2.1.2 Object Localisation

Undoubtedly, knowing the exact object location after detection provides more valuable information for deeper automatic understanding of the image. In most of the cases, the detected object is localised in a bounding box [Lampert et al., 2008; Ikizler-Cinbis and Sclaroff, 2010; Feyereisl et al., 2014], which can only indicate the approximate location of the object without knowing the exact pose information. For instance, the detection method successfully recognises a bird in the middle of the image, but gives no information about whether the bird is flying or standing for a rest, which can be inferred from the accurate pose of the bird. In recent years, some articles have demonstrated that their work can delineate the object position more accurately with the edge contours labeled, such as the work in [Yu and Shi, 2003; Gu et al., 2009; Hariharan et al., 2014].

Object Localisation in A Bounding Box

Many different methods have been dedicated to localise the object in a bounding box. [Dalal and Triggs, 2005] proposed HOG features to train a linear support vector machine (SVM) classifier to localise humans. Based on HOG, [Felzenszwalb et al., 2010] learned deformable part-based models to localise variable object classes with latent SVM. [Ferrari et al., 2008] employed the sliding window detector to classify the test class based on the local shape features formed by chains of k adjacent, roughly straight contour segments (kAS). Most of these systems of localising the object in a bounding box are achieved by using a binary sliding window classifier, which is trained to scan over all the possible locations in the image, and the predicted object is present in the sub-window.

Different from the above localisation systems with binary training process, [Blaschko and Lampert, 2008] proposed a structure-output learning method specifically designed for optimising the task of localisation and avoided to model the problem as a binary classification. Based on this work, [Feyereisl et al., 2014] introduced a structured predication algorithm with privileged information to improve the predictive models in training and consequently improve the localisation accuracy. However, the benefit of using privileged information is dependent on the number of training examples.

All these sliding window methods need exhaustive search, which results in expensive computation. Some researches have been dedicated to reduce the computing load. [Lampert et al., 2008] proposed Efficient Subwindow Search (ESS) based on branch-and-bound approach to find the globally maximal region with results equivalent to the exhaustive sliding window search. [Yeh et al., 2009] introduced a data-dependent branch-and-bound formalism to improve both the localisation accuracy and efficiency. [Serrano et al., 2013] proposed a data-driven localisation method by transferring the bounding box of the most similar image from the database. Each image was represented as a probability map consisted of patches with scores computed based on the patch classifier. It avoided doing the sliding window search by finding the nearest neighbour for the probability map, which dramatically reduced the computing time of retrieval process.

A common limit the above-introduced methods have is to only indicate the approximate position of the object in a bounding box. It is desired to segregate the object from the image background, or illustrate the localised appearance of the object in the image. To this end, the positions of the edge contours of the object are required to be well provided in the localised result. A straightforward method to localise the object edge contours is to utilise the shape information. Though some articles used edge contour features or segmentation results to assist the detection task, they still did not give the detection result with edge contours localised [Shotton et al., 2005; Schlecht and Ommer, 2011; Parkhi et al., 2011].

Object Localisation with Edge Contours Labeled

To delineate the localised object more accurately, shape information of the target object has been utilised in many articles. [Zhu et al., 2008] introduced a Contour Context Selection framework for detecting objects in cluttered images. Shape context [Belongie et al., 2002] was employed to characterise the control points those nearby the semantic contours. The one-to-one correspondence matching between the semantic contours made the shape detection more immune to the accidental alignment to the contours from the background. However, the performance was quite sensitive to the semantic contour selection. [Heitz et al., 2009] proposed an approach called Localising Object Outlines using Probabilistic Shape (LOOPS) based on learning deformable shape models using landmarks on the outline of the object shape. Therefore, the final localisation result was not represented by the real shape contours of the target object, but by the produced shape outline. [Ferrari et al., 2010] extended their early work by learning class models based on the local shape feature kAS and detecting class instances in new images with boundaries localised. However, the final localisation result was the output of the learned shape model, which was still not the localisation of the real shape contours of the object.

Different from the above-introduced approaches, some articles proposed methods to localise the object by cooperating segmentation techniques. [Borenstein and Ullman, 2002] proposed a top-down class-specific segmentation protocol to identify the structure of an object by means of high-level information, instead of using the image-based criteria. This method segmented the object out from the background with the object boundaries delineated by means of previously learned shape primitives, which was still not by the real object contours from the test image. [Gu et al., 2009] introduced a recognition framework based on matching regions, and utilised the segmentation result to recognise and segregate the object from the image background. [Yang et al., 2012] dealt with object detection as a labelling or matching problem between the object model parts and the test object parts. A correspondence graph, which contained vertexes representing pairs of model segments and test segments, was constructed and optimised to a dominant sparse set by certain constrains, which finally labeled the object segments in the test images. [Dai and Hoiem, 2012] improved the localisation of detected objects by using the colour model, edge cues, and segment cues, based on the initial localisation in a bounding box derived from Deformable Parts Model (DPM) [Felzenszwalb et al., 2010]. A common step the above methods used is to learn object models in order to deal with the intra-class variation or shape deformation of the object. Therefore, the final localisation result is heavily dependent on the quality of the learned models. Moreover, It is an expensively computational step for learning object models especially when the size of the dataset is increasing.

Localising object with the shape delineated is a difficult research topic when the object is localising in a cluttered image. Most of the localisation systems are implemented by utilising shape contour features or segmentation techniques, which makes the localisation performance heavily dependent on the shape contour detection result or the segmentation result. Since shape information is essential to the object localisation, and edge points are the basic components for the object shape, directly matching edge contour points rather than linear shape contours can provide an alternative method to localise object accurately. Moreover, there are so many factors affecting the final localisation result, such as intra-class variation or shape deformation of the object, feature detection and representation, and feature matching. Few articles propose object localisation system by firstly evaluating on images without intra-class variation or shape deformation but with only affine transformations. In this thesis, a 2D object localisation system is investigated based on matching edge contour points in a simple situation, where the test object has no intra-class variation and shape deformation, but only has 2D rotation changes, which could provide further insight of how the feature extraction and matching techniques influence the localisation performance without regarding the model learning step that is used to deal with intra-class variation or shape deformation.

2.2 Feature Extraction

As one of the most significant steps of object detection and localisation, feature extraction techniques are overviewed in this section. In general, object features could be extracted globally or locally for image processing. Global feature is generated as a reduced dimension vector compared to the original image but represents the image information in a compact way. It brings certain benefits for vision analysis such as image classification. However, when it comes to the image containing occlusions or cluttered background around the target object, the global feature will mis-represent the object information. Therefore, local features achieve more attention because they can avoid such weakness as global features have. The main work of this thesis is based on local features. Accordingly, the related work of local feature extraction is overviewed in this section.

The first step of feature extraction is usually to decide where to extract the feature. They could be rather sparsely sampled as most of the feature detectors do, or very densely sampled at each local pixel of the image. These sparsely sampled features can be efficiently computed and represent the object distinctively to complete further vision processes. However, for certain cases, only using sparse features may not achieve desired results, such as analysing objects which do not have enough distinctive interest points. These objects may not be distinguished by the less representative features. Therefore, dense sampled features can be the alternative choice to provide more information for vision tasks, though it results in more expensive computation. In summary, whether to extract local features sparsely or densely is really dependent on the type of images and tasks.

2.2.1 Feature Representation

The ideal feature descriptors are desired to be distinctive and invariant to different kinds of image variations, such as geometric and photometric transformations. In addition, they also need to deal with noise and cluttered background interruption when it comes to natural image processing. The simplest method to represent a feature is using the intensity value of the feature position. However, the intensity value could be changed easily with different illuminations and transformations of the image. Therefore, applying certain functions in the neighbourhood of the feature position is a potential way to provide transform-invariant feature descriptors. A significant number of local feature descriptors have been introduced in recent decades. We will give an overview of the state-of-the-art descriptors, which could be generally divided into two categories: histogram-based floating point descriptor and binary descriptor.

Floating-point Descriptor

Many state-of-the-art local descriptors have been proposed based on histogram of local oriented gradients. A simple overview of it is illustrated in Figure 2.2. Scale Invariant Feature Transform (SIFT), as a gold milestone in feature extraction area was first introduced in [Lowe, 1999] and then extended in [Lowe, 2004]. It detects the feature points in the multi-scale space constructed by Difference of Gaussian operator, and then a local patch around the feature point with 16×16 is defined to compute the feature descriptor. This patch is subdivided into 4×4 subregions to accumulate the orientation histograms, which are weighted by the gradient magnitude and then concatenated to form the descriptor vector. Due to its impressive performance, SIFT has been a widely used benchmark and applied in many computer vision tasks, such as object detection [Piccinini et al., 2012], image retrieval [Deselaers et al., 2008], and object tracking [Zhou et al., 2009].

Based on SIFT, PCA-SIFT [Ke and Sukthankar, 2004] claimed to have faster matching and higher accuracy in image retrieval performance than SIFT. Different from SIFT, PCA-SIFT extracts a 41×41 patch centred on the feature point and computes the vertical and horizontal gradient maps to generate the feature vectors. These vectors are then projected into the eigenspace to achieve the final compact descriptor through Principle Component Analysis (PCA) [Wold et al., 1987]. Both SIFT and PCA-SIFT are normalised by the vector magnitude to reduce the feature sensitivity to illumination changes.

Gradient Location and Orientation Histogram (GLOH) [Mikolajczyk and Schmid, 2005], proposed as an extension to SIFT, computes the gradient distributions in a log-polar grid with

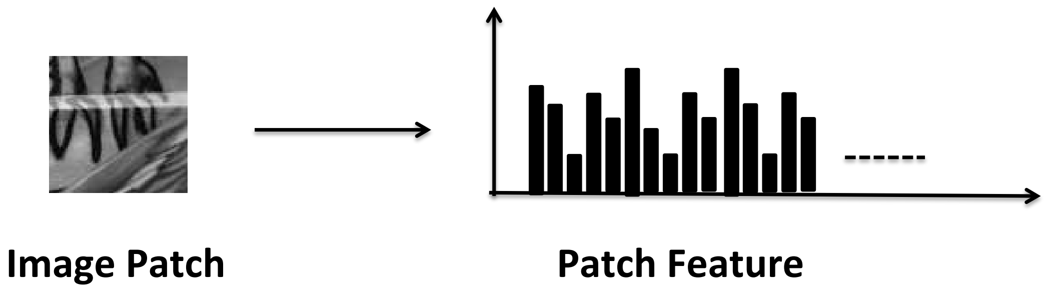


Figure 2.2: An image patch is represented by a histogram of local oriented gradients.

3 bins in radial direction and 8 bins in angular direction. Then PCA is also implemented to reduce the final descriptor dimensionality. The difference of the sampling grid between SIFT and GLOH is shown in Figure 2.3. The evaluation performances show that GLOH does not have better robustness and distinctiveness than SIFT.

In order to compute descriptors faster without sacrificing the performance SIFT achieves, Speeded Up Robust Features (SURF) was proposed in [Bay et al., 2006]. The patch centred around the feature point is subdivided into 4 $\times$ 4 subregions, in each of which a sum of Haar wavelet responses in horizontal and vertical directions is computed efficiently through integral images, and only a 4-dimensional descriptor is generated, which results in a 64-dimensional descriptor for the whole patch and provides faster feature matching compared to the 128-dimensional SIFT descriptor. [Agrawal et al., 2008] introduced a modified SURF (M-SURF), whose feature region is firstly expanded and then subdivided into overlapped subregions. The vector is weighted by a Gaussian function centred at the feature point, while each wavelet response in the subregion is weighted first by a Gaussian function centred on the subregion centre. Based on this way, M-SURF is claimed to have better ability to handle the boundary effects. In [Alcantarilla et al., 2012], KAZE feature was introduced by combining M-SURF with nonlinear diffusion filtering, in order to improve the localising accuracy of features compared to the features generated based on Gaussian filtering.

Different from the above descriptors, some descriptors are proposed initially for specific

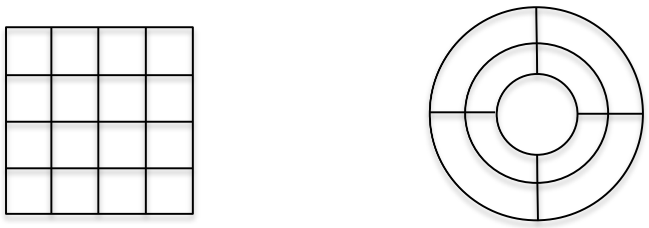


Figure 2.3: Sampling grids: the left is the Cartesian sampling grid in SIFT, redrawn from [Mikolajczyk and Schmid, 2005], the right is the log-polar sampling grid in GLOH, redrawn from [Mikolajczyk and Schmid, 2005].

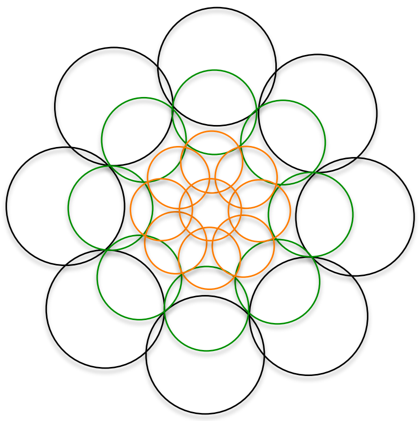


Figure 2.4: DAISY sampling structure: each circle indicates a Gaussian smoothed region, and the centres of them are the sampling positions for computing DAISY, redrawn from [Tola et al., 2008].

applications. [Dalal and Triggs, 2005] proposed Histograms of Oriented Gradients (HOG) descriptors for human detection. The study showed that locally normalized HOG descriptor achieved better performance for human detection compared to other existing feature descriptors. It employed the edge orientation histogram like SIFT does, but computed on a densely overlapped grid of uniformly spaced cells. [Tola et al., 2008] proposed another local image descriptor named DAISY designed for dense wide-baseline matching purpose. It was inspired by SIFT and GLOH, but computed densely and quickly at every single pixel without degrading the feature robustness. The computational efficiency was improved through replacing the weighted sums of gradient norms by convolutions of the original image with quantised oriented derivatives of Gaussian filters. The sampling grid of DAISY is different from the rectangular grid of SIFT. It uses a circular grid similar as GLOH but with 25 overlapped sampling locations as shown in Figure 2.4.

There are also some descriptors proposed based on the variations of the above-introduced features [Chandrasekhar et al., 2009; Alcantarilla et al., 2012]. However, all of those descriptors are almost extracted in the same pipeline as shown in Figure 2.5, which was first concluded by [Winder and Brown, 2007] then analysed more deeply in [Brown et al., 2011], where the framework of generating such patch-based local descriptors was concluded as following 4 blocks:

- *G-block*, an optional step in a pre-processing stage to smooth the input patch with a Gaussian kernel;
- *T-block*, a step to transform the smoothed patch by linear or non-linear operations and give the elements of the feature vector;
- *S-block/E-block*, a stage to provide a parametric pooling of accumulating weighted vectors from the previous stage and concatenating them to form the patch descriptor,

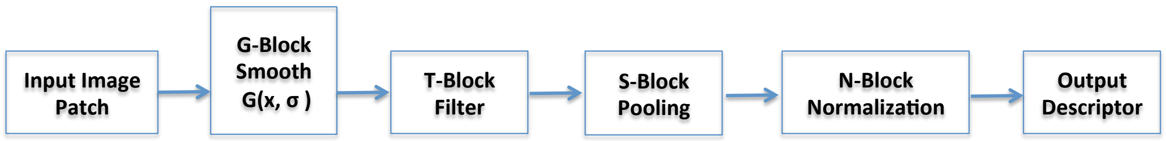


Figure 2.5: Processing stages for the generic descriptor algorithm, redrawn from [Winder and Brown, 2007].

or a non-parametric pooling by projecting the vectors into a certain eigenspace and then achieving more robust descriptor with less dimensions;

- *N-block*, the final stage with normalising the descriptor to remove the dependency on image contrast, and sometimes it can be followed by another *E-block*.

In the above framework, the candidate algorithms can be cast into the corresponding blocks with learned optimised parameters to generate robust and distinctive feature descriptors. All the above-introduced descriptors computed based on this framework have been proved to have high quality performances in many computer vision applications. However, they will suffer from the expensive computation when they are required to serve the real-time application on smart devices with a huge number of descriptors. Therefore, fast computing with low storage requirement is preferred for local feature descriptors.

Binary Descriptor

With the requirement of computing feature descriptors faster, binary coding method is employed for patch-based descriptors. Binary Robust Independent Elementary Features (BRIEF) [Calonder et al., 2010] is such a binary string descriptor, which contributes to better efficiency of computing, matching and storage without decreasing the recognition performances compared to SURF and U-SURF on certain public datasets. It is constructed based on the pairwise comparisons of intensity in an image patch without considering rotation and scale changes. The similarity between the descriptors is efficiently computed by Hamming distance rather than L_2 norm distance. The binary intensity test τ for constructing the binary string descriptor is defined on patch p of size $S \times S$ by:

$$\tau(p; x, y) = \begin{cases} 1 & \text{if } p(x) < p(y) \\ 0 & \text{otherwise,} \end{cases} \tag{2.6}$$

where $p(x)$, $p(y)$ are the Gaussian smoothed pixel intensities at point x and y, respectively.

The feature string is defined as:

$$f_n\left(\mathbf{p}\right) := \sum_{1\leq i\leq n} 2^{i-1}\tau\left(\mathbf{p};\mathbf{x}_i,\mathbf{y}_i\right). \tag{2.7}$$

Based on BREIF, some variations of it are proposed to cope with more complicated applications. D-BRIEF [Trzcinski and Lepetit, 2012] achieves the compact binary descriptor with only 32 bits by thresholding coordinates in the more discriminative subspace, which is similar as the space projection in [Strecha et al., 2012]. Oriented FAST and Rotated BRIEF (ORB) [Rublee et al., 2011] is a scale and rotation-invariant version of BRIEF. The scale invariance is achieved by detecting FAST features [Rosten and Drummond, 2005] at each level of the multi-scale pyramid image. The intensity centroid is employed to measure the key-point orientation, which is used to generate the steered BRIEF. A learning step is also processed to pick out the elements of pairs with high variance as the discriminative features.

To provide better immunity to rotation and scale changes for the binary descriptor, Binary Robust Invariant Scalable Keypoints (BRISK) [Leutenegger et al., 2011] is generated from the region centred on the keypoint in a more generally sampled pattern, which has something similar to DAISY. The sampling pattern is illustrated in Figure 2.6. The sampling points are smoothed by a Gaussian with the standard deviation proportional to the distance between the points on the respective circle. By using the same feature detection method as BRISK, [Alahi et al., 2012] proposed Fast Retina Keypoint (FREAK) descriptor based on a fast retinal sampling pattern illustrated in Figure 2.6. The exponentially changed kernel size for each sampled point provides more overlapped fields, which could bring more information and discriminative power for the key-point descriptor.

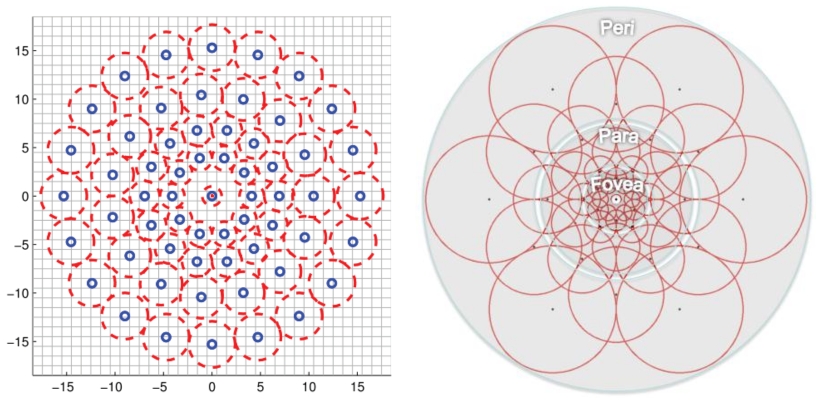


Figure 2.6: Left : BRISK sampling pattern with $N = 60$ points, reproduced from [Leutenegger et al., 2011]. Right: FREAK sampling pattern which is similar to the retinal ganglion cells distribution, reproduced from [Alahi et al., 2012]

Different from the above-introduced binary descriptors, Local Difference Binary (LDB) [Yang and Cheng, 2012] computes binary bits not only from the pixel intensity comparisons but also from the gradient comparisons. The image patch is regularly divided into a set of grids, then the average intensity and gradient in x and y directions are compared between each two grids, respectively. Therefore, one pair of grids produces a 3-bit vector. Like many other descriptors, the gradient information in LDB is computed after Gaussian smoothing, which inevitably results in the loss of object boundary information. To preserve more information of the object boundaries for feature extraction, [Alcantarilla et al., 2013] proposed the rotation and scale invariant descriptor Accelerated-KAZE by combining the modified LDB descriptor with the non-linear scale space with low requirements of computation and storage.

From floating-point descriptors to binary descriptors, local feature extraction has been greatly developed in terms of both effectiveness and efficiency. However, the size of feature receptive field always needs to be defined first to compute the distinctive local information. In the real world, there are many cues for a human being to recognise an object from an image, such as pixel intensities, edge contours, local patterns, a semantic structure or the whole shape of the object. Only focusing on a small part of the object limits the contribution of other useful information for object recognition. A large number of key-points from the images can be detected and represented as feature descriptors, but they might capture the local information that is too generic and can be easily detected in different objects. As in Figure 2.7, two different objects share a generic pattern which is covered by the red square, and the pattern is repetitive in each object. Using the above overviewed features to represent the pattern cannot help to distinguish the two different objects. However, if the local pattern size is increased to be as big as the blue squares in Figure 2.7, the pattern in the blue square can be distinctive enough to make the two objects distinguished. Therefore, hierarchical mechanism with changing the feature receptive field dynamically is an effective method to avoid pre-defining a fixed size of the feature receptive field, based on which the sampling feature may not be distinctive.

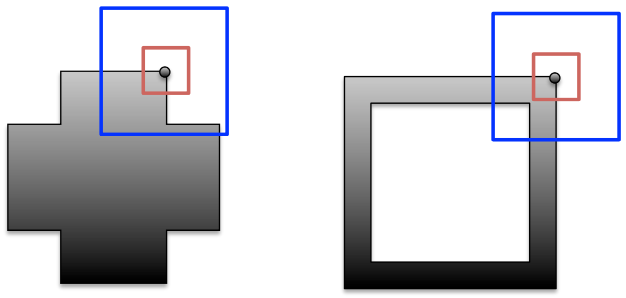


Figure 2.7: A generic pattern shared by different objects.

2.2.2 Feature Hierarchy

Extracting local features hierarchically is expected to utilise more valuable information from different levels to complete computer vision tasks. [Epshtein and Ullman, 2005] proposed an automatically extracted feature hierarchy based on the top-down mechanism. A big patch of the image is extracted first as the top level feature fragment (e.g. the face of a person), then the face fragment is broken down successively into different smaller components (e.g. eyes, lips, nose). This process is repeated until these fragments cannot be decomposed any more (e.g. eye corner, eyelid, eye pupil). It is a recursively decomposing process for object initial extracted features. Based on this method, they developed a a semantic hierarchy to represent the appearance of object parts and their statistical dependencies from different levels [Epshtein and Ullman, 2007]. The semantic hierarchy is shown as in Figure 2.8. Similarly, [Kokkinos and Yuille, 2009] also proposed a top-down hierarchy by decomposing the object into parts, breaking the parts into contours, and partitioning the contours into straight edge segments. Accordingly, a parsing tree is constructed with the root showing the whole object structure and the leaves depicting the edge tokens. [Porway et al., 2010] extended this hierarchical structure to the application of scene understanding. These top-down methods could provide detailed interpretation between the subparts of the feature hierarchy. However, a relatively big patch of the image needs to be localised first as the initial root of the hierarchy, which makes the top-down hierarchy difficult to construct in cluttered images.

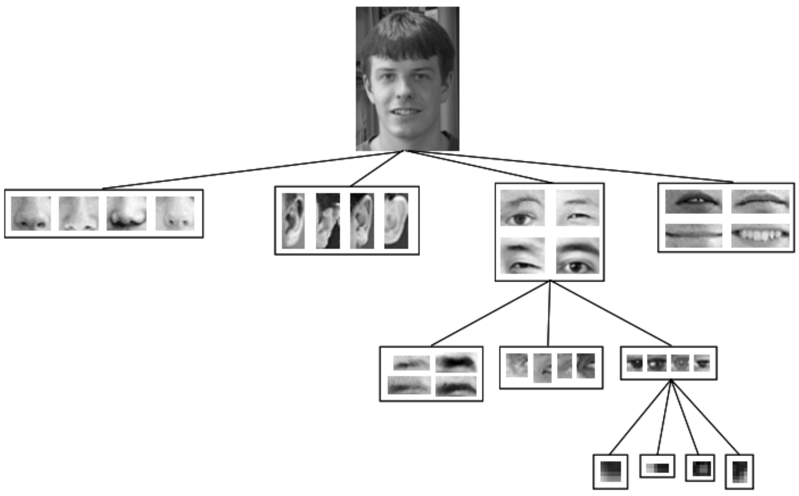


Figure 2.8: Schematic illustration of a semantic hierarchy, represented from [Epshtein and Ullman, 2007].

Different from the top-down hierarchy, the bottom-up hierarchy could provide the analysis of the image from a local small part of the image, which can be easier to extract in a dense or sparse grid. Hyperfeatures [Agarwal and Triggs, 2006] is one of such hierarchical features. The image is treated as a collection of separate fragments, and the spatial co-occurrence of these fragments is utilised to improve the feature distinctiveness. The low level fragments are extracted densely using the first order statistical features with histograms, then the descriptor is vector quantised by the learned codebook. Consequently, a histogram of the code distribution over a local neighbourhood is accumulated to generate a higher level feature which is defined as *hyperfeatures*. The limitation of this feature hierarchy is that the final feature descriptor has missed information of the spatial relationship between the neighboured patches. [Leonardis and Fidler, 2011] introduced a hierarchical representation of objects for visual recognition and detection. Gabor filter is used to describe the local oriented edges, and the image is transformed into a list of parts, which is represented by the vector with position, orientation and the parameters denoting the principal axes of an elliptical Gaussian that encodes the variance of the position around the parts. The local parts in lower layer are combined into larger units in manner of spatially flexible composition. The architecture of the hierarchy is illustrated in Figure 2.9. This method encodes the spatial information between the local parts as the feature descriptor. However, the step to learn shape models in each layer of the hierarchy is computationally expensive.

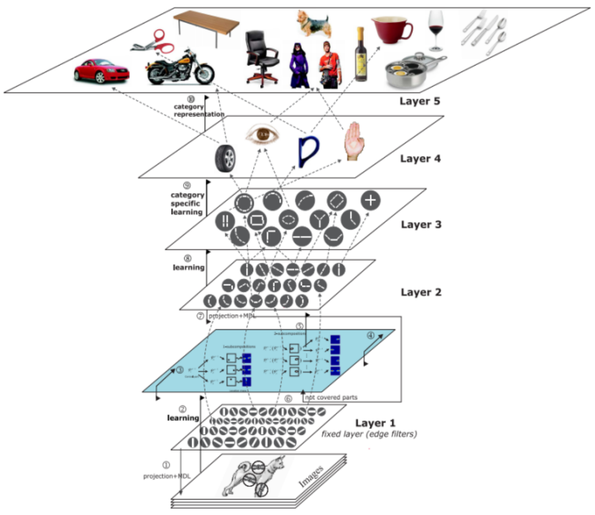


Figure 2.9: Hierarchical compositional representation of object categories, represented from [Leonardis and Fidler, 2011].

In addition, there are also some hierarchical object representation models constructed based on deep learning algorithms. [Lee et al., 2008, 2009, 2011] proposed methods for hierarchical representation all based on the deep belief network [Hinton et al., 2006], in which the feature hierarchy is constructed by layer-wise progressive training: each layer of the hierarchy is a learned representation consisting of a set of functions mapping from the previous layer. [Yang and Yang, 2011] proposed a deep learning model termed as Hierarchical Model with Sparsity, Saliency and Locality (HSSL). It utilises three steps to build up the hierarchical representation: sparse coding, saliency pooling and local grouping of each layer of the hierarchy. It is claimed to be more generically applied to different vision tasks, because the sparse coding is directly from images in an unsupervised data-driven manner, which is different from [Yang et al., 2009] using sparse coding and max pooling in a spatial pyramid matching structure based on hand-crafted descriptors. [Bo et al., 2011] proposed hierarchical kernel descriptors to compute image-level features in a recursive process. [Bo et al., 2010] generated hierarchy of kernel descriptors by aggregating spatially nearby patch-level features, which are learned by transforming pixel attributes using match kernels.

These deep learning-based methods to construct the feature hierarchy have great potential to learn stable and faster local features. However, they also have limits in common at current stage: The system always needs a large number of images to train, which is computationally expensive. Besides, it is not certain that the learned system can also work for images from completely different domains. To develop the capability of machine visual perception, it is essential to understand clearly each step of the learning process, however, it is difficult to interpret what the deep learning system has exactly done during the learning step. Therefore, in this thesis, a feature hierarchy is constructed by grouping local patches in a simple but effective mechanism to generate distinctive local feature descriptors.

2.2.3 Orientation Assignment of the Local Features

The conventional method to achieve rotation invariance of local features is to align the feature receptive field with its dominant orientation. Therefore, one of the key factors of achieving rotation invariance is to compute the local dominant orientation. The common approaches to computing the local dominant orientation could be concluded as follows:

SIFT method: Firstly, SIFT [Lowe, 2004] computes the local gradient magnitude *M* and orientation *O* at each pixel *p(x,y)* of a local region in the Gaussian smoothed image *G*:

$$M(x,y) = \sqrt{(G(x+1,y)\text{-}G(x\text{-}1,y))^2 + (G(x,y+1)\text{-}G(x,y\text{-}1))^2}, \tag{2.8}$$

$$O(x,y) = arctan(G(x,y+1)\text{-}G(x,y\text{-}1)/G(x+1,y)\text{-}G(x\text{-}1,y)). \tag{2.9}$$

Then an orientation histogram is formed by distributing all the orientations in the local region into 36 bins. Each Orientation *O(x,y)* added to the histogram is weighted by its magnitude *M(x,y)*, which has been Gaussian weighted by a circular window covered over the region. The highest peak of the histogram and up to three other local peaks (within 80% of the highest peak) are fitted by a parabola to refine the peak accuracy, which then is used as the local dominant orientation.

SURF method: In SURF [Bay et al., 2006] method, the Haar wavelet responses in x and y directions are calculated within a circular neighbourhood around the key-point and weighted

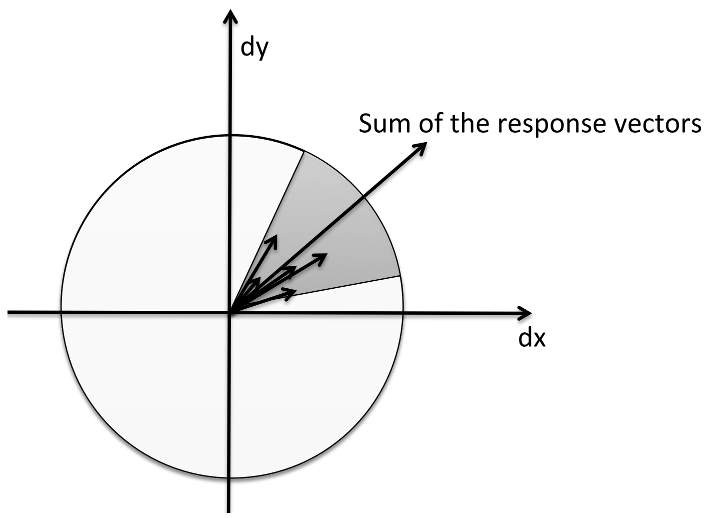


Figure 2.10: SURF orientation assignment [Bay et al., 2006]. The vector represents the wavelet response in x and y directions at each pixel in the neighbourhood.

by a Gaussian window. The wavelet response at each pixel in the neighbourhood is represented as a two dimension vector. A sliding orientation window of size $\frac{\pi}{3}$ as shown in Figure 2.10 is used to sum up all the vectors in its range and generates a long orientation vector. Compute all the long orientation vectors of all the sliding windows through the circular neighbourhood and define the longest one as the dominant orientation of the key-point.

Intensity centroid: This method was initially analysed in [Rosin, 1999] and then employed by ORB [Rublee et al., 2011]. Firstly, the local patch moments are defined as :

$$m_{pq} = \sum_{x,y} x^p y^q I(x,y), \tag{2.10}$$

then the centroid of the patch is given by :

$$C = (\frac{m_{10}}{m_{00}}, \frac{m_{01}}{m_{00}}). \tag{2.11}$$

A vector is constructed from the key-point centre to the centroid, and the orientation of this key-point is determined as:

$$\theta = atan2(m_{01}, m_{10}). \tag{2.12}$$

BRISK method: In BRISK [Leutenegger et al., 2011], the local patch gradient was estimated by:

$$g(\mathbf{x}, \mathbf{y}) = \frac{1}{N} \sum_{p \in \mathcal{S}} (\mathbf{p}_j - \mathbf{p}_i) \cdot \frac{I_j - I_i}{||\mathbf{p}_j - \mathbf{p}_i||^2}, \tag{2.13}$$

where $\mathbf{p}$ is the position vector of the sampling points in the local patch around the key-point. $\mathcal{S}$ is a set of sampling pairs with certain spatial distance and N is the number of pairs in $\mathcal{S}$.

I is the Gaussian smoothed intensity value of the sampling point. Then the local dominant orientation is defined as:

$$\theta = atan2(\mathbf{g}(\mathbf{y}), \mathbf{g}(\mathbf{x})). \tag{2.14}$$

The above four introduced methods are the dominant methods to compute the local dominant orientation for local features. The method employed to compute the local dominant orientation is usually selected in accordance with the method used to represent the local descriptor. For example, SIFT method is always employed by orientation histogram-based descriptors and binary string-based descriptors are often coupled with BRISK method. Computing the dominant orientation is an important step to achieve rotation-invariant feature descriptors, as it influences the tolerance of local features to rotation changes. In this thesis, with the new feature descriptors proposed, the new method to compute the local dominant orientation will also be introduced.

2.3 Image Processing on Hexagonal Grid

Digital image processing on hexagonal grid could be traced back to 1960s. Though the traditional method to do digital image processing is based on square grid, in human visual system, the rods and cones in the fovea are arranged in roughly hexagonal topology [M.H.Pirenne, 1967; Curcio et al., 1987; Hubel et al., 1995], which attracts great attention to analyse digital images in hexagonal grid. However, the device for image acquisition and visualisation is not currently commercially available. This is the main reason to limit the development of machine vision processing on hexagonal grid.

Though the conventional acquisition sensors and display devices are based on square grid, there still many researchers proposed methods to resample the image from the square sampled grid onto a hexagonal grid. [Hartman and Tanimoto, 1984] generated each pixel in the hexagonal grid by averaging the two vertically adjacent square pixels. [Her and Yuan, 1994] tested different interpolation methods to generate a pseudo-hexagonal grid on a regular rectangular grid device. [Watson and Ahumada Jr, 1989] constructed a hexagonal orthogonal-oriented pyramid based on the affine relationship between the square and hexagonal structures. [Overington, 1992; Fitz and Green, 1996] employed the idea 'Brick wall', which is an approximation to the hexagonal grid, achieved by shifting half a pixel for each pixel in alternate rows on the square grid. Researchers have never stopped trying different methods to construct the hexagonal grid from the square grid with better visualisation and less quantisation error of the image [Van De Ville et al., 2002, 2004; Finckh et al., 2014].

The intrinsic advantages of using hexagonal grid to represent images are concluded in the

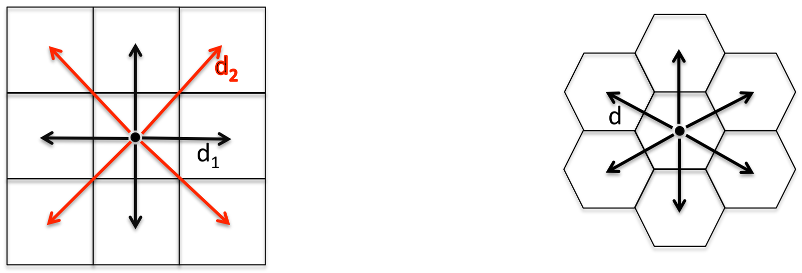


Figure 2.11: In the square grid, the pixel has two types of distance from the neighbours to it, while in the hexagonal grid, all the pixel neighbours have the same distance to the pixel.

paper [He and Jia, 2005]. Hexagonal grid has high sampling efficiency, which needs fewer sampling points and less computation time compared to square grid when equal image information is maintained [Mersereau, 1979]. It also has consistent connectivity with each neighbour of a hexagonal pixel having equal distance to the pixel, while for each pixel in square grid, there are two types of distance between the neighbours to the pixel, as shown in Figure 2.11. [Kamgar-Parsi et al., 1989] developed mathematical tools to estimate quantisation error in hexagonal sensory configurations, and yielded smaller quantisation error compared to the square configuration. Though there are three possibilities for the regular tiling in the 2D plane: a hexagon, a square, and a triangle, the hexagon provides the maximum number of symmetries. The high symmetry of hexagon also brings better angular resolution, which makes the image lose less information when it is rotated compared to the same rotation on the square grid.

These advantages of representing images on hexagonal grid have motivated people to research more about image processing on hexagonal structure. [Vidya et al., 2009] implemented various edge detection techniques based on square grid and hexagonal grid. The performance analysis indicated that edge detection based on hexagonal grid achieves better results and visual appeal of images compared to that on square grid. [Gardiner et al., 2008] proposed a multi-scale hexagonal gradient operator exclusively designed for images represented on hexagonal grid. [Shima et al., 2010] compared image alignment performances between on square lattices and on hexagonal lattices, and got better results with respect to accuracy and success rate on hexagonal lattices. [He et al., 2007] computed Local Binary Pattern for human detection on hexagonal structure, and achieved more efficient and accurate results than those on square structure. [Middleton and Sivaswamy, 2006] introduced a comprehensive work by developing a framework of hexagonal image processing and studied specific issues in the framework. These promising applications and the intrinsic advantages of the hexagonal structure inspire the hierarchical feature extraction work in this thesis.

2.4 Summary and Discussion

This chapter reviewed the related work of object detection and localisation, feature extraction, and image processing based on hexagonal grid. The main goal of this thesis is to localise objects with edge contours labeled, which is a critical requirement for many image analysis and robotics applications, such as image retargeting and robot grasping. The straightforward method to localise the object edge contours is to match linear edge contours detected from the image. However, this would rely greatly on the ability to extract such linear bounding contours consistently and reliably. Defining edge contours belonging to the object of interest is a challenging research problem, especially when the object is embedded in a cluttered background, where many false contours from the background could disturb the detection and recognition of the bounding contours. If a detected contour segment contains edge points partly comprising the desired object and partly comprising noise or background clutter, this contour will be either partially labeled incorrectly or perhaps not labelled at all. To reduce this phenomenon, a 2D object localisation system with the edge contours labeled is proposed in this thesis by matching 2D image features sampled at all edge contour points, rather than matching only linear edge contours. This is inspired by that edge contour points are the key indicators of the presence of object shape contours, and they are also simpler to obtain than linear edge contours. Accordingly, local feature descriptors sampled at the edge contour points are also investigated in this thesis.

Many state-of-the-art local feature descriptors are able to achieve rotation invariance by pre-rotating the feature receptive field to align with the local dominant orientation. Therefore, several different methods to compute the local dominant orientation have been investigated in the community. No matter which method is used to compute the local dominant orientation, rotating the feature support region according to the dominant orientation is an essential step to endow the feature descriptor with rotation invariance. It is not a computationally expensive process to rotate and resample the rectangular grid regarding the processing ability of current computers. However, it is of course increasing the computing time proportionately with the number of features extracted in the image. Accordingly, computational efficiency will be an issue when features are extracted at edge locations which are always many more than corner locations and of course when features are sampled on uniform dense grids. In addition, due to the growing requirement to process images with high resolution in real-time, reducing the time consumption of each step when generating a feature descriptor would be valuable. To this end, new sampling structures and new descriptor formulations of local features with rotation invariance are also investigated in this thesis. Inspired by the improved sampling efficiency and angular resolution of the hexagonal grid, as compared to the square grid, a set of local descriptors is proposed in this thesis based on the hexagonal structure to achieve rotation invariance. Moreover, a feature hierarchy is constructed to provide more distinctiveness to the local features.

The next chapter will present an initial investigation of the local feature descriptor based on a hexagonal structure in combination with a hierarchical grouping mechanism. This research represents the underlying work, on which the subsequent research of this thesis into generating rotation-invariant feature descriptors using hexagonal structures is based.

Chapter 3

Initial Investigation of Hierarchical Feature Representation Based on Hexagonal Grouping Structure

This chapter sets out to give an initial investigation of the hierarchical feature representation based on hexagonal grouping structure. Since images are displayed on square grid, a straightforward method to construct a hierarchical feature descriptor is to employ the rectangular grouping structure. Therefore, a local hierarchical feature descriptor is generated based on the hexagonal grouping structure and compared to that computed based on the rectangular grouping structure. The evaluation of the proposed descriptor is implemented by using images from both the Amsterdam Library of Object Images (ALOI) dataset and our newly collected natural images. With this initial investigation, the hierarchical feature representation based on hexagonal grouping structure is acknowledged as a potential approach to distinctive local representations of the object for computer vision tasks.

3.1 Motivation and Objective

Motivated by the biological visual system, representing an object in a hierarchical structure has been a common approach to characterising the object from locally to globally. As reviewed in the previous chapter, the feature hierarchy has different definitions in different literatures. In this thesis, the definition of hierarchy is related to the bottom-up grouping mechanism, which generates higher level features by increasing the patch size in a grouping mechanism to include more information into the feature descriptor. This hierarchically grouped feature descriptor serves two underlying objectives: firstly, the conventional method to define the feature region with a fixed hand-configured size is avoided and instead, the higher level grouping is driven by the underlying data in a recursive way; secondly, constructing a descriptor by a recursively grouped vector, which is generated by relatively smaller ”base” descriptors, allows the descriptors to be matched in a coarse-to-fine strategy and deals with objects with part occlusion.

Digital images are conventionally displayed on the square grid, which directly leads to the rectangular grouping mechanism for generating a hierarchical descriptor. In this chapter, in order to give an initial investigation of the hierarchical descriptor constructed based on the hexagonal grouping structure, a hierarchical feature descriptor generated based on the rectangular grouping structure is also computed. The histogram of oriented gradients (HoG) descriptor of a local patch is employed to cooperate with the two grouping structures, because it has been demonstrated to have the good property of capturing local appearance information well in many published articles, such as [Lowe, 2004; Mikolajczyk and Schmid, 2005; Dalal and Triggs, 2005; Tola et al., 2010]. Therefore, two grouping structures cooperated with HoG descriptor are explored to generate two hierarchical feature descriptors in this chapter. The property of the hexagonal grouping structure is then analysed through comparing the evaluation performances of the two descriptors applied for local feature matching.

3.2 Hierarchical Feature Extraction

In this section, the steps of generating two hierarchical feature descriptors based on rectangular grouping structure and hexagonal grouping structure are thoroughly described. The evaluation performance of their application to local feature matching will be described in the next section.

3.2.1 Feature Detection

Extracting distinctive features from parts of the object is always the first process to analyse the object. Choosing proper features can improve the accuracy and efficiency for computer vision tasks. A large number of feature detectors have been developed and could be roughly divided into two categories: sparse feature detector and dense feature detector. Many sparse feature detectors, such as corner detectors and edge detectors, have demonstrated their fast computation and good performance for vision analysis, but sometimes they might loose information which is quite important for the object analysis, while dense feature detectors can capture more information but loose the time efficiency. In this work, feature detection is not in the investigation scope. Since the core work of this chapter is to investigate new feature descriptors, corner detector as a common and well-performed technique is selected for the initial feature detection, which gives sparse sampling positions showing where to extract

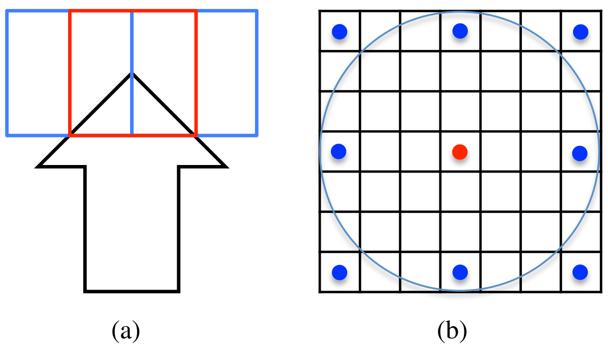


Figure 3.1: (a) Overlapped sampling of local patches; (b) A single HoG covers the patch centred at the red key-point, where the 9 blue points are the neighbours with 3-pixel chessboard distance to the red key-point, and the circle is the Gaussian kernel size for extracting HoG descriptor in this patch.

feature descriptors. Harris corner detector [Harris and Stephens, 1988] is employed in this chapter because of its wide application and good performance.

3.2.2 Feature Representation

RecHoG Descriptor

Grouping a local patch with its nearest neighbours is the direct way to cast space information into local features. Finding 8 nearest neighbours to construct a rectangular grouping is the most convenient method. Some local features such as HOG [Dalal and Triggs, 2005] and DAISY [Tola et al., 2008] apply overlapped sampling between neighboured patches to achieve more stable groupings. In this case, the overlapped patch sampling prevents the information missing especially at corners. As Figure 3.1 (a) shows, the blue squares are the basic sampling patches over the image containing the pattern of an arrow. If the patches were sampled with every patch-size shift, the angle information of the pattern would be missing. Therefore, an overlapped sampling method is needed to capture the angle information as in the red patch. Accordingly, a hierarchical feature descriptor is generated in this chapter based on the overlapped rectangular grouping structure combining with HoG representation and is termed as RecHoG.

The details of constructing RecHoG are described as follows: for a detected key-point, its 8 neighbouring points, each of which is 3 pixels away in terms of chessboard distance from it, are recorded as patch centres. Then 9 patches which are centered at the 9 centres including the key-point are extracted with size of 7×7 shown as in Figure 3.1 (b). These parameters are decided according to the empirical results from the published work in [Dalal and Triggs, 2005; Calonder et al., 2010]. HoGs of the 9 patches are computed and then concatenated to generate the level 1 rectangularly grouped feature descriptor *RecHoG1*. This process is

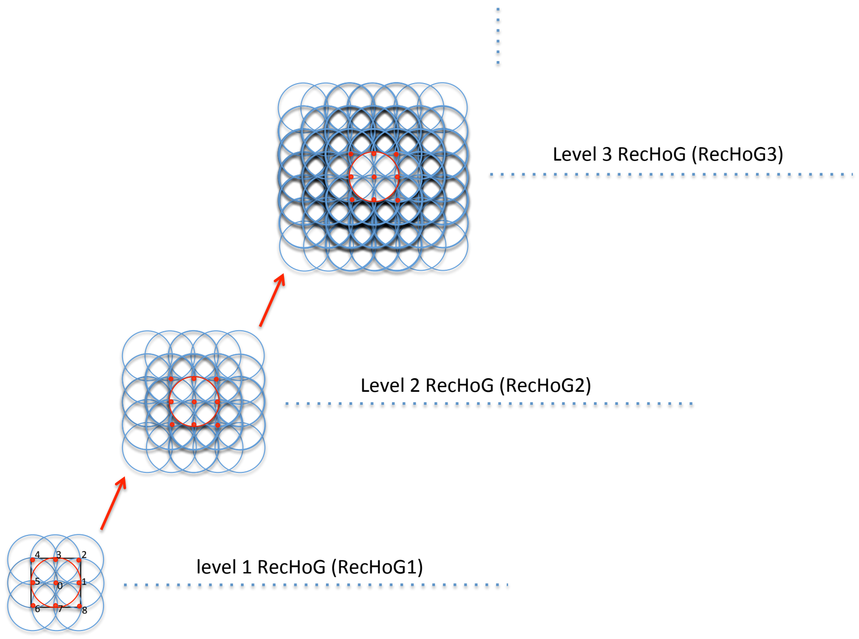


Figure 3.2: Hierarchical structure of the rectangularly grouped feature descriptor RecHoG. The 9 red points are the sampling positions of the rectangular grouping. Each circle represents a gaussian weighted patch represented by HoG descriptor.

described as shown in Algorithm 1. Then level 2 grouping descriptor *RecHoG2* is generated by concatenating *RecHoG1* descriptors of the 9 centres. As a consequence, the higher level feature descriptor based on the rectangular grouping structure is recursively constructed in the way described in Algorithm 2. The recursive grouping structure is illustrated in Figure 3.2. At each level of the hierarchy, the final feature descriptor is normalised by its magnitude to achieve the robustness to illumination changes.

Algorithm 1 Level 1 Construction of RecHoG

```
$< Px_0, Py_0 >$: $RecHoG$ centre, i.e., interest point location
$< Px_i, Py_i >$ $(i \leftarrow 1...8)$: eight neighbours with $d$-pixel away in terms of chessboard distance from $< Px_0, Py_0 >$, respectively
for $i \leftarrow 0 : 8$ do
    Construct $HoG_i$ at point $< Px_i, Py_i >$
end for
$RecHoG1 \leftarrow Normalize(HoG_0, HoG_1, ..., HoG_7, HoG_8)$
```

Algorithm 2 Recursive Construction of Higher Level RecHoG

```
$< Px_0, Py_0 >$: $RecHoGLevel_t$ centre, i.e., interest point location
$< Px_i, Py_i >$ $(i \leftarrow 1...8)$: eight neighbours with $d$-pixel away in terms of chessboard distance from $< Px_0, Py_0 >$, respectively
for $i \leftarrow 0 : 8$ do
    Construct un-normalised descriptor U-RecHoG$Level_t$ at point $< Px_i, Py_i >$
end for
$RecHoGLevel_{t+1} \leftarrow Normalize$ (U-RecHoG$Level_{(t)0}$, U-RecHoG$Level_{(t)1}$,..., U-RecHoG$Level_{(t)7}$, U-RecHoG$Level_{(t)8}$)
```

HexHoG Descriptor

There are 9 patches sampled for the first level of RecHoG, and more patches are computed with the level increasing. Considering to reduce the computing complexity but without decreasing the feature distinctiveness, the hexagonal grouping structure is employed. As reviewed in Chapter 2, hexagonal grid needs fewer sampling points and less computation time to represent equal image information compared to square grid. The consistent connectivity with equal distance from the pixel to each neighbour also brings more rotation invariance. Therefore, a hierarchical feature descriptor based on the hexagonal grouping structure is introduced to combine with HoG descriptor and termed as HexHoG.

The method to generate HexHoG is similar to that of RecHoG. For a detected key-point, 6 points, each of which is 3 pixels away from the key-point in terms of Euclidean distance, are computed as vertexes of a hexagon centred at the key-point. Two of the hexagon vertexes are on the horizontal direction. Then 7 patches centred at the hexagon centre and vertexes are extracted respectively with the size of 7×7 as shown in Figure 3.3. Therefore, the first level descriptor *HexHoG1* is generated by concatenating 7 HoGs and the higher level descriptor of HexHoG is constructed recursively by concatenating 7 lower level descriptors of HexHoG based on the hexagon structure. The main difference between RecHoG and HexHoG is the grid structure of the sampling positions: one is rectangle while the other is hexagon. The level 1 and higher level constructions of HexHoG are described in Algorithm 3 and Algorithm 4, respectively.

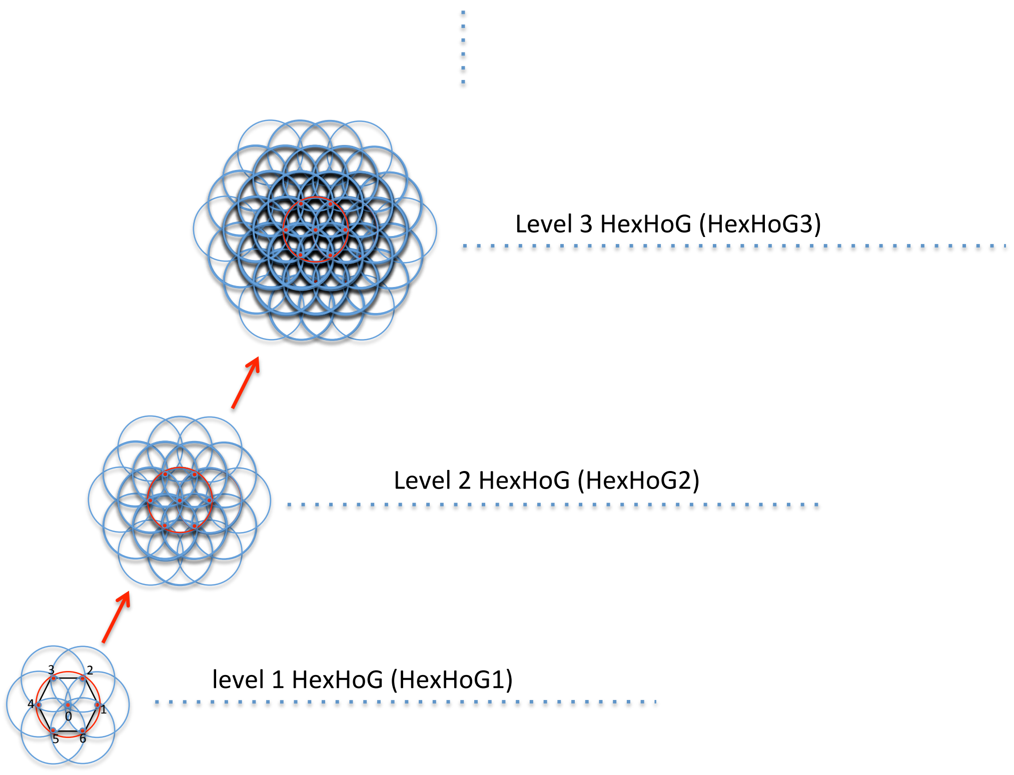


Figure 3.3: Hierarchical structure of the hexagonally grouped feature descriptor HexHoG. The 7 red points are the sampling positions of the hexagonal grouping. Each circle represents a gaussian weighted patch represented by HoG descriptor.

Algorithm 3 Level 1 Construction of HexHoG

```
$< Px_0, Py_0 >$: $HexHoG$ centre, i.e., interest point location
$< Px_i, Py_i >$ $(i \leftarrow 1...6)$: six neighbours with $d$-pixel away in terms of Euclidean distance from $< Px_0, Py_0 >$, respectively
for $i \leftarrow 0 : 6$ do
    Construct $HoG_i$ at point $< Px_i, Py_i >$
end for
$HexHoG1 \leftarrow Normalize(HoG_0, HoG_1, ..., HoG_5, HoG_6)$
```

Algorithm 4 Recursive Construction of Higher Level HexHoG

```
$< Px_0, Py_0 >$: $HexHoGLevel_t$ centre, i.e., interest point location
$< Px_i, Py_i >$ $(i \leftarrow 1...6)$: six neighbours with $d$-pixel away in terms of Euclidean distance from $< Px_0, Py_0 >$, respectively
for $i \leftarrow 0 : 6$ do
    Construct un-normalised descriptor U-HexHoG$Level_t$ at point $< Px_i, Py_i >$
end for
$HexHoGLevel_{t+1} \leftarrow Normalize$ (U-HexHoG Level$_{(t)0}$,U-HexHoG Level$_{(t)1}$,..., U-HexHoG Level$_{(t)5}$, U-HexHoG Level$_{(t)6}$)
```

3.3 Evaluation

In order to evaluate the distinctiveness and rotation invariance of the proposed HexHoG descriptor constructed based on the hexagonal grouping structure, a local feature matching system is designed in this section to compare the performances between HexHoG and RecHoG that is constructed based on the rectangular grouping structure. The experimental hypothesis is that HexHoG achieves better rotation invariance without losing much distinctiveness than RecHoG, and the higher level descriptors of the hierarchical feature are always performing better than the lower level descriptors in the limited range of levels.

Dataset

The proposed hierarchical features are evaluated based on the experiments of matching objects with different rotations. Two different datasets of images are employed for the evaluation: The first dataset (named as *Date1*) comprises 5 reference images randomly selected from Amsterdam Library of Object Images (ALOI) [Geusebroek et al., 2005] and 450 test images generated by rotating each reference image by 1° per step in range $[1, 90]^\circ$. The 5 reference images are illustrated in the first row of Figure 3.4. The second dataset (named as *Date2*) contains newly collected images with 5 reference images as shown in the second row of Figure 3.4, and 90 corresponding test images photographed by physically rotating the reference object in the real world every 5° per step in range $[1, 90]^\circ$. The third row of Figure 3.4 illustrates the examples of test images with 10 degrees physical rotation in *Date2*.

Implementation

Harris corner detector is applied in the reference image to find key-point positions, and accordingly, the corresponding key-points in the test images are also recorded. The hierarchical

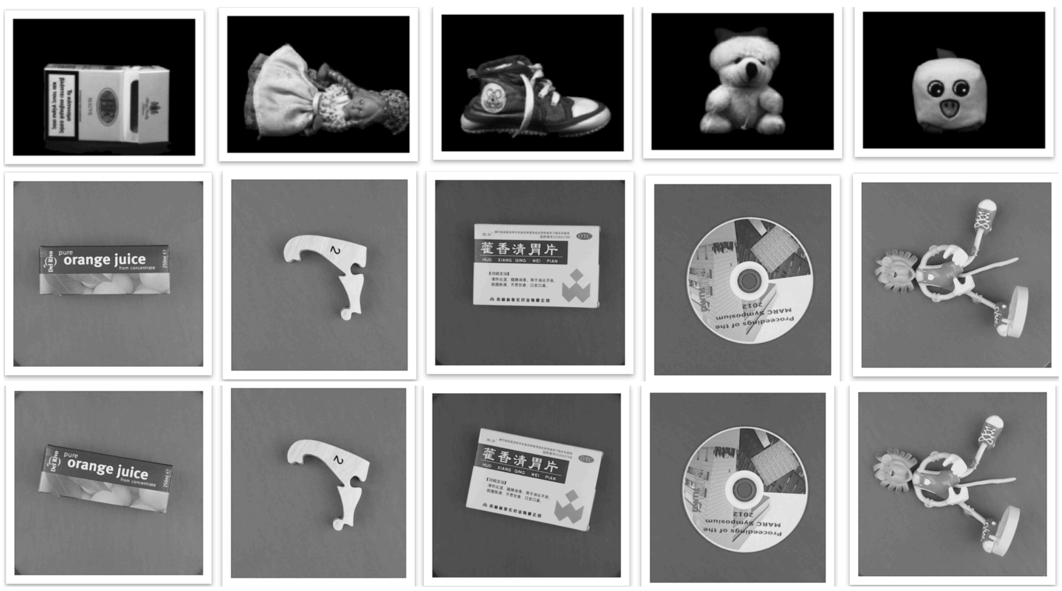


Figure 3.4: Used data for evaluation: The first row shows the 5 reference images in *Date1*. The second row shows the 5 reference images in *Date2*, and the third row illustrates the test images containing the corresponding objects with 10 degrees physical rotation in the real world.

features of these key-points are computed and compared between the reference images and the corresponding test images with rotation issue. Both RecHoG and HexHoG descriptors with four levels are evaluated in this system. In order to achieve rotation invariance, they are both generated by first rotating the feature receptive field according to the local dominant orientation which is computed by SIFT method. The matching score and the correct matching rate are both computed to evaluate the rotation invariance and distinctiveness of the proposed descriptors. The matching score is defined to measure the similarity between the two matched descriptors by means of computing the dot product. The correct matching rate is defined as similar as the recognition rate in BRIEF [Calonder et al., 2010], which can be described as: The number of detected key-points in the reference image is N. For each corresponding key-point in the test image, it is matched to all the N key-points in the reference image. If the most matched feature is from the correct corresponding key-point, this can be defined as a correct match. Then the correct matching rate is defined as S/N, where S is the number of correct match. Accordingly, for each matching pair between a test image and the corresponding reference image, a correct matching rate and an average of the matching scores for all the matched descriptors are computed.

Performance

The results of the mean matching score (rotation invariance) and the mean correct matching rate (distinctiveness) over 5 different objects are illustrated as a function of rotation degree

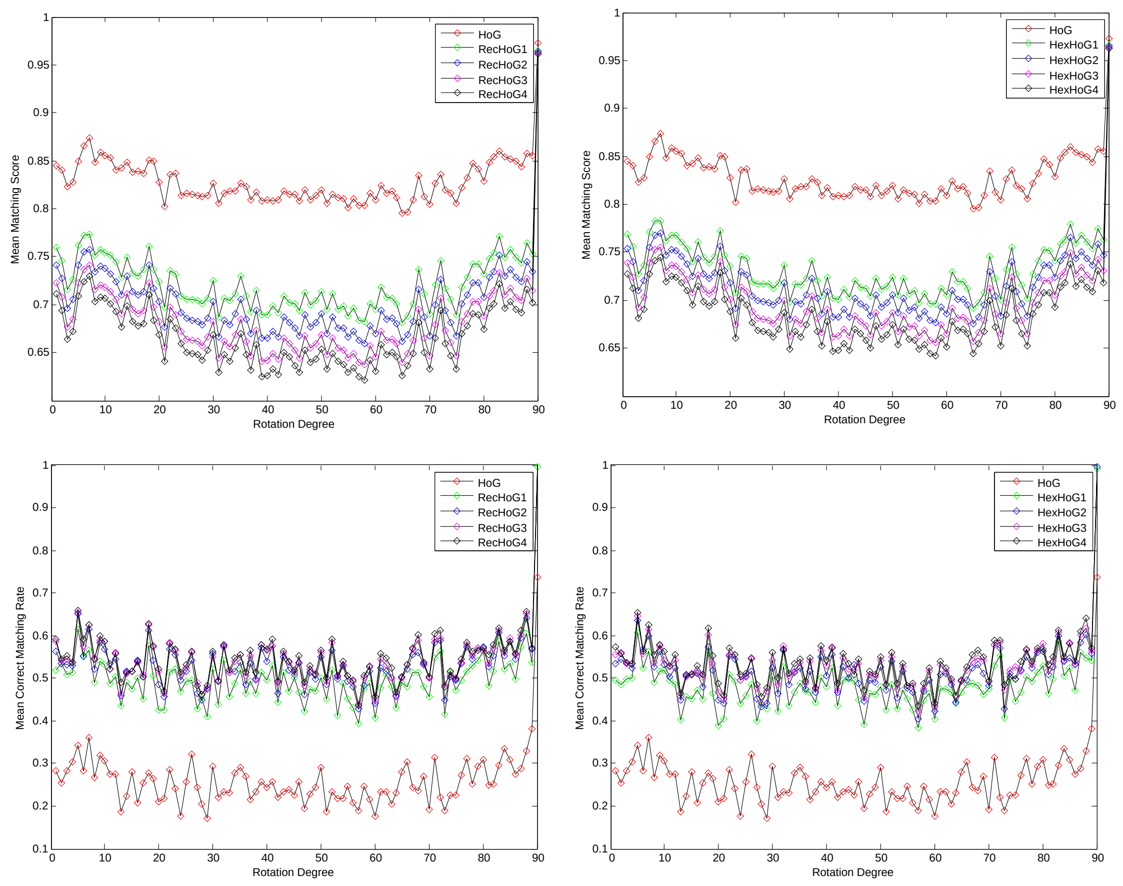


Figure 3.5: The first row shows the results of mean matching score (rotation invariance) for different levels of RecHoG and HexHoG as a function of rotation degree based on *Data1*. The second row shows the results of mean correct matching rate (distinctiveness) for different levels of RecHoG and HexHoG as a function of rotation degree based on *Data1*. Each subfigure illustrates the comparison among four different levels of the hierarchical descriptor.

in Figure 3.5 and Figure 3.6, based on *Data1* and *Data2* respectively. For each figure, subfigures in the first column show the performances of RecHoG descriptor, and subfigures in the second column show the performances of HexHoG descriptor. Moreover, each subfigure illustrates the performances of four levels of the herarchical descriptor. For comparison, the performance by using the single HoG descriptor (level 0 of the hierarchy) is also given. As shown in Figure 3.5 and Figure 3.6, the consistent performances are illustrated for both RecHoG and HexHoG. In detail, subfigures in the first row of each figure indicate that the mean matching score decreases with the hierarchical level increasing, while the mean correct matching rate increases with the hierarchical level increasing as indicated in subfigures of the second row. Moreover, for the sake of clarity, Figure 3.7 and Figure 3.8 (displayed at the end of this chapter) are produced to show the performance difference of each single level between RecHoG and HexHoG, where Figure 3.7 is based on *Data1* and Figure 3.8 is based on *Data2*. From Figure 3.7 and Figure 3.8 we can see, the consistent performances are illustrated, and for each single level, RecHoG performs better than HexHoG in terms of

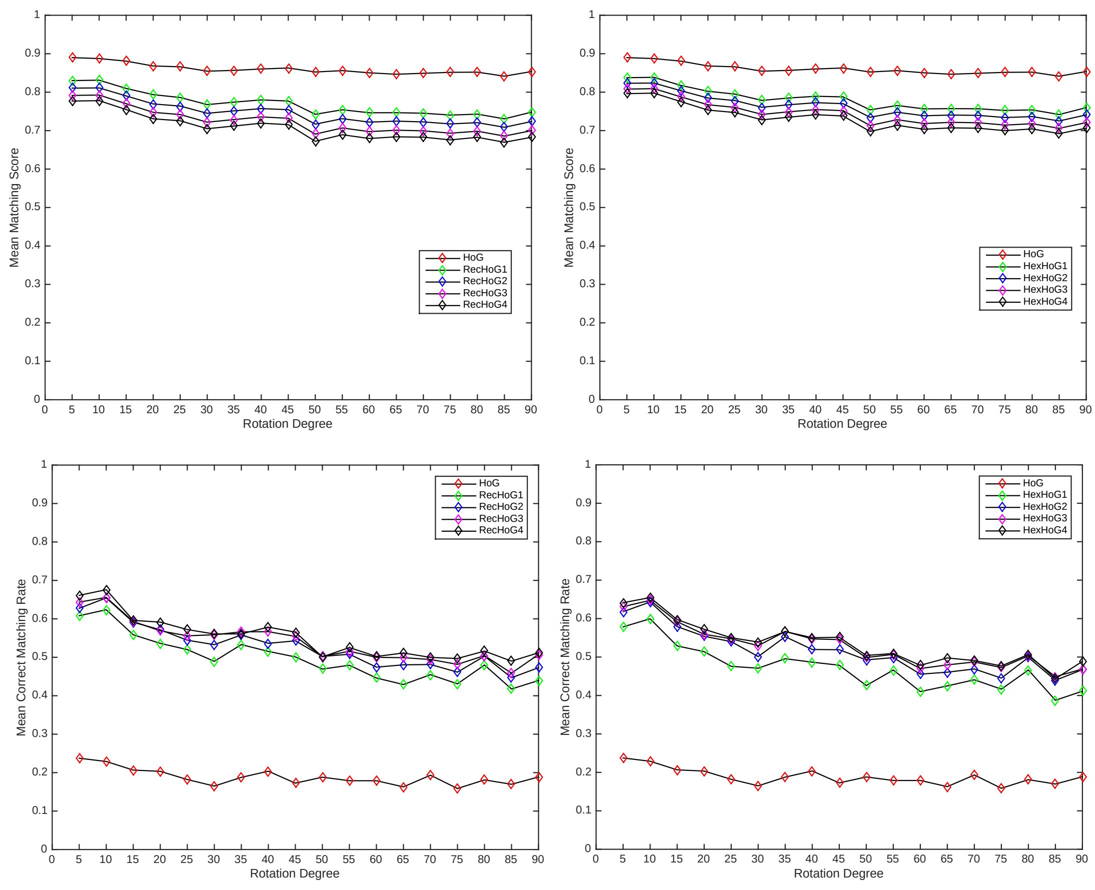


Figure 3.6: The first row shows the results of mean matching score (rotation invariance) for different levels of RecHoG and HexHoG as a function of rotation degree based on *Data2*. The second row shows the results of mean correct matching rate (distinctiveness) for different levels of RecHoG and HexHoG as a function of rotation degree based on *Data2*. Each subfigure illustrates the comparison among four different levels of the hierarchical descriptor.

distinctiveness, while it performs worse than HexHoG in terms of rotation invariance.

To further clarify the performance difference between RecHoG and HexHoG as shown in Figure 3.5 to Figure 3.8, Table 3.1 to Table 3.4 are used to indicate the MEAN value and standard deviation (SD) of the mean matching score and mean correct matching rate for different levels of the hierarchical descriptor, where the MEAN value and SD are calculated over all the rotated degrees of the test images. In detail, Table 3.1 and Table 3.2 show the MEAN value and SD for all the four levels of the hierarchical descriptors respectively based on *Data1*. From Table 3.1 we can see, with the hierarchical level increasing, the MEAN values of the mean matching scores for both RecHoG and HexHoG (i.e. Rec_score and Hex_score) decrease while the MEAN values of the mean correct matching rate for both RecHoG and HexHoG (i.e. Rec_rate and Hex_rate) increase. Furthermore, it is also observed that RecHoG has higher mean correct matching rate but lower mean matching score than HexHoG in each single level, except Level 0 (i.e. basic HoG descriptor) which is the basic component of both RecHoG and HexHoG. The SD values in Table 3.2 demonstrate

Table 3.1: Mean value over the rotation degrees of both mean matching score and mean correct matching rate for each level of the hierarchical descriptor based on *Data1*.

MEAN	Level0	Level1	Level2	Level3	Level4
Rec_score	0.8280	0.7231	0.7021	0.6816	0.6678
Hex_score	0.8280	0.7337	0.7168	0.6992	0.6865
Rec_rate	0.2596	0.4997	0.5345	0.5443	0.5494
Hex_rate	0.2596	0.4836	0.5177	0.5310	0.5382

Table 3.2: Standard deviation (SD) over the rotation degrees of both mean matching score and mean correct matching rate for each level of the hierarchical descriptor based on *Data1*.

SD	Level0	Level1	Level2	Level3	Level4
Rec_score	0.0240	0.0357	0.0383	0.0407	0.0423
Hex_score	0.0240	0.0348	0.0366	0.0387	0.0402
Rec_rate	0.0668	0.0696	0.0667	0.0656	0.0657
Hex_rate	0.0668	0.0698	0.0676	0.0663	0.0658

that, for each single level of the hierarchical descriptor, the performances are relatively constant (close to the MEAN) when the rotation degree changes from 1 to 90, which means the proposed descriptors are robust to rotation changes. Likewise, Table 3.3 and Table 3.4 show the similar performance trends as Table 3.1 and Table 3.2, but they are based on *Data 2*.

To test the statistical significance of the performance difference between RecHoG and HexHoG, Wilcoxon signed-rank test [Siegel, 1956] is employed for this experiment. The sample numbers in *Data 1* and *Data 2* are 450 and 90, respectively, which are the numbers of image matching pairs with rotation issue. The test results of both *Data 1* and *Data 2* reject the null hypothesis and accept the alternative hypothesis that the performance difference between RecHoG and HexHoG is statistically significant with the significance level 0.05.

Table 3.3: Mean value over the rotation degrees of both mean matching score and mean correct matching rate for each level of the hierarchical descriptor based on *Data2*.

MEAN	Level0	Level1	Level2	Level3	Level4
Rec_score	0.8600	0.7692	0.7467	0.7242	0.7076
Hex_score	0.8600	0.7749	0.7623	0.7440	0.7302
Rec_rate	0.1882	0.4963	0.5272	0.5404	0.5509
Hex_rate	0.1882	0.4710	0.5141	0.5279	0.5344

Timings of computing RecHoG and HexHoG descriptors are recorded on a computer running Ubuntu 14.04 (64-bit), with an Intel Core i5 3.10 GHz processor. All the experiments are

Table 3.4: Standard deviation (SD) over the rotation degrees of both mean matching score and mean correct matching rate for each level of the hierarchical descriptor based on *Data2*.

SD	Level0	Level1	Level2	Level3	Level4
Rec_score	0.0137	0.0309	0.0320	0.0337	0.0346
Hex_score	0.0137	0.0296	0.0307	0.0323	0.0332
Rec_rate	0.0215	0.0595	0.0575	0.0541	0.0554
Hex_rate	0.0215	0.0577	0.0581	0.0567	0.0571

implemented in Matlab. The computing time of a single descriptor is given by the average time of computing 50 descriptors, and the results are averaged over 10 runs. Table 3.5 clearly shows that each level of HexHoG is much faster to compute than that of RecHoG respectively, especially when going for higher levels of the hierarchy.

Table 3.5: Computing time for a single descriptor

Timing [ms]	Level1	Level2	Level3	Level4
$RecHoG$	1.6	15.4	138	1330
$HexHoG$	1.2	8.6	60	418

From the performances evaluated based on different datasets: *Data1* and *Data2*, which comprise test images generated in different mechanisms, we can clearly see that, for both RecHoG and HexHoG, the higher level descriptors achieve better feature distinctiveness but less rotation invariance than the lower level descriptors. The reason of this is that the higher level descriptors are extracted from bigger regions which include more distinctive information than the lower level ones. However, the region size should be controlled in a range because larger regions may also bring more noise interruption. Moreover, HexHoG has better rotation invariance but less distinctiveness than RecHoG, and the better distinctiveness of RecHoG is due to more sampling positions for construction, which sacrifices the computing efficiency.

3.4 Summary and Discussion

This chapter introduces an initial investigation of hierarchical feature descriptors based on rectangular grouping structure and hexagonal grouping structure. Both of the structures are constructed in a recursive way once the single patch size has been decided. The HoG descriptor of a single patch is employed to combine with the hierarchical grouping structures, which generates two different descriptors RecHoG and HexHoG. To evaluate the proposed descriptors, a local feature matching system is designed based on two different datasets. Then the mean matching score and the mean correct matching rate are computed to evaluate rotation invariance and distinctiveness of the proposed descriptors.

The evaluation results show that, by sacrificing the computing efficiency with sampling more positions of the local region of the object, RecHoG can achieve a slightly better distinctiveness but worse rotation invariance than HexHoG. In addition, with the hierarchical level going up, the distinctiveness increases gradually while the computing efficiency decreases dramatically. Though level 4 features achieve the best distinctiveness, they do not have impressive improvement compared to level 3 features for both RecHoG and HexHoG. Therefore, considering the tradeoff between the distinctiveness and computing efficiency, for the later work of investigating the hierarchical descriptor, only 3 levels of the descriptor will be considered.

All in all, the work in this chapter is considered as the first step of exploring hierarchical features based on the hexagonal grouping structure. The evaluation results illustrate that the hierarchical feature descriptors extracted based on hexagonal grouping structure provide potential benefits to represent local features. The rotation invariance of RecHoG and HexHoG is achieved by pre-rotating the feature receptive field to align with the local dominant orientation based on SIFT method. In order to improve the distinctiveness and computing efficiency, in the next chapter, a new rotation-invariant hierarchical descriptor will be proposed without pre-rotating the local field but with utilising the rotational symmetry of the hexagonal structure. In addition, a 2D object localisation system will also be proposed based on matching the proposed descriptors of the edge contour points.

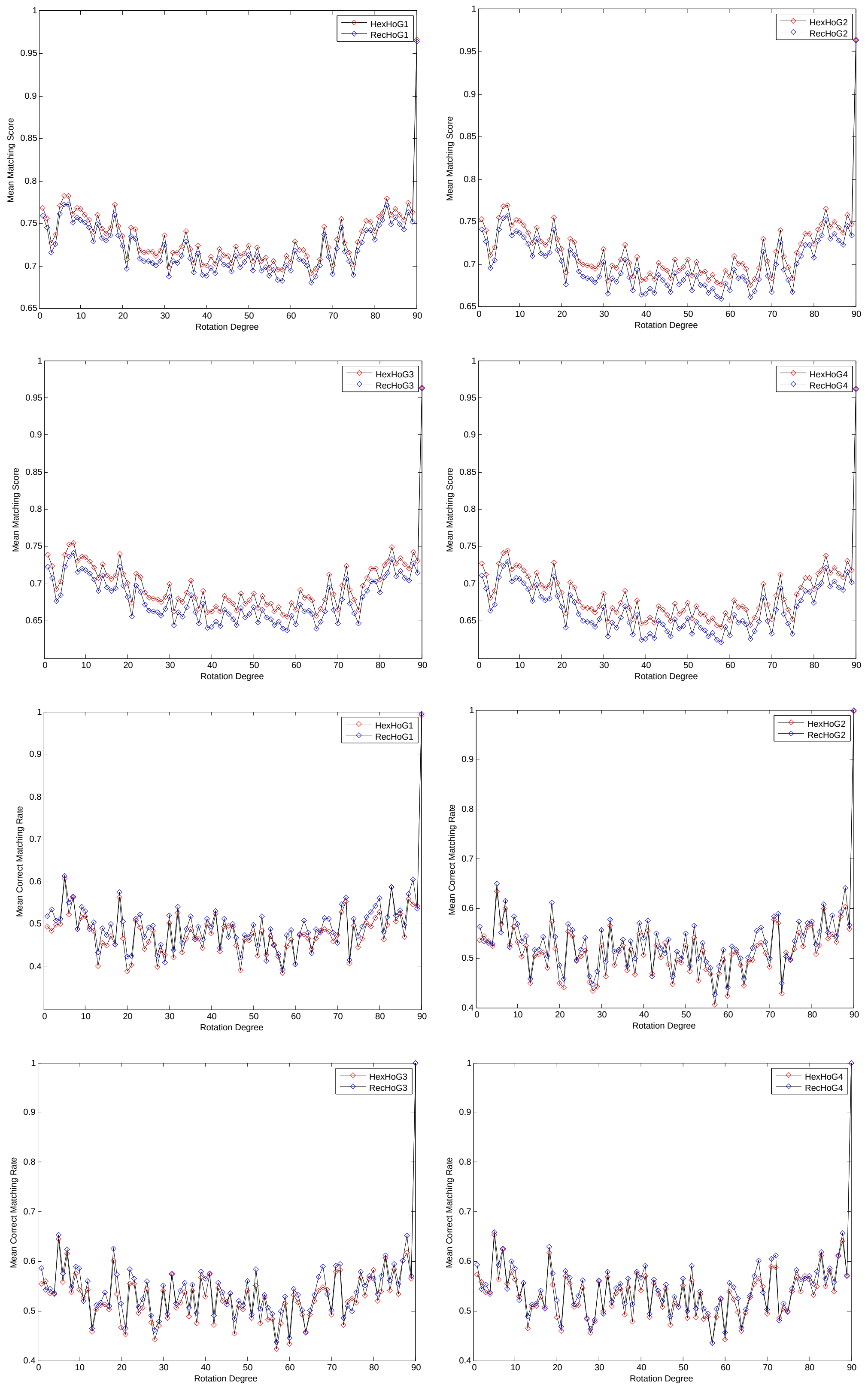


Figure 3.7: The comparison between RecHoG and HexHoG in each single level based on *Data1*. The first four subfigures illustrate results of the mean matching rate (rotation invariance). The last four subfigures show results of the mean correct matching rate (distinctiveness).

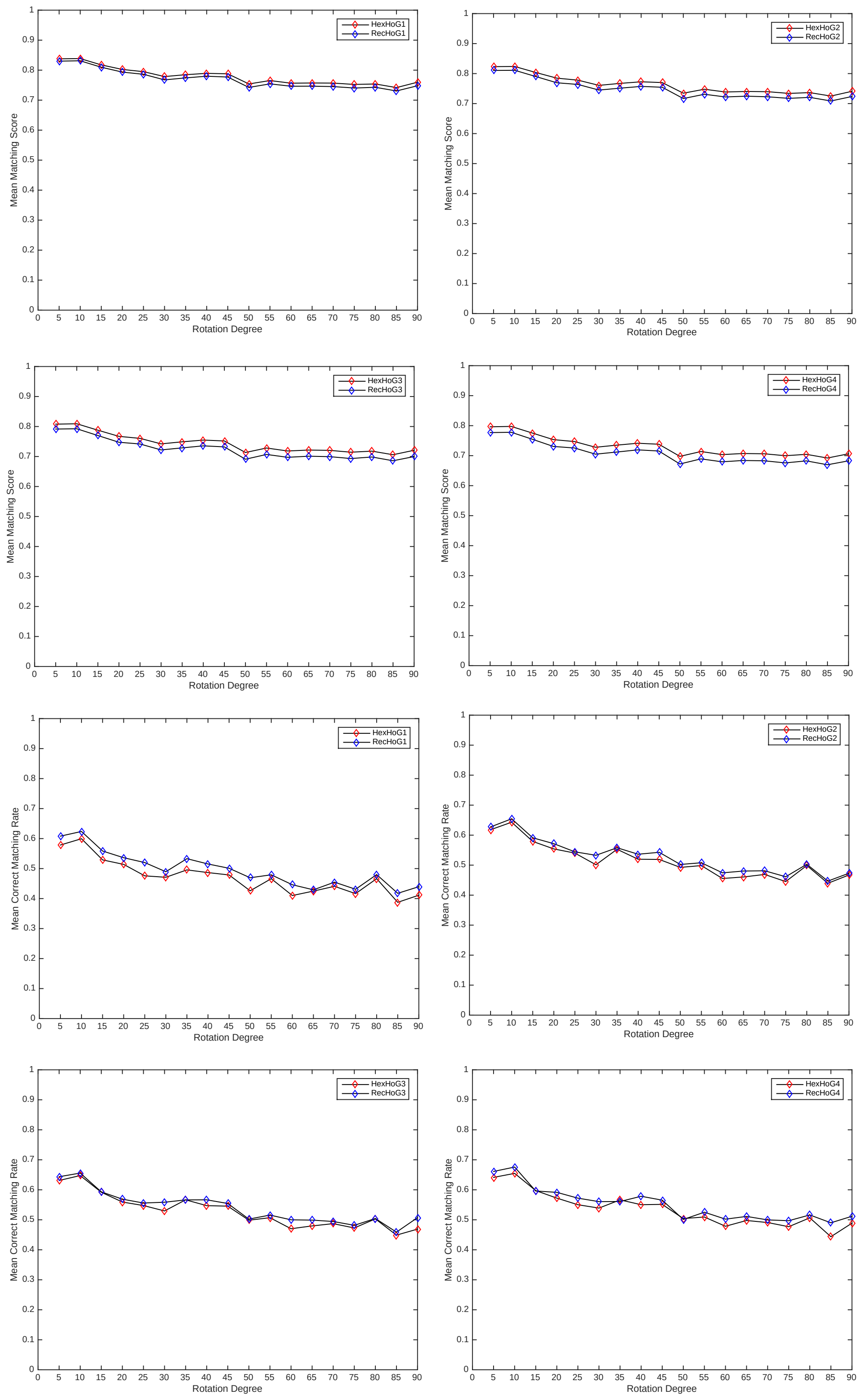


Figure 3.8: The comparison between RecHoG and HexHoG in each single level based on *Data2*. The first four subfigures illustrate results of the mean matching rate (rotation invariance). The last four subfigures show results of the mean correct matching rate (distinctiveness).

Chapter 4

2D Object Localisation Based on Matching HexSHoG Descriptors

The previous chapter has given an initial investigation of a local hierarchical descriptor HexHoG, which is generated based on the hexagonal grouping structure. With less sampling for the local feature descriptor, HexHoG has shown rather stable properties compared to RecHoG, which is generated based on the rectangular grouping structure. Therefore, this chapter introduces the work keeping on researching about the hexagonal grouping structure, but with new strategies to construct the hierarchical descriptor and achieve the rotation invariance without pre-rotating the feature receptive field. Moreover, a 2D object localisation system is proposed by matching edge contour points. The new proposed descriptor in this chapter is applied to represent the features of edge contour points in the 2D object localisation system.

4.1 Motivation and Objective

An ideal feature descriptor is desired to have rotation invariance to deal with the rotation changes of the target object. Though the rotated target could find a correct match through matching features to the template dataset that comprises features without rotation invariance but extracted from all the rotated versions of the target object, this is a very expensive process in terms of time consumption. In recent decades, the conventional method to achieve rotation invariance of local features is to align the feature receptive field with its local dominant orientation before generating the feature descriptor. Therefore, the feature receptive field needs to be rotated and resampled. The total time consumption of computing rotation-invariant features will increase with the feature number growing. To avoid the process of rotating and resampling the feature receptive filed, a new method of generating rotation-invariant

HexHoG descriptor is proposed in this chapter based on the advantage of the rotational symmetry of a hexagon.

It is an essential ability to detect and localise an object of interest from an image containing a cluttered background in computer vision. Most of the published work for object detection could indicate the approximate position of the detected object in a bounding box [Shotton et al., 2005; Schlecht and Ommer, 2011]. Some researchers have proposed methods of object detection with localisation based on matching edge contour features [Zhu et al., 2008; Ferrari et al., 2010]. Object contours are the essential information of the object shape boundary, which is a significant feature to illustrate the accurate object position. However, contour detection is a challenging research topic because the object boundary is always encompassed by a cluttered background which results in the difficulty of detecting object contours accurately out from the background. For instance, when a detected contour comprises edge points partly from the object and partly from the image background, matching the edge contour features will lead to either miss part of the object edges or define the edges from the background wrongly as part of the object. To reduce this phenomenon, an alternative method to localise the detected object is through matching edge contour points rather than edge contours. Therefore, in this chapter, a 2D object localisation system is investigated through matching the edge contour points by employing the new proposed descriptor.

4.2 Rotation-Invariant Feature Extraction

4.2.1 Rotation-Invariant Hexagonal Structure

In order to remove the process of rotating feature receptive field but still achieve rotation invariance for the feature descriptor, a new hexagonal structure is proposed in this section. For comparison, the previous structure of HexHoG proposed in Chapter 3 is given in Figure 4.1 (a), which indicates that there are always two vertexes of the hexagon lying on the direction of the X-axis after rotating the receptive field according to the dominant orientation. Differently, without rotating the receptive field, the vertexes of the new proposed hexagonal structure are sampled by making the local dominant orientation as the canonical reference rather than the X-axis, which means there are always two vertexes lying on the direction of the dominant orientation as shown in Figure 4.1 (b).

Moreover, another process is employed to provide HexHoG more rotation invariance: in the standard HoG descriptor of a single patch, the highest bin, which exhibits the dominant gradient orientation of the local patch, is barrel-shifted to the head position of the histogram, which means the histogram starts with the frequency value of the dominant orientation as shown in Figure 4.2. Therefore, the rotation invariance of a single HoG is achieved by simply

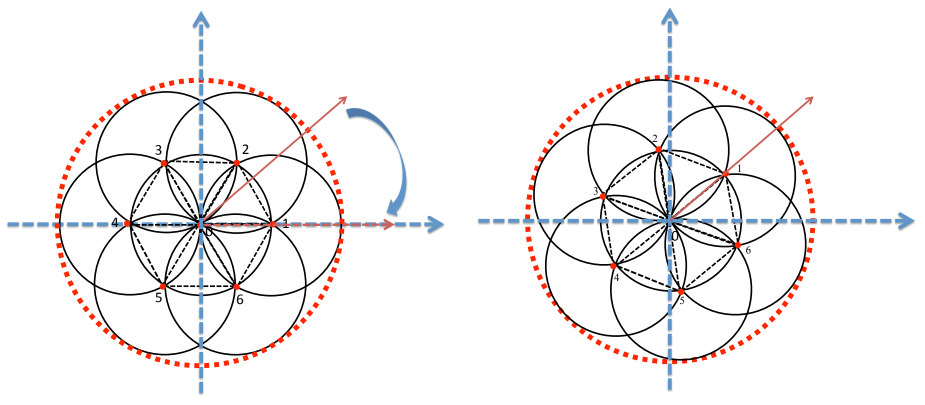


Figure 4.1: The red arrow is the dominant orientation of the area covered by the red dot circle. In (a), the figure shows the conventional method to achieve rotation invariance of the local descriptor, which is achieved by first rotating the local area to make the dominant orientation aligned with the X-coordinate then sampling two of the hexagonal vertexes along the horizontal line. In (b), there is no rotation of the local area, but the hexagonal vertexes are sampled directly according to the dominant orientation with two vertexes aligning with the dominant orientation.

shifting the histogram. This orientation normalised new HoG is termed as SHoG while the standard HoG descriptor with rotation invariance achieved in the way by first rotating the local patch is termed as DHoG. The pseudocode for constructing SHoG is described in Algorithm 5. As a consequence, the new hexagonal grouping structure combined with SHoG is adopted to generate the new rotation-invariant descriptor termed as HexSHoG.

Algorithm 5 SHoG Construction

```
$HoG$: histogram of oriented gradients
$Num\_Bin$: number of bins in HoG
$p$: index to the peak value of HoG bin
$SHoG \leftarrow [HoG(p : Num\_Bin), HoG(1 : p - 1)]$
```

4.2.2 HexSHoG Generation

To construct the sampling structure of HexSHoG descriptor, the parameters of the structure need to be first defined. The radius of the circular region defining the region size covered by a single HoG and the distance between the neighbouring vertexes of the hexagon could be freely parameterised. These parameters control the overlapping size between the HoG fields of each grouping, which influences the degree of rotation invariance and the distinctiveness for HexSHoG descriptor. The exploration of how the parameters affect the performance of hexagon-based descriptors is not discussed in this chapter but will be deeply investigated

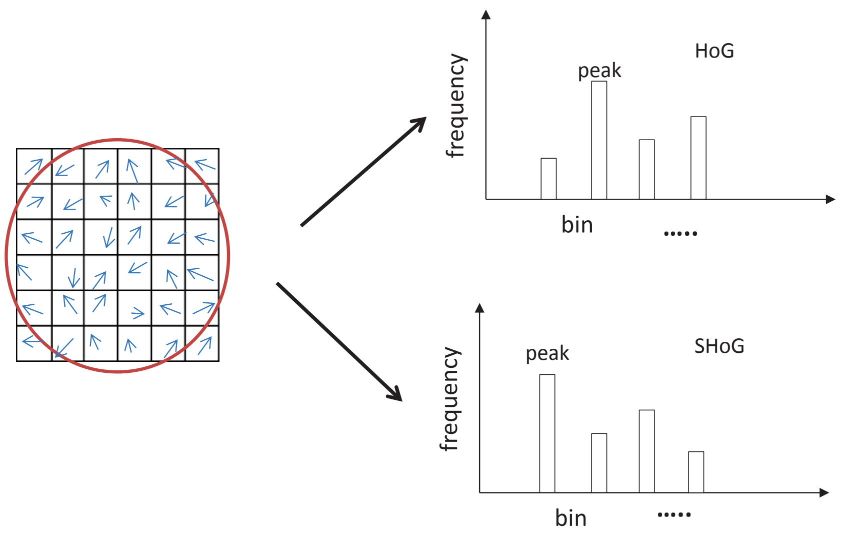


Figure 4.2: Local patch represented by HoG and SHoG. The upper histogram is the standard HoG, and the lower histogram is SHoG. SHoG is achieved by shifting the peak value of HoG to the head position of the histogram.

in Chapter 6. However, HexSHoG descriptor will use the same parameters as for HexHoG introduced in the previous chapter.

After determining where to sample the 6 vertexes of the hexagon structure according to the local dominant orientation, 7 SHoGs will be computed in the fields centred at the hexagonal centre and vertexes. Then the first level of HexSHoG can be generated by concatenating $SHoG_i(i{=}0,1,2,...6)$ by first assigning the central SHoG descriptor to the head of the grouped descriptor, followed by the SHoG descriptor which is aligned to the dominant orientation. All of the remaining $SHoG_i$ descriptors will be subsequently concatenated in counterclockwise order. The feature is then normalised by its magnitude to achieve robustness to illumination changes. This process is described as in Algorithm 6.

The steps of generating the first level of HexSHoG are applied recursively to compute higher level HexSHoG descriptors with the same concatenating mechanism. Accordingly, the second level descriptor *HexSHoG2* is generated based on the 7 first level descriptor *HexSHoG1s* centred on the red points as shown in Figure 4.3. For clarity, the edge length of the first level hexagon is enlarged to make it easier to illustrate. The ordering mechanism used to concatenate *SHoG* for *HexSHoG1* is consistent with the above description. The blue arrow in Figure 4.3 shows the computed dominant orientation of the central region covered by $HexSHoG1_0$ descriptor. The dominant orientations of all the other 6 $HexSHoG1_i$ (i=1,2,...,6) descriptors are all defined by the blue arrow for convenience. The right graph in Figure 4.3 illustrates the generation of a *HexSHoG1* descriptor for the red dashed region. The pseudocode to generate higher level *HexSHoG* is given in Algorithm 7.

Algorithm 6 Level1 Construction of HexSHoG

```
$< Px_0, Py_0 >$: HexSHoG centre, i.e., key-point location
$r$: hexagon edge length
$\theta$: computed local dominant orientation for the region centred at $< Px_0, Py_0 >$
$< Px_i, Py_i >$ $(i \leftarrow 1...6)$: six vertex positions of the hexagon centred at $< Px_0, Py_0 >$
$ts \leftarrow pi/3$
**for** $i \leftarrow 1 : 6$ **do**
    $tv \leftarrow (i-1)ts + \theta$
    $Py_i \leftarrow Py_0 + r\sin(tv)$
    $Px_i \leftarrow Px_0 + r\cos(tv)$
**end for**
**for** $i \leftarrow 0 : 6$ **do**
    Construct $SHoG_i$ at point $< Px_i, Py_i >$
**end for**
$HexSHoG1 \leftarrow Normalize(SHoG_0, SHoG_1, ..., SHoG_6)$
```

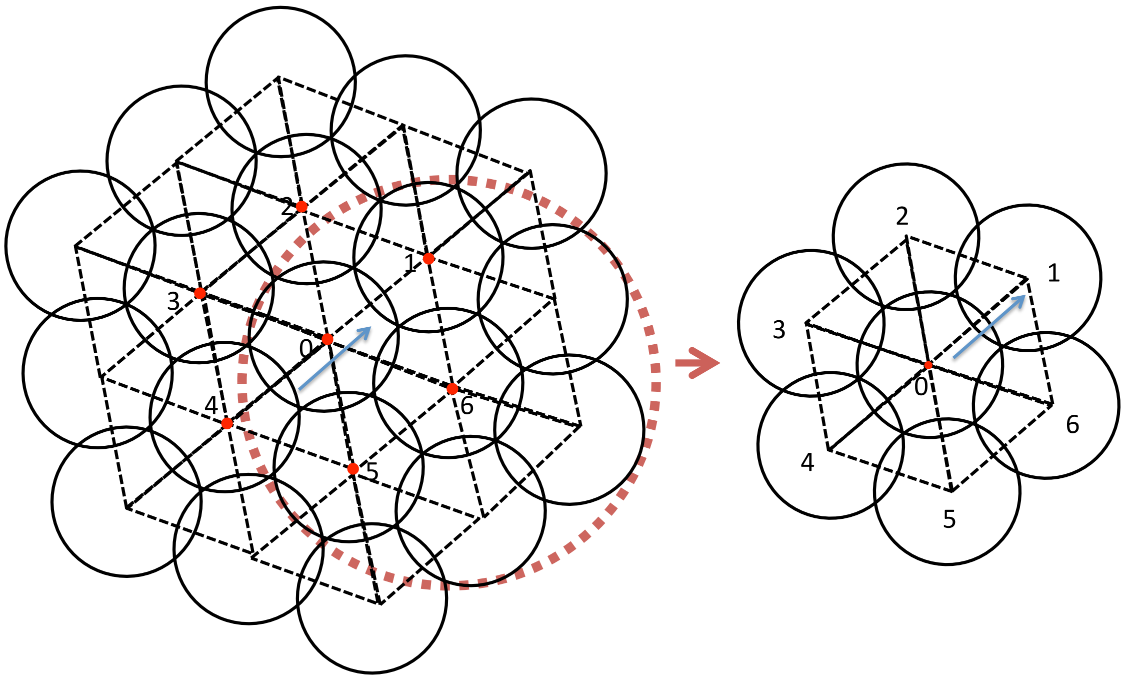


Figure 4.3: The left figure illustrates the second level structure of HexSHoG . The right figure shows the structure of the red dot covered region in the left figure.

Algorithm 7 Generation of Higher Level HexSHoG Descriptors

```
$< Px_i, Py_i >$ $(i \leftarrow 0, 1...6)$: hexagon centre and vertex positions
$r$: hexagon edge length
$\theta_0$: computed local dominant orientation for the region centred at $< Px_0, Py_0 >$
$\theta_i (i \leftarrow 1...6)$: defined local dominant orientation for the region centred at $< Px_i, Py_i >$
$ts \leftarrow pi/3$
for $i \leftarrow 1 : 6$ do
    $tv \leftarrow (i-1)ts + \theta_0$
    $Py_i \leftarrow Py_0 + r\sin(tv)$
    $Px_i \leftarrow Px_0 + r\cos(tv)$
    $\theta_i \leftarrow \theta_0$
end for
for $i \leftarrow 0 : 6$ do
    Generate un-normalised first level descriptor U-HexSHoG1$_i$ centred at $< Px_i, Py_i >$
    with local dominant orientation $\theta_i$
end for
$HexSHoG2_0 \leftarrow Normalize($U-HexSHoG1$_0$, U-HexSHoG1$_1$, ..., U-HexSHoG1$_6)$
$HexSHoGLevel_{(t+1)0} \leftarrow Normalize($U-HexSHoG Level$_{(t)0}$, U-HexSHoG Level$_{(t)1}$, ...,
U-HexSHoG Level$_{(t)6})$
```

4.2.3 Rotation-Invariant Performance of HexSHoG

To evaluate the rotation invariance of HexSHoG constructed based on SHoG, the local feature matching experiments are implemented based on the similar validation method introduced in the previous chapter. 20 different images containing different objects are randomly selected from ALOI as a reference set, and each of them is rotated by 1° per step in range $[1, 90]^\circ$ to generate a test set, respectively. The descriptor for each keypoint in each test image is computed and compared to the descriptor of the corresponding point in each reference image. The dot product of the corresponding descriptors and the average dot product over 20 different test images are computed as a function of degree of in-plane rotation. The experiment result is illustrated in Figure 4.4.

This experimental hypothesis is that SHoG can achieve better performance than DHoG in terms of rotation invariance. Besides, as demonstrated in the previous chapter, the higher level descriptor of HexSHoG has weaker rotation invariance than the lower level descriptor. As the first graph shows in Figure 4.4, comparing the rotation-invariant performances of DHoG, SHoG and the standard HoG, DHoG and SHoG achieve good rotation invariance with SHoG having better performance than DHoG, while the performance of the standard HoG declines quickly after about 20 degrees of rotation. This indicates that SHoG achieves better

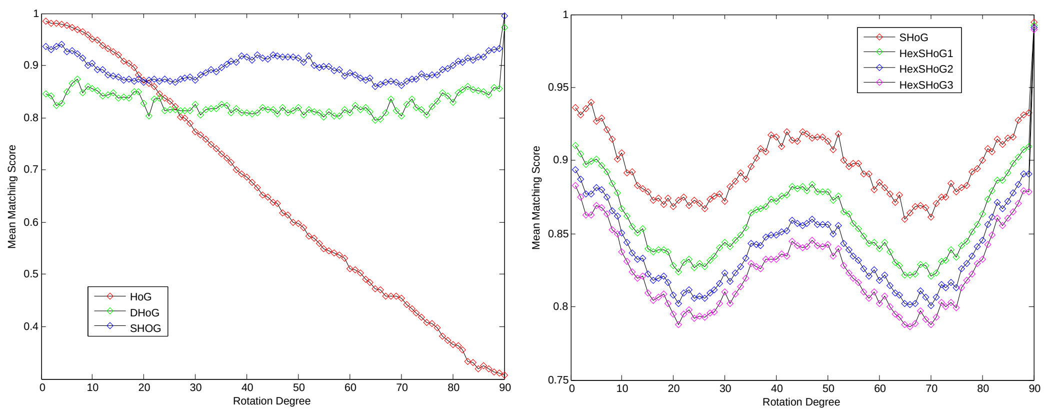


Figure 4.4: The results of mean matching score (rotation invariance) as a function of rotation degree. The left figure shows the comparison among HoG, DHoG and SHoG. The right figure illustrates the comparison among four different levels of HexSHoG. SHoG denotes level 0 of HexSHoG. The scale of Y axis is different between the two figures.

rotation invariance by means of simply shifting the histogram bins of the standard HoG. The performance of HexSHoG generated based on SHoG is also illustrated in the second graph of Figure 4.4. In the experiment, 8 histogram bins are used to record the relative frequency of local gradient orientations. This explains the periodic performance observed every 45° for all the proposed features in Figure 4.4. Although the rotation invariance of the feature is getting weaker as the grouping level increases, *HexSHoG3* can still produce a mean matching dot product mostly greater than 0.8 over all the rotation degrees, which defines the matching threshold employed later to evaluate the 2D object localisation system.

4.3 2D Object Localisation

In this section, a 2D object location system is proposed, where the overview structure of the system is summarised in Figure 4.5. Specifically, the system is mainly divided into two sections: object pose estimate and edge labelling. Accordingly, this edge labelling process relies on a correct prior object detection result and the quality of the pose estimate obtained during the prior detection process. In this system, the above-proposed descriptor HexSHoG is applied for the edge labelling process to localise the edge contours of an object within a cluttered background.

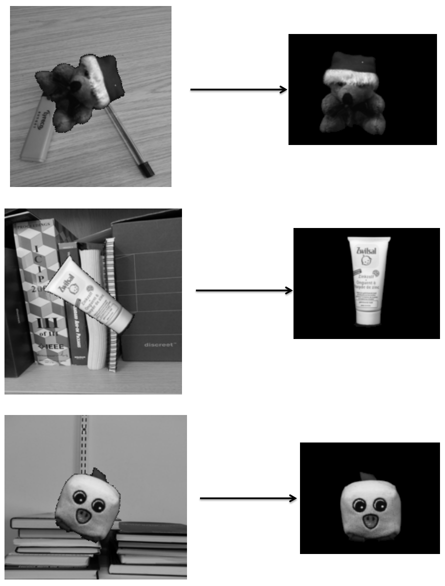


Figure 4.6: Each test image is directly matched to the corresponding reference image for object detection and pose estimate.

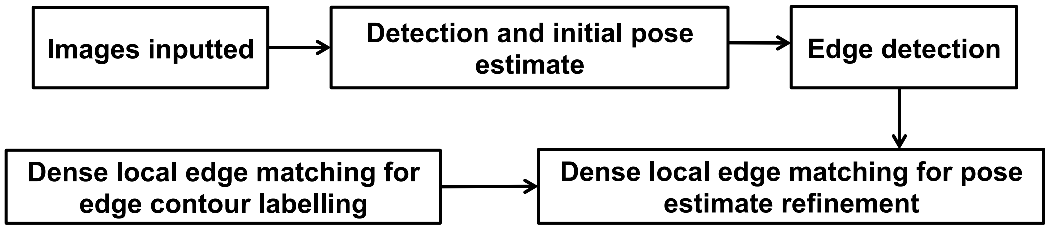
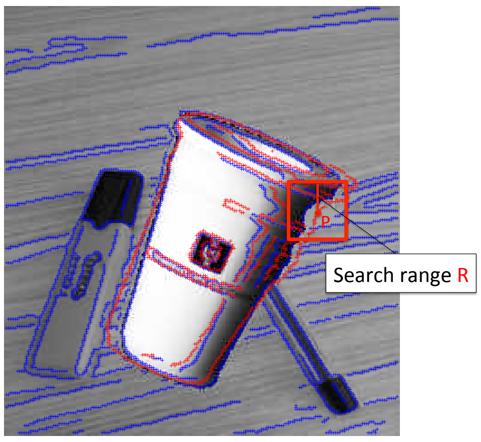

Images inputted
Detection and initial pose estimate
Edge detection
Dense local edge matching for pose estimate refinement
Dense local edge matching for edge contour labelling

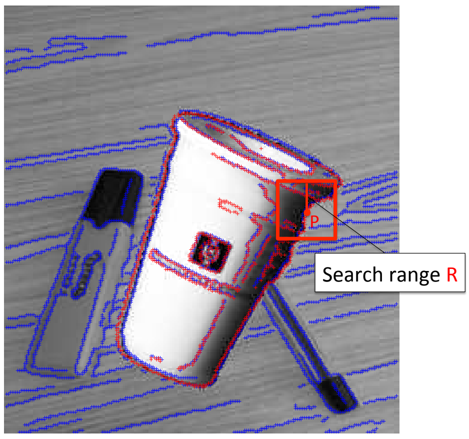
Figure 4.5: The brief summary of the 2D object localisation system.

4.3.1 Detection and Pose Estimate

Since SIFT [Lowe, 2004] is an established benchmark for state-of-the-art performances in object detection, it is adopted in this experiment for the purposes of object detection and initial pose estimate. However, the main work of this thesis focuses on the pose estimate after the correct detection, so that there is no need to add extra obstacles to test the detection performance by SIFT. Therefore, the sparse SIFT descriptors extracted from a test image are directly matched to the corresponding SIFT descriptors extracted from a reference image, as illustrated in Figure 4.6. Then the matched pairs are grouped and filtered using the algorithms of Generalised Hough Transform (GHT) [Ballard, 1981] and Random Sample Consensus (RANSAC) [Fischler and Bolles, 1981] to compute an initial pose estimate for the successfully detected object in the test image. In order to define whether it is possible to achieve more accurate pose estimate, a further refinement step is proposed by means of dense local HexSHoG matching based on edge points, described as follows:



(a)



(b)

Figure 4.7: Pose refinement and edge contour labelling processes. The two graphs show the same test image with its edge contours labelled in blue, and the red edges are the projected edges of the reference object according to the estimated pose information. (a) is the local searching process for pose refinement after initial pose estimate; (b) is the local searching process for edge contour labelling after pose refinement.

1. Compute edge label (edgel) maps for both the test image and the corresponding reference image using the Canny edge detector;
2. Project the edgels of the reference image into the test image edgel map according to the initial pose estimate;
3. Find the set of the test image edgels that neighbour each projected edgel from the reference image edgel map, within a constrained square search area for each projected edgel as in Figure 4.7 (a);
4. From the set of neighbouring test-image edgels, find the test image edgel with the best matching for each projected edge point by comparing their HexSHoG features, computed from the inputted images;
5. Re-estimate the pose transformation from the reference image to the test image, based on all the matched edgel-pair correspondences obtained above.

The constrained search area reduces false-positive matches between background clutter edgels and the reference object's edgels, while the use of edgel-located feature matching provides many more feature correspondences than corner-based features alone, especially when the reference object inherently lacks corners, i.e., contains mainly smooth edge contours. The matched edgel-pair correspondence is valid only when the corresponding feature similarity is bigger than the matching threshold and the chessboard distance between the corresponding positions is less than the threshold R as in Figure 4.7.

4.3.2 Edge Labelling for Object Localisation

To achieve object localisation, the edge labelling process is implemented based on the pose refinement result. The estimated edgel positions in the test image can be found by projecting the reference edgels using the refined pose estimate. As Figure 4.7 (b) shows, the search process, constrained to a limited range in X and Y axis with the chessboard distance R, is repeated to match between the edgel positions projected according to the pose refinement result and the edgels in the test edgel map. The refinement process of pose estimate could minimise the local search range for the edgel matching. The edgels within the test image, which match to the projected reference image edgels, are then labelled in the test image as being contour edgels. There might be some breaks between labelled edges, so an edge connectivity post-process is executed as follows: if an edgel in the test image is labelled as a contour edgel, all connected edgels (comprising 8 nearest neighbours) will be likewise labelled. This process is then repeated for each newly labeled contour edgel. The iteration of this is implemented in order to label those edgels which comprise the object's edge contours and thereby potentially capture the shape of the detected object in terms of observed edgels. Then the detected objects could be more completely localised with edge contours labelled in the test image.

4.4 Performance Evaluation

In this section, the performance evaluation of the proposed object localisation system by using HexSHoG descriptor is demonstrated. The experimental hypothesises are: the pose estimate performance can be improved after doing the pose refinement process by using HexSHoG descriptor, and through the HexSHoG descriptor matching of edge contour points, most of the object contours can be labeled. Before implementing the steps of the object localisation system as described above, the parameters of constructing a HexSHoG structure are defined as the same for HexHoG introduced in the previous chapter: the Gaussian weighted patch size is set to be 7 pixels wide for SHoG, and edge length of the sampling hexagon is set to be 3 pixels, which results in the HexSHoG grouping structure has half patch size overlapped between each neighboured SHoG pairs. The validation data employed in this work are obtained from the Amsterdam Library of Object Images (ALOI) [Geusebroek et al., 2005] which contains 1000 different objects. All of the 1000 different images from the ALOI database are employed as reference images. Each of these reference images is randomly rotated in plane and embedded into 5 different backgrounds to generate a test image set comprising 5000 images. Figure 4.8 illustrates examples of the image sets described above.

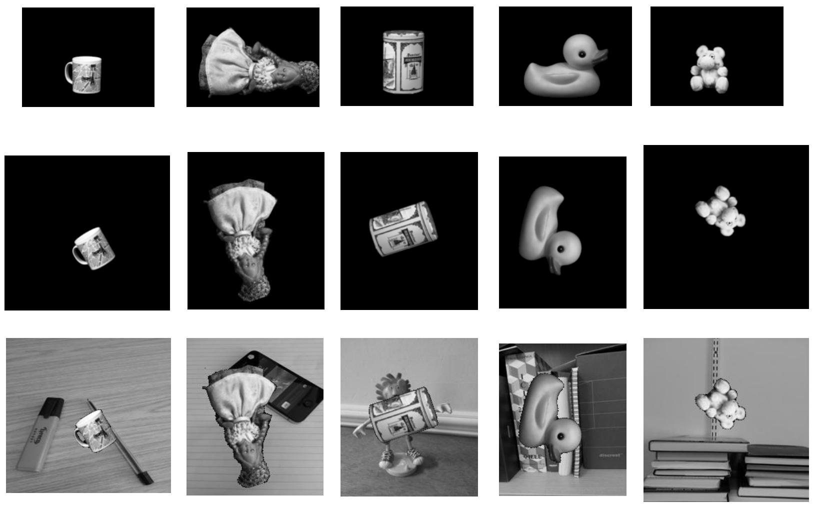


Figure 4.8: Examples of the data used in the application experiment: The top row shows the reference images from ALOI. The middle row shows the corresponding in-plane rotated versions. The bottom row shows the test images generated by embedding the rotated images into different backgrounds.

Before the pose estimate refinement is implemented, an experiment for selecting which level of HexSHoG descriptor to use is designed. Since the descriptor is expected to give the most accurate pose localisation, the experiment is designed as follows to determine the displacement error resulting from local edge matching based on HexSHoG descriptor, with the hypothesis that the third level descriptor *HexSHoG3* can give the most accurate pose estimate. 20 different images from ALOI are randomly selected as a reference set and then rotated incrementally in range $[1, 90]^\circ$ to form a test set. Therefore, for each edgel in each test image generated, the corresponding edgel in the original reference image is known. The HexSHoG feature for each reference image edgel is then computed and compared to the features computed within a local neighbourhood of 2 pixels, centred on the corresponding test image edgel. The best dot product match is found and its position will then be recorded. The spatial distance between the matched position and the corresponding true feature position is computed for each reference edgel as the displacement error for local matching. Thereafter, the average error is computed over the 20 reference images and the displacement error distribution is obtained as a function of rotation degree for 3 different levels of HexSHoG feature, as shown in Figure 4.9. It is observed in Figure 4.9 that the level 3 HexSHoG descriptor *HexSHoG3* gives the smallest displacement error for most of the applied rotations, which suggests that *HexSHoG3* can provide better localisation performance compared to other less grouped features for the purpose of pose estimate refinement.

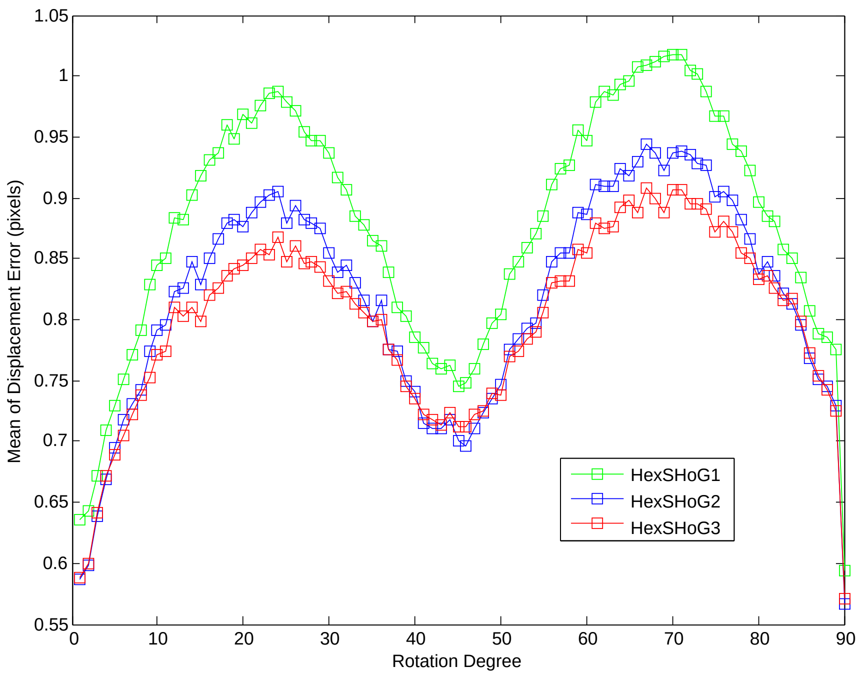


Figure 4.9: Displacement error of local HexSHoG matching as a function of rotation degree.

4.4.1 Pose Refinement Performance

In order to evaluate how well the proposed pose estimate refinement method performs, for each test image, the ground-truth information specifying the rotation and translation used to embed the reference object pixels into a background image is recorded. Therefore, the precise location of edge contours of the reference object in the test image can be known. According to the pose estimate information, the estimated object edgel positions are first obtained by projecting the reference edgels into the test image. Then for each reference edgel, the distance between its estimated position and its ground-truth position is computed to give its pose estimate error. Finally, the mean and standard deviation (SD) of matched point displacement error for the test set are used to evaluate the pose estimate performance.

In this experiment, 2892 images are successfully detected out of 5000 test images for object detection and initial pose estimate. Those images containing object of interest that has less distinctive corners and is not sufficiently distinguishable from the background are failed to be detected. A selection of failed examples is shown in Figure 4.10. Consequently, the pose estimate refinement and edge labelling processes are only applied to test image examples containing successfully detected object instances.

The pose refinement performance is investigated with respect to the constrained search bounds, by varying the X, Y search range from $\pm$ 1 to $\pm$ 10 pixels, and the refined pose estimate error is computed accordingly. By comparing the refined pose estimate error to the initial pose estimate error for each test image, the number of test images which exhibit an improvement in pose estimate due to the refinement process could be determined. Both the

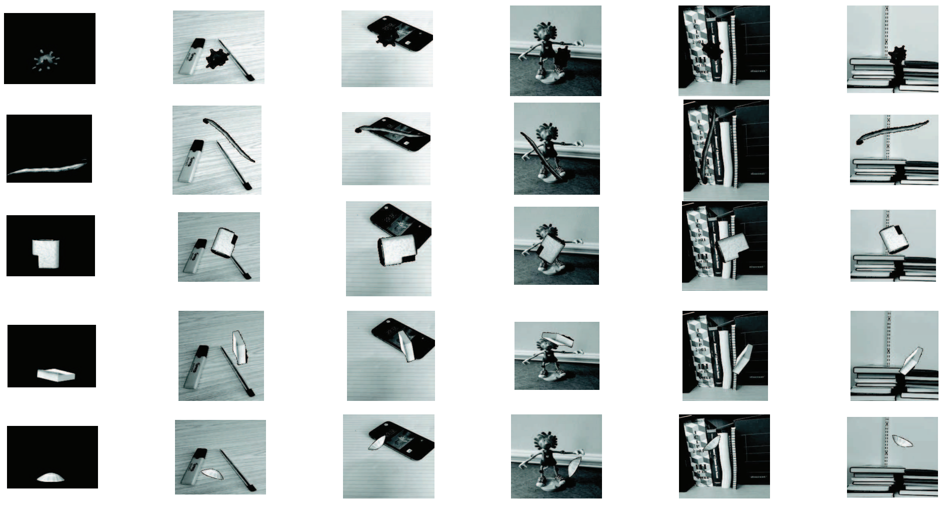


Figure 4.10: Failed examples of detection by SIFT: The first column shows the reference objects. The remaining columns show the test objects with backgrounds.

mean pixel error and the SD for those successfully detected test images of initial estimations, and refined estimations, are computed as shown in Table 4.1. Accordingly, the pose refinement performance as a function of the search range, with error bars showing 95% confidence interval for the mean pose error, is also illustrated in Figure 4.11. It is clearly shown in Figure 4.11 that, there is no confidence interval overlap between the initial pose error by SIFT and the refined pose error by HexSHoG3, which indicates that the difference between the initial pose error and the refined pose error with different search ranges is statistically significant.

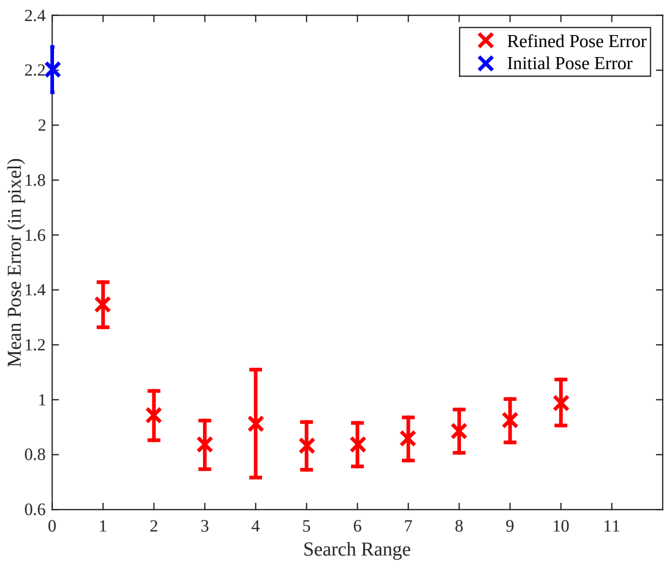


Figure 4.11: The mean pose error results for the refined pose estimation are illustrated as a function of pixel search range. The initial pose error given by SIFT with no search process is also displayed in this figure for clear comparison. The error bar shows 95% confidence interval for the mean pose error.

In Table 4.1, the mean pose error and SD (in pixel units) of the refined pose estimate are pre-

Table 4.1: Pose estimate refinement performance as a function of pixel search range and the initial pose estimate performance by SIFT

Search Range ±	Mean	SD	Number of Improved Images	Improved Ratio
1	1.35	2.68	2807	97.06
2	0.94	2.93	2771	95.82
3	0.84	2.89	2757	95.33
4	0.91	6.42	2738	94.67
5	0.83	2.84	2720	94.05
6	0.84	2.59	2696	93.22
7	0.86	2.56	2669	92.29
8	0.89	2.57	2637	91.18
9	0.92	2.58	2607	90.15
10	0.99	2.73	2560	88.52
Initial Pose Estimate	2.20	2.69		

sented with different search bounds. The number of images having improved pose estimate after refinement and their corresponding improved ratio of the pose refinement test set are also presented. From Table 4.1 we can see, when the search range for edgel matching is constrained to less than 10 pixels, the *HexSHoG3*-based pose estimator achieves an improvement in over 90% of the initially successful object detections. The search range in the region of ±5 or ±6 pixels can make the least pose estimate error (as a reference point for comparison, the *HexSHoG3* used for matching is 28 pixels in diameter), and the pose estimate refinement process improves the mean pose estimate error for the test dataset by approximately a factor of 2. The SD of the pose error over the test set is kept around 2.5 pixels, except when the search range is ±4. However, the number of images having improvement in pose estimate declines monotonically with search range. Therefore, there is a tradeoff between the degree of pose refinement and the number of object detections that are improved. Accordingly, for subsequent edge contour labelling experiments, reported below, a search range of ±5 pixels is chosen. A selection of examples of pose estimate refinement is illustrated in Figure 4.12.

4.4.2 Edge Labelling Performance

Finally, dense local edge matching is re-applied in order to label directly the edgels detected within the test image that comprises the contour edgels of the object of interest, according to the recovered pose estimate (using a ± 5 pixel search range). Figure 4.13 shows some examples of the labelling results obtained by matching three different grouping levels of HexSHoG descriptor. When the image background is too cluttered, or the object outer boundary is not easily distinguished from the background, missed object boundary detections can result and background edgels close to the object can be mis-labelled as belonging to the object. It is observed in Figure 4.13 that each level of HexSHoG descriptor produces slightly different labelings, making it difficult to conclude which level of HexSHoG feature grouping gives best results. It would appear that the distraction from the background is greater for larger higher level descriptors (which straddle both the object boundary and the background to a greater degree) while the lower level descriptors have less reliability. Therefore, in the later chapter, a more completed and sophisticated edge contour localisation system is proposed with additional processes and employing 3 different levels of the hierarchical descriptor together to contribute to the edge labelling.

4.5 Summary and Discussion

There are two main contributions in this chapter: one is the new rotation-invariant hierarchical descriptor HexSHoG, the other is the proposed system for 2D object localisation based on pose estimate refinement and edge contour labelling. Based on the recursive HexHoG structure introduced in the previous chapter, the sampling positions of the hexagonal structure are extracted according to the local dominant orientation by taking advantage of the rotational symmetry of a hexagon, which directly improves the time efficiency for the local descriptor construction with rotation invariance. Moreover, SHoG is also employed to improve the rotation invariance and has better performance than DHoG. As a consequent, the hierarchical descriptor constructed based on the new structured sampling mechanism and SHoG is proposed as HexSHoG. It is then applied on feature extraction of edge contour positions to use the object edge information for pose estimate refinement. A complete pose refinement system is introduced with statistical validation results, based on which the edge contour labelling process for 2D object localisation is implemented and some initial test results are illustrated.

The pose refinement results illustrate that matching HexSHoG features, which are based only on appearance information computed at edgel locations, has the potential to improve the performance of object pose estimate by approximately a factor of 2. By improving the accuracy of the pose estimate process, it is then possible to project contours from the reference image into the test image and annotate the location of a detected object with sufficient accuracy for many practical tasks such as grasping in robotics. However, improved pose estimate also improves the search constraints required to match test image edge contours directly, to allow HexSHoG matching to offer the possibility of recovering the actual edgel labels detected in the test image that correspond to contour edgels in the reference image, as described above.

However, the proposed method is computationally expensive because of the recursive steps for constructing HexSHoG descriptor based on all the dense extracted edges, especially when applied in a large dataset. Therefore, a faster method to construct hierarchical descriptors is required for more efficient applications. In the next chapter, in order to improve the time efficiency but without reducing the performance quality, a new hierarchical descriptor constructed based on binary coding mechanism is proposed and termed as HexBinary. Moreover, a complete 2D object localisation system will be introduced with statistical evaluation results by employing the new proposed HexBinary descriptor extracted at edge contour points.

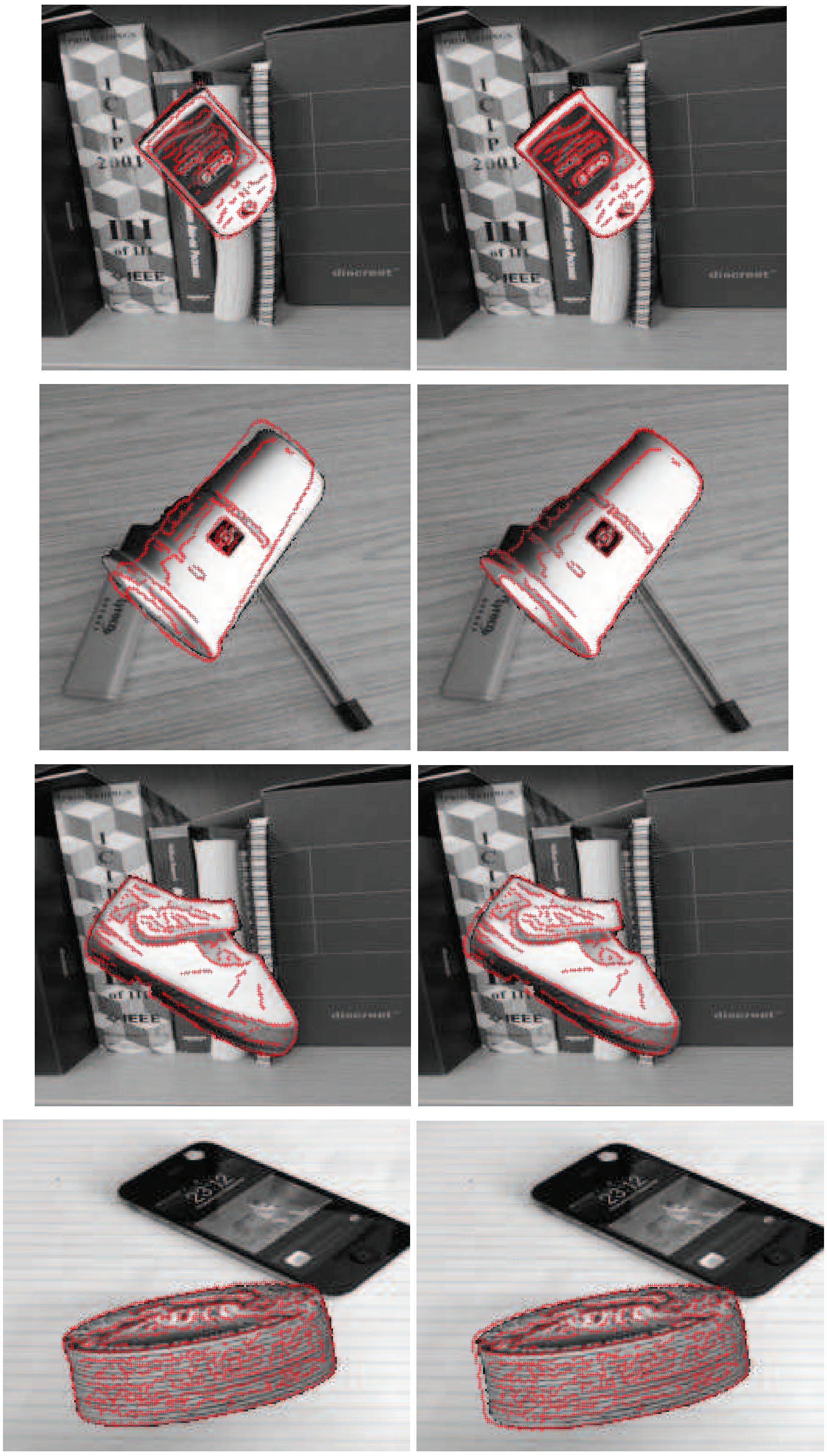


Figure 4.12: Edge projection from the reference objects into the test images according to the initial and refined pose estimate, shown in left and right columns, respectively. The first two rows show the examples with improvement after pose estimate refinement. The last two rows show the examples that failed to achieve improvement after pose estimate refinement.

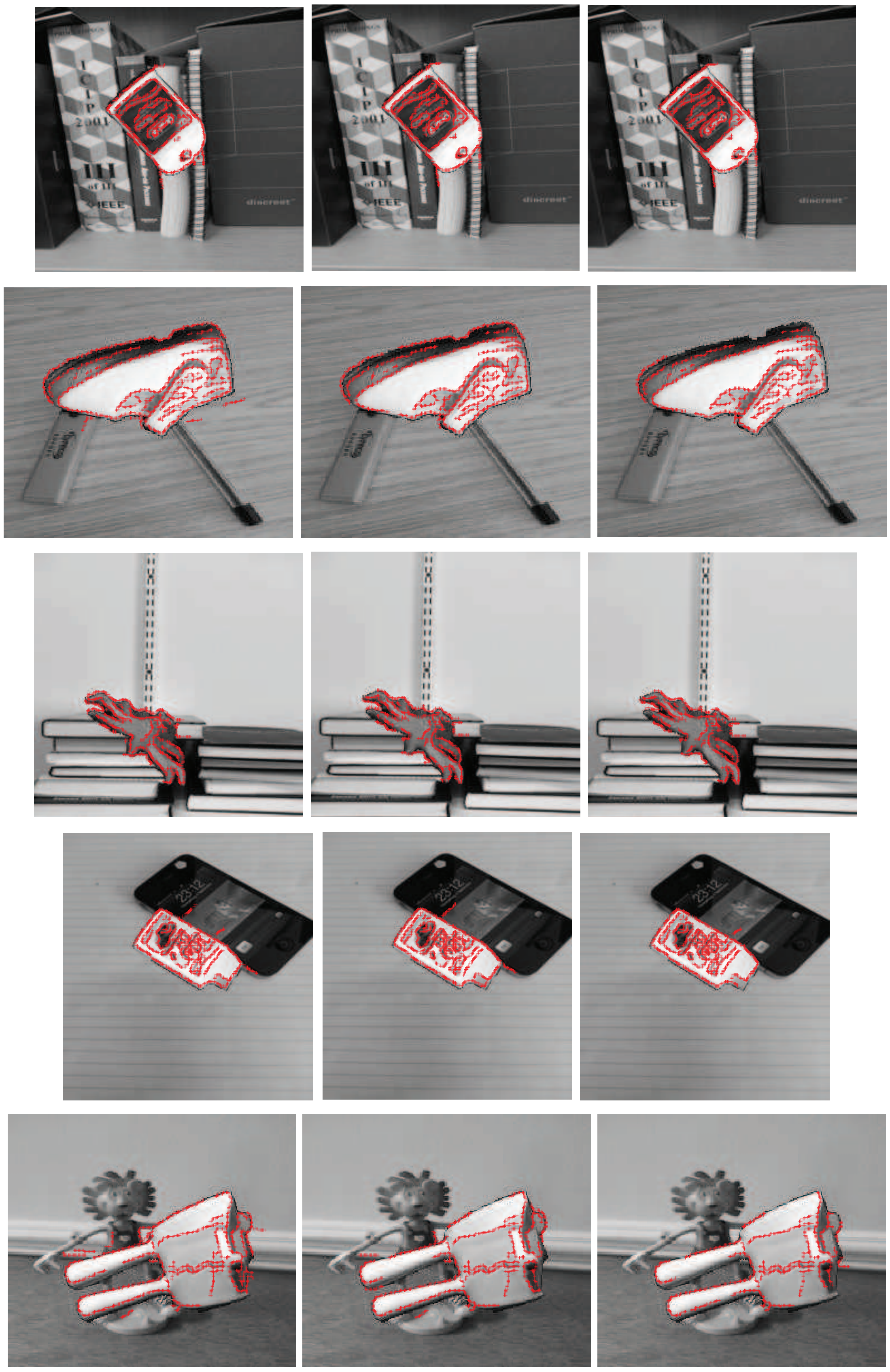


Figure 4.13: Object edge contour labelling results: from the first column to the third column, edge labelling results by using *HexSHoG1*, *HexSHoG2*, *HexSHoG3*, are displayed respectively.

Chapter 5

2D Object Localisation Based on Matching HexBinary Descriptors

In the previous chapter, HexSHoG is proposed as a hierarchical feature descriptor, which achieves the rotation-invariant property by taking advantage of the hexagonal structure. It shows high capability to represent local features well through the application of describing the edge point features for the object localisation system. However, the computation cost can be a big issue when it is asked to serve the real-time/on-line application. Therefore, based on the hexagonal grouping structure of HexSHoG, a new hierarchical descriptor HexBinary is introduced in this chapter with faster computing and lower storage requirement by employing the binary coding mechanism. In addition, HexBinary is also applied to represent the edge features in the 2D object localisation system, and achieves superior performance of pose refinement to HexSHoG. Moreover, a complete system for 2D object localisation with labelling edge contour points is also introduced in this chapter, with complementary statistical analysis of the edge labelling performance based on the system proposed in the previous chapter.

5.1 Motivation and Objective

Local feature extraction has been explored extensively in the field of computer vision. The dominant local feature descriptors can be generally divided into two categories: floating-point descriptors and binary descriptors, as reviewed in Chapter 2. The floating-point descriptor is normally generated by a histogram of quantised gradient orientations based on a local patch, such as the descriptors introduced in [Lowe, 2004; Dalal and Triggs, 2005; Mikolajczyk and Schmid, 2005; Brown et al., 2011]. SIFT [Lowe, 2004] has served as a standard benchmark for evaluating local feature performance because of its good performance

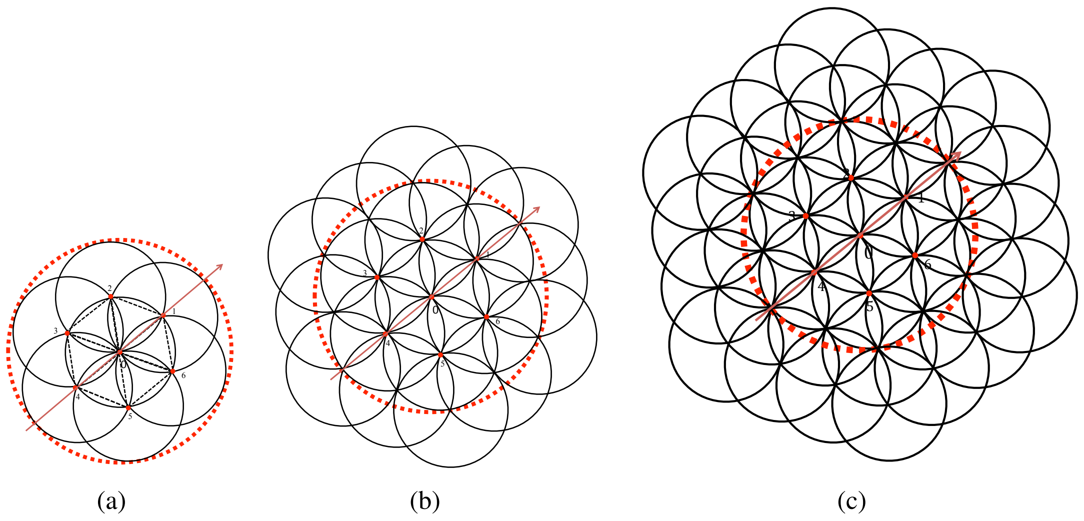


Figure 5.1: The hierarchically hexagonal structure for HexSHoG: (a) The first level sampling pattern of the hexagonal structure (the red arrow shows the dominant orientation of the red dotted region); (b) The second level sampling pattern of the hexagonal structure; (c) The third level sampling pattern of the hexagonal structure. Each black circle represents the Gaussian kernel size which could be freely parameterised.

in many computer vision applications and widespread availability. To improve the time efficiency, PCA-SIFT [Ke and Sukthankar, 2004] was proposed to achieve faster matching by reducing the descriptor dimensions from 128 to 36 elements via Principal Components Analysis. SURF [Bay et al., 2006] was developed to improve the computational efficiency of the feature extraction process by employing Haar-wavelet filters, efficiently implemented by means of integral images. However, the total computational cost and the storage requirement of these floating-point descriptors can still make issues when they are used for real-time / on-line applications. Therefore, Binary String (BS) descriptors have recently been devised and intensively explored since their first inception, BRIEF [Calonder et al., 2010], appeared. BS descriptors are generated by computing pairwise intensity comparisons within a local sampling pattern and have been demonstrated to exhibit much lower computational cost and storage requirements without decreasing the discriminability than floating-point descriptors. Due to this, different BS descriptors were proposed with different sampling structures of the point-pairs for intensity comparison [Rublee et al., 2011; Leutenegger et al., 2011; Alahi et al., 2012]. The difference between these BS descriptors is computed efficiently by means of their Hamming distance.

The hexagonal grouping structure of HexSHoG, which was proposed in the previous chapter, is illustrated in Figure 5.1. SHoG descriptors computed in the regions covered by black circles are concatenated together to generate HexSHoG, which leads to expensive computation with the hierarchical level increasing. Inspired from the improvement of local feature extraction execution rates and matching performance rates achieved by using BS descriptors, in this chapter, BS descriptor is employed to substitute SHoG to improve the time efficiency of computing the hierarchical descriptor based on the hexagonal structure of HexSHoG, which produces the new hierarchical descriptor HexBinary. In addition, a system for 2D object localisation, by means of fine-to-coarse local edge feature matching, is also presented in this chapter based on the system proposed in the previous chapter, with the qualitative and quantitative results analysed for pixel-level edge contour localisation using the ALOI dataset.

5.2 HexBinary Feature Extraction

Several different binary string descriptor sampling configurations [Calonder et al., 2010; Rublee et al., 2011; Leutenegger et al., 2011; Alahi et al., 2012] have been reported, ranging from regular symmetric to randomly sampled. The binary bit computed by comparing the sign of difference in the intensity pairs of sampling points in effect encodes the sign of the first order derivative. Intuitively, this encoding mechanism captures the relative local spatial configuration of light and dark in the local image region sampled by the descriptor. Furthermore, if the sign of the second order differences is computed, this would correspond to the local spatial configuration of the intensity gradients. In order to sample the descriptor efficiently, point-wise differences that correspond to approximations of the orthogonal first and second polar derivatives are utilised in this chapter. Therefore two different comparison schemes are proposed to construct HexBinary, one of which is first order and the other is second order. Moreover, two image pre-filtering methods are employed here: simple Gaussian low-pass filtering to suppress image noise and aliasing and Laplacian of Gaussians (LoG) isotropic filtering that captures second order gradient information, potentially useful for encoding boundary information.

5.2.1 Orientation Assignment

As shown in Figure 5.2, the original first level sampling configuration is located at a hexagon centre p_0 with the hexagon vertices p_1 and p_4 aligning with the X-axis. In order to make the descriptor invariant to rotation, the local dominant orientation will be firstly determined. The orientation based on the sample pairs defined between the hexagon centre point and the vertexes is computed essentially the same mechanism as utilised in BRISK [Leutenegger et al., 2011]. The image is first filtered by a Gaussian kernel with standard deviation σ, and the smoothed intensity values with respect to the hexagon centre and vertexes are $I_i(i{=}0,1,...6)$. Then the local gradient of the red dotted circle covered region in Figure 5.2 is computed

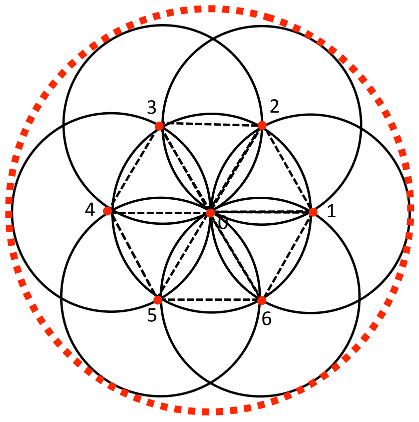


Figure 5.2: The original first level sampling pattern of the hexagonal structure. Two of the hexagonal vertexes are sampled on the direction of X-coordinate. Each black circle represents the Gaussian kernel size.

using:

$$g(\mathbf{x},\mathbf{y}) = \frac{1}{N}\sum_{p\in\mathcal{S}}(\mathbf{p}_j - \mathbf{p}_i)\cdot\frac{I_j - I_i}{\|\mathbf{p}_j - \mathbf{p}_i\|^2}, \tag{5.1}$$

where $\mathbf{p}$ is the position vector of the hexagon centre and vertexes, and N =12 is the number of pairs in set $\mathcal{S}$, where $\mathcal{S}$ is composed by two subsets of point pairs: the subset approximating the polar tangential derivatives comprises adjacent pairs of samples taken at the hexagon vertexes and subtracted in a counterclockwise direction $\mathcal{S}_1 = \{(p_6,p_1),(p_i,p_{i+1})(i\text{=}1,...5)\}$; the subset approximating the radial polar derivatives comprises the group of sample subtractions taken from the hexagon centre to each single vertex $\mathcal{S}_2 = \{(p_0,p_i)(i\text{=}1,...6)\}$. According to this computed local gradient, the dominant orientation is defined as $\theta = \mathbf{arctan2}(g(y), g(x))$.

5.2.2 First Order HexBinary

Given the local dominant orientation θ, the hexagon vertex points are resampled as Figure 5.1 (a) shows, based on which a new set $\mathcal{Q}$ that has the same sampling scheme and binary encoding method as in $\mathcal{S}$ is extracted. The 12 point pairs in $\mathcal{Q}$ are used to generate a 12 bit binary string, where each bit τ corresponds to:

$$\tau(I;i,j) = \begin{cases} 1 & \text{if } I_i < I_j \\ 0 & \text{otherwise.} \end{cases} \tag{5.2}$$

A 12 bit first level descriptor *HexBinary1* sampled at point p_0 is generated as above, and the second level descriptor *HexBinary2* is generated by concatenating the *HexBinary1*$_i$ descriptors sampled at positions $p_i(i\text{=}0,1,...6)$. The second level and third level hexagon structures are shown in Figure 5.1 (b) and (c), respectively. The mechanism for computing the hierarchical descriptor HexBinary is described in Algorithm 8, which is basically the same as for computing HexSHoG.

Algorithm 8 Generation of Hierarchical HexBinary Descriptor

```
$p_i(i \leftarrow 0, 1...6)$: the first level hexagon centre and vertex positions
$r$: hexagon edge length
$\theta_0$: computed local dominant orientation for the region centred at $p_0$
$\theta_i(i \leftarrow 1...6)$: defined local dominant orientation for the region centred at $p_i$
$ts \leftarrow pi/3$
for $i \leftarrow 1 : 6$ do
    $tv \leftarrow (i-1)ts + \theta_0$
    $p_{y(i)} \leftarrow p_{y(0)} + r\sin(tv)$
    $p_{x(i)} \leftarrow p_{x(0)} + r\cos(tv)$
    $\theta_i \leftarrow \theta_0$
end for
for $i \leftarrow 0 : 6$ do
    Generate the first level descriptor $\mathit{HexBinary1}_i$ centred at $p_i$ with local dominant orientation $\theta_i$
end for
$HexBinary2_0 \leftarrow \mathit{HexBinary1}_0, \mathit{HexBinary1}_1, ..., \mathit{HexBinary1}_6$
$HexBinaryLevel_{(t+1)0} \leftarrow \mathit{HexBinary\ Level}_{(t)0}, \mathit{HexBinary\ Level}_{(t)1}, ... ,$
$\mathit{HexBinary\ Level}_{(t)6}$
```

5.2.3 Second Order HexBinary

In order to differentiate the first order HexBinary descriptor and the second order HexBinary descriptor, we denote them as *FHexBinary* and *SHexBinary*, respectively. The process of constructing the *SHexBinary* descriptors follows that of constructing the *FHexBinary* descriptors, except that the pairs of first order intensity difference values are compared. In the first level of the hexagonal structure, a set of first order intensity differences is computed using pairs from set:

$\mathcal{S}' = \{(p_0,p_1), (p_4,p_0), (p_0,p_2), (p_5,p_0), (p_0,p_3), (p_6,p_0), (p_6,p_1), (p_i,p_{i+1})(i\text{=}1,...5)\}.$

Accordingly, a corresponding first order intensity difference value set $\mathcal{D}$ is attained, from which 9 pairs are selected to generate *SHexBinary* with each bit τ corresponding to :

$$\tau\left(D; i, j\right) = \begin{cases} 1 & \text{if } D_i < D_j \\ 0 & \text{otherwise,} \end{cases} \tag{5.3}$$

where$(\mathbf{D}_i, \mathbf{D}_j)$ is a spatially adjacent pair, e.g., $(\mathbf{D}_i = I_1 - I_0, \mathbf{D}_j = I_0 - I_4; \mathbf{D}_i = I_1 - I_6, \mathbf{D}_j = I_2 - I_1)$. The *SHexBinary* descriptor is recursively generated using the same construction scheme used to compute *FHexBinary*.

To encapsulate all the information sources including the first order and second order intensity derivative information together, *FHexBinary* and *SHexBinary* descriptors are concatenated to construct a new descriptor *CHexBinary* in each single level of the hierarchy, respectively.

5.3 Application to 2D Object Localisation

In Chapter 4, a 2D object localisation system was proposed with two main steps: pose estimate and edge labelling. HexSHoG descriptor was employed in the process of pose refinement and edge labelling. In this section, the details of how to apply the new proposed descriptor HexBinary in the 2D object localisation system are described. The performance evaluation compared to HexSHoG is also given in the following section.

5.3.1 Pose Refinement

In the 2D object localisation system, SIFT is applied to give an initial pose estimate of the detected object located in a cluttered background. HexBinary features of the object's edge contour points are then computed to refine the pose estimate, since the edge contour points anchor a large set of samples which define the structure of the object. These descriptors are used to specify the relative pose of the object as follows:

1. The edge labels (edgels) of the reference image (black background) and the test image (cluttered background) are firstly detected by the Canny edge detector, and a morphological operation is applied to the reference and test edge maps to remove isolated edgels. This process also eliminates some noise points and renders the detected reference object edgels more consistent with the detected test object edgels.

2. The reference edgels are projected into the test image based on the initial pose estimation. A small local area of the test image surrounding each reference edgel is then searched for the test edgel which has the best match to a (corresponding) reference edgel and which also exceeds a certain matching threshold.

A set of corresponding edgel pairs is generated based on the above steps and used to re-estimate the pose of the object by means of RANSAC.

5.3.2 Edge Labelling for Object Localisation

The main idea of the edge labelling step is the same as described in the previous chapter. However, to better utilise the HexBinary descriptor, some details need to be changed. When coming to the object boundary points, HexBinary might suffer more background disturbing than HexSHoG at the boundary points. Therefore, to label the edge points by HexBinary, the main steps are as follows:

1. For each black-background reference image *R1*, generate a second reference image *R2* with a white background (to allow the boundary descriptors to match over positive or negative background contrast phases). Detect edgels for both *R1* and *R2*, and transform the edgel positions into the test image according to the refined pose estimation;

2. Classify each reference object edgel as being an interior edgel or a boundary edgel;

3. For each reference object interior edgel from *R1*, the best matching test edgel is searched in a local area in the test image, as in the pose refinement process. If the match score exceeds a detection threshold, this matched test edgel and its edgel neighbours within 1 pixel distance will be all labeled as test object edgels;

4. For each reference object boundary edgel from *R1*, the same search process as in Step 3 is implemented, but there is a different strategy to undertake feature extraction, as described below. This process is repeated again for the corresponding reference object boundary edgel from *R2*;

5. A fine-to-coarse approach is used to match the HexBinary features sampling on edgel locations. Contour edgel matching commences by first matching the highest level of the grouped HexBinary descriptor, and then proceeding to attempt to match using the next lower level of the descriptor grouping. If no match is detected at the current grouping level, and if the descriptors at all (lower grouping) levels have been used without finding a successful edgel match, then no test object edgel will be labeled to a corresponding reference edgel.

The idea behind Step 2 above is inspired by [Arbelaez et al., 2011]. In detail, a strong gradient magnitude is more likely to occur at boundary positions. Therefore, χ^2 distance is employed to classify boundary edgels and their corresponding tangential directions. This process is applied to the reference edgels. In the reference image *R1*, a local patch, which is centred on an edge point, is extracted and divided it into two halves in the horizontal direction. The intensity histogram of each half patch is computed and compared by χ^2 distance to measure the gradient magnitude between the two parts. Each such patch is rotated for every 10 degrees per step in order to find the position where the biggest χ^2 for this patch occurs. When a maximum χ^2 is found, and if one of the half patches A_1 includes a sufficient number of background valued pixels, exceeding the number of those found in the other half patch A_2, A_1 is deemed to cover pixels from the background, and this edge point is defined as a reference boundary point on the object. Otherwise it is defined to be a reference edgel inside the object. Thereafter, HexBinary descriptors are extracted centred on those interior reference edgels to serve edgel matching. However, a different approach is taken for reference boundary edgels: in order to avoid background clutter disrupting descriptors located on the object bounding contour, the centre of these HexBinary descriptors is displaced in a direction normal to the edge boundary contour towards the reference object interior by r pixels (side length of the first level hexagon). This process substantially eliminates background clutter pixels from object boundary descriptor samples. Descriptors located on bounding contour edgels within the corresponding search area in the test image are shifted in the same manner to generate a matching descriptor.

5.4 Performance Evaluation

To evaluate the hypothesis that HexBinary descriptors can provide better pose refinement performance with less time consumption than HexSHoG in the 2D object localisation system, the dataset used in [Liu and Siebert, 2014] is employed in this section: For each reference image from Amsterdam Library of Object Images (ALOI) dataset [Geusebroek et al., 2005], five corresponding test images are generated by embedding randomly assigned object poses into five different backgrounds. Therefore, the validation test set comprises 1000 reference and 5000 test images. The parameters of HexBinary descriptor are empirically set as follows: The hexagon edge length is 3 pixels and the Gaussian kernel size for smoothing the image is 9 pixels with standard deviation 2. The matching threshold is 0.2 which is a *dissimilarity* threshold as described below in the range of [0 1].

5.4.1 Rotation-Invariant Performance of HexBinary

In order to firstly evaluate the rotation-invariant property of HexBinary descriptor, 20 different reference images are selected randomly from ALOI dataset and the rotated versions are generated in 1 °step over 180 °. HexBinary descriptors of the key-points detected by Fast Corner Detector [Rosten and Drummond, 2006] are extracted and compared to find matches from the reference image to those of each of its rotated images. The experimental hypothesises of this experiment are: HexBinary descriptor achieves rotation invariance by using the hexagonal grouping structure of HexSHoG, and the rotation-invariant performance decreases with the hierarchical level increasing. The total number of bit differences between compared binary strings normalised by the bit string length is used as the matching *dissimilarity* score which is averaged over the 20 different reference images, and plotted as a function of rotation. For all the above-proposed descriptors, they show similar performance results to those in Figure 5.3, illustrating the degree of rotation invariance of three different

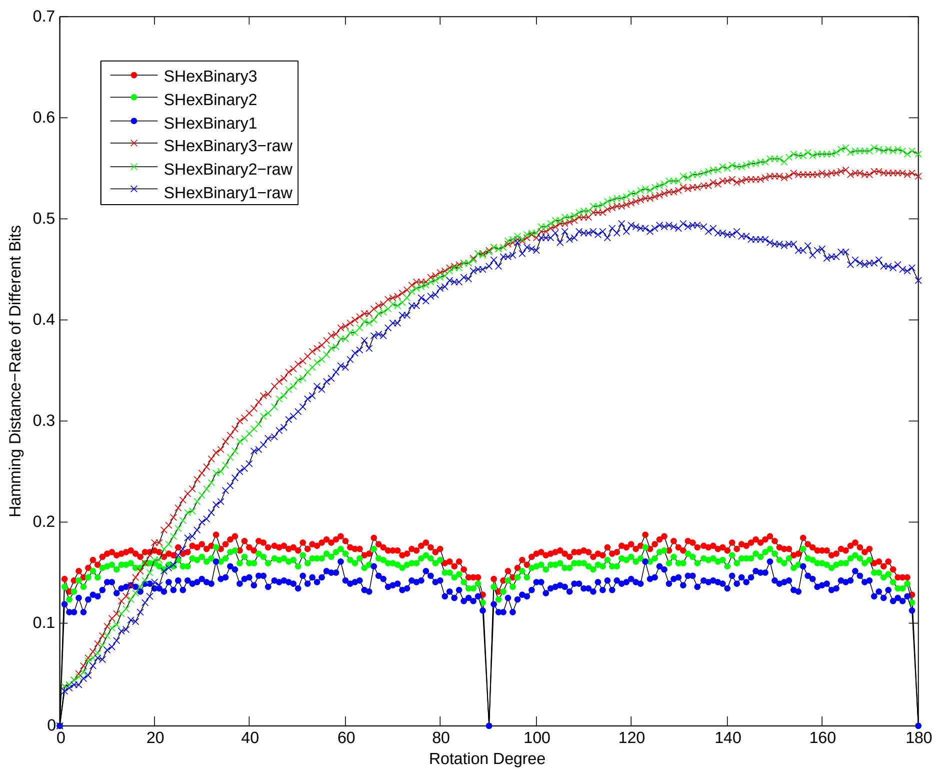
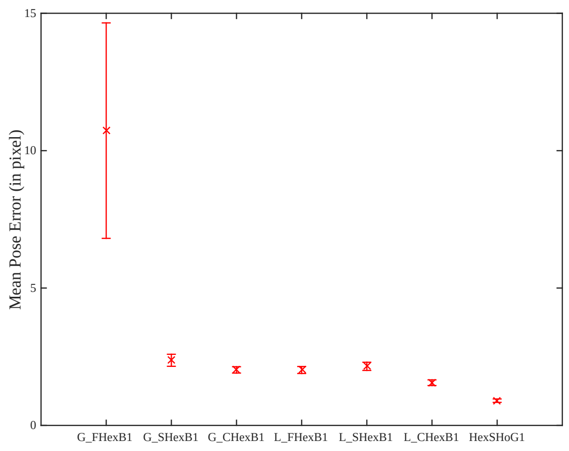


Figure 5.3: The matching results for different HexBinary descriptors as a function of rotation degree. The difference between two binary descriptors is computed by Hamming Distance.

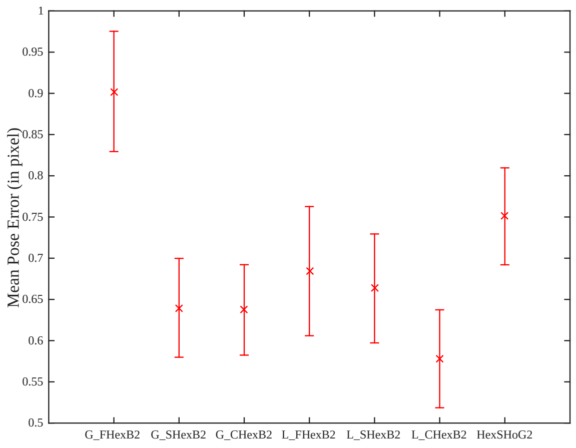
grouping levels of the second order descriptor *SHexBinary* from Gaussian filtered images. The results demonstrate that all the 3 levels of *SHexBinary* descriptor provide rotation invariance, returning matching (dissimilarity) scores smaller than 0.2 for all compared rotations, while the performance of all levels of the raw (rotation-variant) descriptors *SHexBinary-raw* decreases gradually with increasing compared rotation. It also clearly shows that the lower level descriptor has better rotation invariance than the higher level descriptor of *SHexBinary*.

5.4.2 Pose Refinement Performance

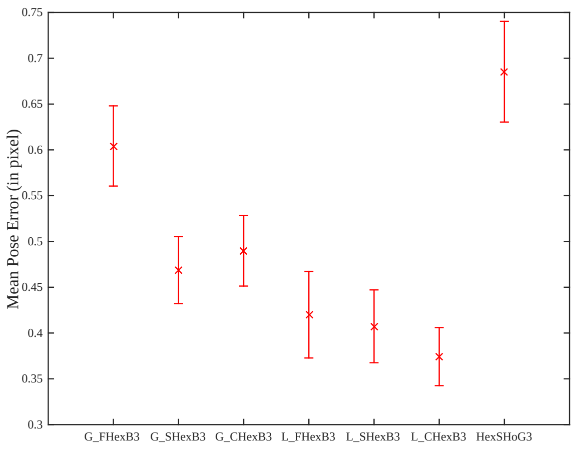
As described in the previous chapter, the pose refinement process is based on an initial pose estimate by using standard SIFT. 5000 test images are tested and only 2892 of these images are successfully detected by SIFT to provide an initial pose estimate. The failures of initial pose estimate tend to be due to lack of detected key-points. The pose refinement process is only applied to those images which provide an initial pose estimate based on each of the proposed hierarchical HexBinary descriptors in isolation. Following the validation protocol introduced in the previous chapter, a search range of ±5 pixels gives superior performance for HexSHoG. Therefore, the same search range for each HexBinary descriptor is used to make comparisons with HexSHoG. The HexBinary descriptors generated by sampling LoG



(a)



(b)



(c)

Figure 5.4: Pose refinement performance with 95% confidence interval for the mean pose error: (a) The level one performance of different hierarchical descriptors; (b) The level two performance of different hierarchical descriptors; (c) The level three performance of different hierarchical descriptors.

filtered images are also tested. The edgel displacement error is computed by the distance between the estimated edgel position and the ground truth edgel position, from which the mean and the standard deviation of the local edgel displacement error are provided after pose refinement and corresponding to the initial pose estimate value, for all images whose pose estimate improved after refinement.

Figure 5.4 illustrates the pose refinement performance as a function of different descriptors, with the error bars showing 95% confidence interval for the mean pose error of single edgel position. It is observed in Figure 5.4 that, except in level one, there is at least one HexBinary descriptor performing statistically significantly better than HexSHoG in other two levels of the feature hierarchy. In detail, the mean and the standard deviation of the pose error after pose refinement for different hexagon-based descriptors are described in Table 5.1 and Table 5.2, respectively. The number of improved images after pose refinement for different de-

Table 5.1: Mean error of single edgel position for each level of descriptor and the corresponding reduced value Δ from the initial mean error (G_FHexB means the first order HexBinary descriptor generated from Gaussian filtered images; L_SHexB means the second order HexBinary descriptor generated from LoG filtered images).

Descriptors	Level1	Δ_1	Level2	Δ_2	Level3	Δ_3
G_FHexB	10.7297	0.5345	0.9023	1.3188	0.6043	1.5178
G_SHexB	2.3705	0.4036	0.6399	1.5096	0.4687	1.7259
G_CHexB	2.0208	0.4860	0.6373	1.5099	0.4898	1.6978
L_FHexB	2.0475	0.6178	0.7049	1.5707	0.4200	1.7901
L_SHexB	2.2066	0.5557	0.6809	1.5634	0.4070	1.8031
L_CHexB	1.5340	0.9767	**0.5822**	**1.6370**	**0.3743**	**1.8327**
$HexSHoG$	**0.9008**	**1.3570**	0.7508	1.4612	0.6853	1.5068

scriptors is also provided in Table 5.3. From these tables it is observed that all the proposed descriptors based on hierarchical hexagon configurations give improved pose refinement, and that the pose refinement results improve with increasing levels of hierarchical descriptor grouping. Although the first level HexSHoG outperforms all the first level HexBinary descriptors, as the level of HexBinary grouping increases, HexBinary descriptors give better performance than HexSHoG. As predicted, the higher level HexBinary is always superior to the lower level HexBinary in the previous 3 levels. We can see from Table 5.3 that the third level of $G\_SHexBinary$ descriptor achieves the largest number of improved images for pose refinement. However, regarding the results in Table 5.1 and Table 5.2, the third level of the combined descriptor $L\_CHexBinary$ performs best in terms of the final pose estimate because it includes both the first order and second order derivatives of the feature intensity, which provides more information than both $FHexBinary$ and $SHexBinary$ descriptors. The tables also indicate that the descriptors generated from the LoG filtered images mostly work better than those generated from the Gaussian filtered images in terms of mean pose error. This might be due to that the Laplacian filter highlights the positions of rapid intensity change in the image, which leads to a better representation of edge contours by $L\_HexBinary$ descriptors.

Table 5.2: Standard deviation of single edgel position error for each level descriptor and the corresponding reduced value Δ from the initial standard deviation. Abbreviations of the descriptors are defined as the same as in Table 5.1.

Descriptors	Level1	Δ_1	Level2	Δ_2	Level3	Δ_3
G_FHexB	12.9189	0.0912	2.2734	0.0846	1.3909	0.3558
G_SHexB	3.6426	0.0177	1.9085	**0.2653**	1.1681	**1.1439**
G_CHexB	2.4594	0.0149	**1.7439**	0.2542	1.2305	1.0427
L_FHexB	2.7983	-0.0099	2.4379	0.1373	1.4875	0.6502
L_SHexB	3.0726	0.0159	2.0399	0.1334	1.2455	0.8879
L_CHexB	2.6506	-0.0048	1.8427	0.2386	**0.9965**	1.0803
$HexSHoG$	**1.9002**	**0.1166**	1.8327	0.2145	1.7235	0.3265

Table 5.3: Number of improved images after pose refinement. Abbreviations of the descriptors are defined as the same as in Table 5.1.

Descriptors	Level1	Level2	Level3
G_FHexB	31	2683	2789
G_SHexB	766	**2800**	**2826**
G_CHexB	1258	2795	2815
L_FHexB	1060	2088	2734
L_SHexB	896	2064	2707
L_CHexB	1549	2115	2728
$HexSHoG$	**2598**	2685	2720

Timings of computing HexBinary and HexSHoG descriptor are recorded on a computer running Ubuntu 14.04 (64-bit), with an Intel Core i5 3.10 GHz processor. All the experiments are implemented in Matlab. The computing time of a single descriptor is given by the average time of 100 descriptors, and the results are averaged by 20 runs. The computing time and the descriptor dimension length of 3 different levels of HexBinary and HexSHoG are illustrated in Table 5.4 and Table 5.5, respectively. From Table 5.4 we can see, HexBinary descriptor has a clear advantage compared to HexSHoG in each single level. The second order descriptor $SHexBinary$ computes faster than the first order descriptor $FHexBinary$ and the combined descriptor $CHexBinary$, because less number of pair comparisons is required to generate the binary bits of $SHexBinary$, which can be clearly observed from Table 5.5. Regarding each single level of the descriptors respectively, the second order descriptor *SHexBinary* has the shortest dimension comparing to all the other descriptors, while *HexSHoG* has the longest dimension.

Table 5.4: Computing time for a single descriptor

Timing [ms]	Level1	Level2	Level3
$FHexBinary$	1.10	6.3	46.3
$SHexBinary$	1	5.4	38.5
$CHexBinary$	1.13	6.4	47.6
$HexSHoG$	2	12.7	92.7

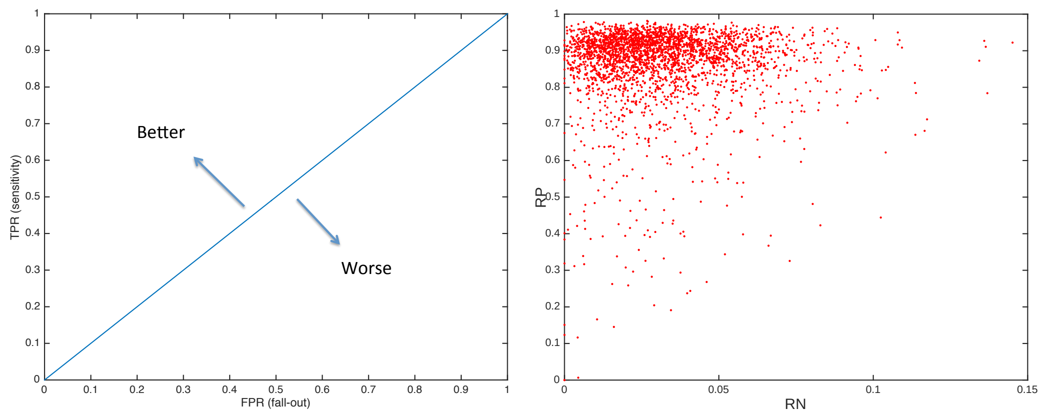
Table 5.5: The descriptor dimension length

Descriptors	Level1	Level2	Level3
$FHexBinary$	12	84	588
$SHexBinary$	9	63	441
$CHexBinary$	21	147	1029
$HexSHoG$	56	392	2744

5.4.3 Edge Contour Localisation Performance

In order to illustrate the performance of edge contour localisation clearly, the Receiver Operating Characteristic (ROC) analysis is employed in this section. The ROC curve provides a clear illustration of how the binary classifier performs according to different thresholds. The *sensitivity* is defined to measure the proportion of correctly identified positives over all the positives, which is also called True Positive Rate (TPR). Likewise, False Positive Rate (FPR) (also named *fall-out*) is defined to measure the proportion of wrongly identified positives, which should be classified as negatives out of all the negatives. In general, the ROC curve is plotted by the *sensitivity* as a function of *fall-out* as Figure 5.5 (a) shows. The closer the ROC point is to the biggest TPR (equalling 1), and to the smallest FPR (equalling 0), the better the performance of the classifier is. ROC analysis provides a direct way to make the optimal decision for the binary classifier. Inspired from the ROC analysis, similar conceptions are defined in this section to give a clear illustration of the edge contour localisation performance.

Because all the HexBinary descriptor variants perform almost equally well in pose refinement, $G\_SHexBinary$ is employed due to its shorter vector length and best performance in terms of number of improved images by pose refinement, for edge contour localisation using a search range= ± 3 pixels. In order to evaluate the results, the reference mask is rotated



(a) (b)

Figure 5.5: (a) ROC curve illustration. (b) Distribution of pixel-level localisation: each red dot represents an edge labelling ROC point, *RP* vs *RN*.

according to the ground truth pose information and projected into the test image with 1 pixel dilation. The number of test edgels inside the mask is termed as *NTP* and the number of the labeled test edgels inside the mask is termed as *NPL*. The number of the reference edgels inside the mask is computed as *NR*, and the number of the labeled test edgels outside the mask is named as *NNL*. *RP=NPL / NTP* is defined as the correct labelling rate, and *RN=NNL / NR* is defined as the false labelling rate. They are used to evaluate the pixel-level localisation performance. RP and RN of each pose-refined image are computed, and the distribution of them is illustrated in Figure 5.5 (b). With the proposed method, the edge contour labelling process achieves viable results with the mean RP=0.8654, and the mean RN=0.0314, over all the pose refined images.

Some representative examples of the object edge contour localisation results are represented as in Figure 5.6. A number of edgels have been miss-labelled because the background is similar in appearance to the object, while a number of object edgels detected in the test image might not have been detected in the reference image due to the edge detector not producing consistent edge labels between these views, and therefore, inconsistent edge labels will be missed. If corresponding edgels are not found within the adopted search range, this also results in missing labels in the test image.

5.5 Summary and Discussion

In this chapter, a novel hexagonally sampled and hierarchically composed BS descriptor is proposed as HexBinary. It is constructed based on Gaussian filtered or LoG filtered images. The binary bit is computed from comparing the intensity pairs and the pairs of intensity difference in the hexagonal structure, which generates the first order and the second order

HexBinary descriptors, respectively. A combined descriptor is also introduced by simply concatenating the two order HexBinarys. In addition, a complete framework for 2D object localisation with qualitative and quantitative analysis is also introduced in this chapter by using HexBinary descriptors based on the system proposed in the previous chapter.

Through the validation performances of pose refinement, HexBinary has demonstrated its effectiveness and efficiency better than HexSHoG for in-plane transformed images. The computing efficiency of the hexagonal descriptor is mainly improved by replacing computing SHoG of all the hexagonal vertex-centred patches by only comparing the pairs of the hexagonal vertexes, which also leads to much faster feature matching with shorter descriptors. Moreover, the 2D object localisation system achieves promising performance with adding extra process to reduce the background influence when extracting the HexBinary descriptors at boundary edges.

Both the previous chapter and this chapter have demonstrated that local feature descriptors of edge points extracted in a hexagonal structure have the capability to represent well the local information around edges. However, the recursive process of constructing such hierarchy produces an excessive level of redundancy with the hierarchical level increasing. Furthermore, the parameters used for generating the descriptors are determined based on the published empirical results, which might limit the power of the hexagonal structure. To better contribute to local feature extraction based on the hexagonal structure, in the next chapter, a new hexagonal grouping structure and a discriminative learning work of the parameters will be introduced based on the construction of HexBinary. Moreover, the new proposed descriptors will also be evaluated based on the application of object pose estimate by matching features of sparse corner points.

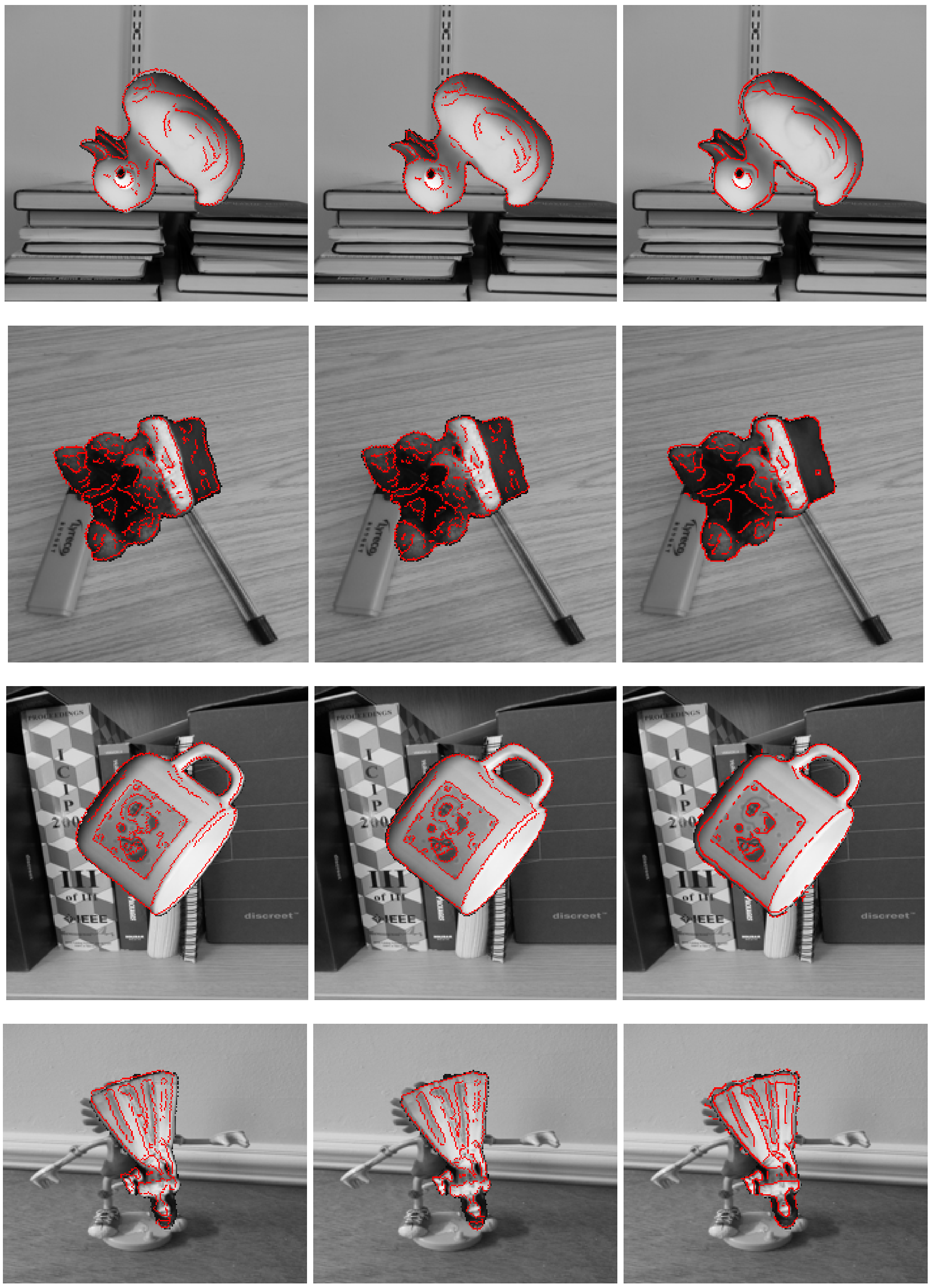


Figure 5.6: Pose estimate and edge contour localisation examples: The first column shows reference contour projection based on initial pose estimate by SIFT. The second column displays reference contour projection based on refined pose estimate by $G\_SHexBinary3$. The third column illustrates directly labelled edge contour points of the test object.

Chapter 6

Learning Better HBDs: Hexagon-Based Binary Descriptors

In the previous two chapters, two rotation-invariant local feature descriptors: HexSHoG and HexBinary, were proposed based on the same hexagonal grouping structure. They are both applied in the proposed 2D object localisation system and achieve promising performances. However, as the level of hierarchical grouping increases, the central part of the feature support region is repeatedly overlapped, which results in the same information being repeated within the descriptor many times, producing an excessive level of redundancy. Therefore, in this chapter, a new hexagonal grouping structure is introduced to reduce the frequency of overlap of the same image area during hierarchical grouping, and two new hexagon-based binary descriptors (HBDs) are proposed using this new grouping mechanism. A parameter learning process for the proposed HBD is also implemented. The new descriptors are evaluated in a pose estimate application and achieve competitive performance compared to the state-of-the-art descriptors.

6.1 Motivation and Objective

HexBinary descriptors employ a hierarchical grouping mechanism which combines vectors representing overlapping image regions to improve the feature's discriminability. However, this approach can result in repeated overlapping of the same local image area to produce excessive redundancy, thereby degrading the descriptor's performance. In addition, the parameters used to compute HexBinary are based on the empirical results according to the published feature BRIEF [Calonder et al., 2010]. Since the structure of HexBinary is different from any other descriptors, it is necessary to determine the appropriate parameters specific to this descriptor. If the sampling structure of HexBinary has been defined, then two more

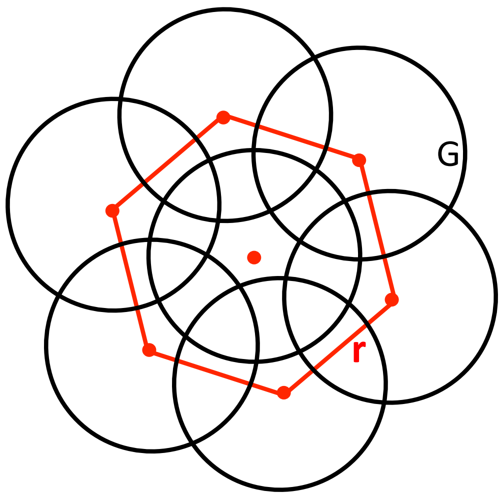


Figure 6.1: The first level hexagonal structure for generating HexBinary descriptors. The black circle G is the Gaussian window for smoothing, and the red edge r is the edge length of the basic hexagon.

parameters need to be considered before computing HexBinary descriptors as illustrated in Figure 6.1 : 1) The edge length of the basic hexagon r at the first level: The hexagon hierarchy is generated by joining more basic hexagons as the level of grouping increases. The edge length of the basic hexagon will directly affect the capability of the descriptor to represent local patterns; 2) The standard deviation of the Gaussian kernel and its support region size: It is always the first step to smooth the input image to reduce irrelevant detail and image noise before the further image processing, i.e., set the level in image scale-space on which the system is operating. The standard deviation of the smoothing kernel defines the degree to which the image will be blurred. An appropriate kernel support region size must be computed according to the Gaussian standard deviation in order to represent this convolution kernel accurately.

Motivated by these, in this chapter, a new sampling structure of generating the hexagonal hierarchy is proposed to reduce the excessively redundant information produced during the grouping steps. The corresponding parameters of constructing the hierarchical structure are also learned to improve the feature distinctiveness. Based on the new proposed structure, two new HBDs are introduced as Hexagon-based Intensity Difference Binary (HexIDB) and Hexagon-based Local Difference Binary (HexLDB). They are also compared to the state-of-the-art descriptors in pose estimate application.

6.2 Approach

In this section, the details of how the new descriptors are generated based on the new sampling structure are described.

6.2.1 Sampling Structure

The sampling structure used for computing the second level of HexBinary is illustrated in Figure 6.2 (a). The red $\star$ indicates the feature point position, and the descriptor for this feature point is computed from the neighboured regions in the hexagonal structure. In the previous chapter, the validation of HexBinary descriptors has demonstrated that this hierarchy, up to the third level, is a good trade off between descriptor matching effectiveness and efficiency. Therefore, in this chapter, the hierarchy is also only considered up to the third level.

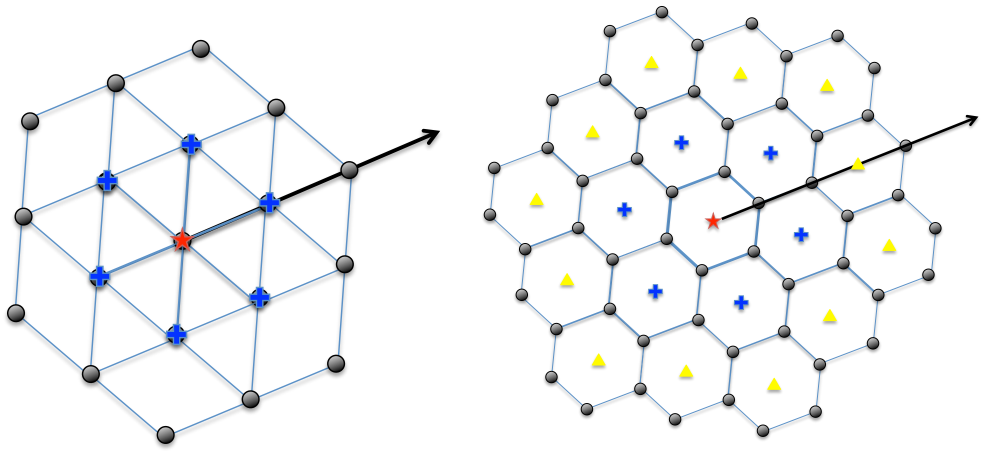

(a) (b)

Figure 6.2: (a) is the second level structure of generating HexBinary. (b) is the new proposed structure of the third level hexagonal descriptor. The red star point represents the key-point position where to sample the local descriptor. The arrow illustrates the dominant orientation of the local region around the key-point. The first level hexagonal grouping structure comprises the basic hexagon centred at the red $\star$, and the sampling positions for generating binary strings are taken from the hexagon centre and the vertexes. The second level hexagonal structure now covers more image area around the key-point since six more hexagon centres are sampled as shown in blue $+$, which are sampled at different positions in (a) and (b). For the third level descriptor, another 12 more hexagon centres as shown in yellow $\triangle$ in (b) are computed together with the previous 7 hexagon centres around the key-point, while in (a), 7 second level structures will be constructed centred on the red $\star$ and blue $+$. All the black points indicate the sampling positions according to the corresponding hexagonal centre positions.

The sampling structure used to construct three levels of HexBinary descriptors is summarised as follows: A hexagon of defined size is constructed centred on the feature point p, so in total 7 points are sampled at: the 6 vertexes and the central location of the hexagon. For the first level HexBinary descriptor termed as *HexBinary1* of p, the binary bits are computed from the 7 sampling points in this single hexagon. For the second level descriptor *HexBinary2*, the 7 sampling positions of the hexagon are treated as 7 feature points. The *HexBinary1* descriptors comprising 7 feature points are similarly computed, and are then concatenated together to form the second level descriptor of p. Similarly, the third level descriptor *HexBinary3* of p is generated by concatenating the *HexBinary2* descriptors of the 7 feature points likewise computed, as described above. Therefore, as the grouping level of the descriptor increases, the information around the 7 sampling positions will be repeatedly overlapped. Although overlapped-sampling can improve the stability of a local descriptor, repeated overlapping will eventually result in excessively redundant information being accumulated, which decreases the discriminability of the feature descriptor. To address the issue of redundant information when generating higher level descriptors, the new proposed hexagonal structure is constructed as in Figure 6.2 (b). The details of how to compute a three level hierarchy are now presented in the following steps:

1. Localise the key-point position, and compute the dominant orientation of the local area around this key-point according to the method introduced in Chapter 5;
2. According to the dominant orientation and the given edge length of the hexagon, the sampling positions of the basic hexagon vertexes are determined. Then the first level hexagonal descriptor can be computed according to the binary descriptor generation mechanism that will be introduced in the later subsection;
3. Sample another 6 positions to be the new hexagon centres as the blue $+$ shown in Figure 6.2 (b). These new hexagon centres are not the 6 vertexes of the basic hexagon generated in first level, which is the main difference from the HexBinary structure in Figure 6.2 (a). Each new hexagon shares an edge with the basic hexagon centred at the key-point;
4. The second level hexagonal descriptor is generated by concatenating the 7 binary descriptors extracted based on the 7 basic hexagons according to the grouping mechanism introduced in Chapter 5;
5. Sample 12 more positions to be the new hexagon centres as the yellow $\triangle$ in Figure 6.2 (b). This is the similar process as step 3. which also differs from the HexBinary structure;
6. Concatenate these first level descriptors extracted centred on the 19 basic hexagons to generate the third level hierarchical descriptor.

Throughout the above steps, the higher level descriptors are generated by extending the feature area without repeatedly overlapping the central area for too many times, while in the HexBinary hierarchical structure, the overlap frequency of the central area is increased as the grouping level of the hierarchy increases. For instance, the blue $+$ areas in Figure 6.2

(a) are repeatedly overlapped by 7 times for constructing the second level descriptor and 49 times for the third level descriptor, while the same positions in Figure 6.2 (b) are only overlapped 3 times for all the higher level descriptors over the first level. The pseudocode to generate the new hierarchical HBDs is presented in Algorithm 9.

Algorithm 9 Generation of New Hierarchical HBD Descriptor

```
$p_0(x, y)$: feature point
$\theta_0$: local dominant orientation centred at $p_0$
$L$: defined edge length of the basic hexagon
$V_i(x, y)(i \leftarrow 1, 2...6)$: vertex positions of the basic hexagon
$ts \leftarrow pi/3$
for $i \leftarrow 1 : 6$ do
    $tv \leftarrow (i - 1)ts + \theta_0$
    $V_i(x) \leftarrow p_0(x) + L \times cos(tv)$
    $V_i(y) \leftarrow p_0(y) + L \times sin(tv)$
end for
Compute the first level descriptor $HBD1_0$ for feature point $p_0$
for $i \leftarrow 1 : 6$ do
    $tv \leftarrow (i - 1)ts + \theta_0 + pi/6$
    $p_i(x) \leftarrow p_0(x) + 2L \times cos(pi/6) \times cos(tv)$
    $p_i(y) \leftarrow p_0(y) + 2L \times cos(pi/6) \times sin(tv)$
end for
Compute the first level descriptor $HBD1_i$ for $p_i(i \leftarrow 1, 2...6)$
Generate the second level descriptor $HBD2_0$ for feature point $p_0$
$HBD2_0 \leftarrow HBD1_0, HBD1_1, ..., HBD1_6$
for $i \leftarrow 7 : 12$ do
    $tv \leftarrow (i - 7)ts + \theta_0$
    $p_i(x) \leftarrow p_0(x) + 3L \times cos(tv)$
    $p_i(y) \leftarrow p_0(y) + 3L \times sin(tv)$
end for
for $i \leftarrow 13 : 18$ do
    $tv \leftarrow (i - 13)ts + \theta_0 + pi/6$
    $p_i(x) \leftarrow p_0(x) + 4L \times cos(pi/6) \times cos(tv)$
    $p_i(y) \leftarrow p_0(y) + 4L \times cos(pi/6) \times sin(tv)$
end for
Compute the first level descriptor $HBD1_i$ for $p_i(i \leftarrow 7, 8...18)$
Generate the third level descriptor $HBD3_0$ for feature point $p_0$
$HBD3_0 \leftarrow HBD1_0, HBD1_1, ..., HBD1_{18}$
```

6.2.2 Descriptor Generation

In the previous chapter, three different HexBinary descriptors were generated based on Gaussian smoothed images, which are based on the first order, the second order and the combination of the two orders of the image intensity, respectively. Although the combined descriptor *CHexBinary* (concatenating *FHexBinary* (first order HexBinary) and *SHexBinary* (second order HexBinary) together, details in Section 5.2 of Chapter 5) achieves the best performance of pose refinement in terms of mean pose error, it has the longest feature vector which consumes more time for computing and matching. *SHexBinary* has the shortest feature vector and obtains the biggest number of improved images after pose refinement. Therefore, to evaluate the new structure, the binary coding mechanism for generating the second order descriptor *SHexBinary* is employed here to construct new descriptors based on the above proposed structure.

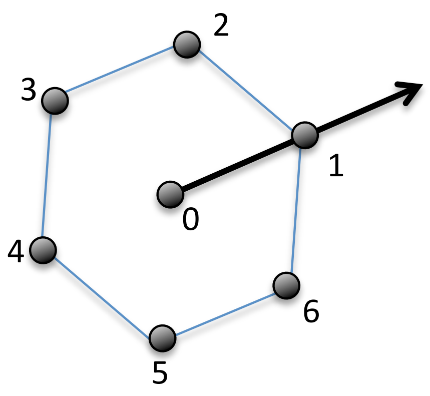


Figure 6.3: The first level structure of the hexagon-based hierarchical descriptor. The arrow indicates the local dominant orientation. The two sampling vertexes of the hexagonal structure are aligned with the local dominant orientation.

A first level structure of HBD defined according to the dominant orientation is shown in Figure 6.3. The image is first filtered by a Gaussian kernel of standard deviation σ, and the smoothed intensity values sampled at the hexagon centre and vertexes are denoted by I_i(*i=0,1,...6*). Then the first level descriptor is computed by comparing the intensity differences. The binary bit τ of the descriptor is corresponding to :

$$\tau\left(D; i, j\right) = \begin{cases} 1 & \text{if } D_i < D_j \\ 0 & \text{otherwise,} \end{cases} \tag{6.1}$$

where$(\mathbf{D}_i, \mathbf{D}_j)$ is a spatially adjacent pair of intensity difference, e.g., $(\mathbf{D}_i = I_1 - I_0, \mathbf{D}_j = I_0 - I_4; \mathbf{D}_i = I_1 - I_6, \mathbf{D}_j = I_2 - I_1)$. Then 9 pairs of $(\mathbf{D}_i, \mathbf{D}_j)$ in the hexagon are selected to generate a 9-bit binary string as the first level descriptor. When going up to the higher level descriptor, each new constructed hexagon in the structure will generate a first level descriptor, and they are concatenated together to form the higher level descriptor. This new generated hierarchical descriptor based on hexagonal structure is termed as Hexagon-based Intensity Difference Binary (HexIDB).

Similarly, another new descriptor Hexagon-based Local Difference Binary HexLDB is proposed by comparing not only the intensity difference, but also the gradient difference. This is similar to but slightly different from Local Difference Binary (LDB) descriptor proposed in [Yang and Cheng, 2012]. LDB generates a 3-bit vector by comparing the differences of the local average intensity, gradients in x and y directions between the pair of grids, respectively. In this chapter, the gradient information is also considered but without being divided into x and y directions. A gradient map is computed, then the comparison pair $(\mathbf{D}_i, \mathbf{D}_j)$ in Function 6.1 could be the gradient difference pair, e.g., $(\mathbf{D}_i = G_1 - G_0, \mathbf{D}_j = G_0 - G_4)$. G_i(i=0,1,...6) represents the gradient value of the sampling position. Therefore, for each pair comparison, a 2-bit vector is generated, and in each level of the Hierarchy, HexLDB will have double length of the vector than HexIDB. The descriptor length and the number of sampling fields of *SHexBinary* and the new proposed HBDs are illustrated in Table 6.1, from which we can see, the new HBDs sample $\sim$61% fewer fields for the third level descriptor with shorter length.

Table 6.1: The descriptor length (L) and the number of sampling fields (N)

$L \mid N$	Level1	Level2	Level3
$SHexBinary$	9 | 7	63 | 49	441 | 343
$HexIDB$	9 | 7	63 | 49	171 | 133
$HexLDB$	18 | 7	126 | 49	342 | 133

6.3 Performance of Rotation Invariance

To evaluate the hypothesis that the new HBDs have better rotation invariance than HexBinary with less redundant information, an experiment is implemented to compare the performances of matching local feature descriptors between the new HBDs and HexBinary. The matching criteria is to find the Nearest Neighbour (NN) and the measurement of the matching performance is referred as $RecognitionRate$, which is defined in the same way as in [Calonder et al., 2010]. $RecognitionRate$ is computed as follows: Firstly, N key-points are detected in the reference image, and N corresponding Key-points are inferred in the test image according to the ground-truth geometric relation between the two images. Secondly, compute the $2N$ key-point descriptors by the method under consideration, and for each descriptor in the

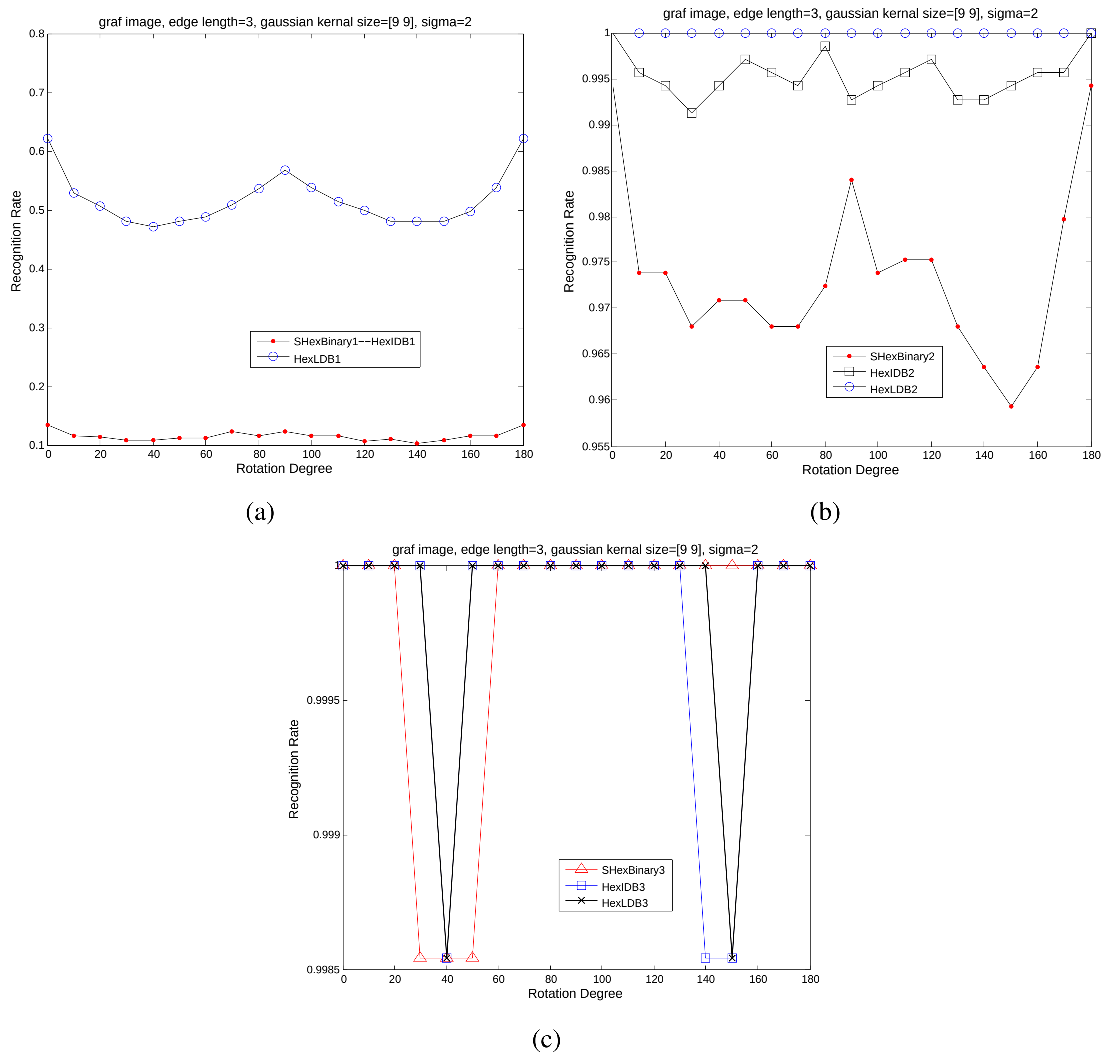


Figure 6.4: $RecognitionRates$ of different level comparisons among *SHexBinary*, *HexIDB* and *HexLDB*. The comparison of each single level is illustrated from (a) to (c), respectively.

reference image, find its NN in the test image. Then $RecognitionRate$ is given by C_n/N, where C_n is the number of correct matches.

In order to have a fair comparison for the descriptors having different structures and coding mechanisms, the edge length of the basic hexagon and the Gaussian smoothing kernel size are set all the same for different descriptors. The edge length is 3, and the Gaussian kernel size is 9×9 with sigma 2, which are the parameters used in the previous chapter. The Graffiti image, which will be shown in Figure 6.5, is employed for this evaluation. The first image of Graffiti sequence is rotated from 0 to 180 degrees in regular steps to generate the rotated images. FAST detector [Rosten et al., 2010] is employed to find the key-points in the original image, and the corresponding points in the rotated images are inferred by the ground truth information.

Figure 6.4 illustrates the rotation-invariant performances of different descriptors with different levels. Since the structure of the first level hexagonal descriptor is all the same for *SHexBinary*, *HexIDB* and *HexLDB*, Figure 6.4 (a) clearly indicates that *SHexBinary1* performs equally to *HexIDB1* while *HexLDB1* performs dramatically better because of its enrichment with the second order gradient comparison information. As shown in Figure 6.4 (b), though all of the three second level descriptors demonstrate impressive rotation-invariant performances, *HexIDB2* performs better than *SHexBinary2*, and *HexLDB2* performs the best. When the structure goes up to the third level, the three descriptors illustrate similar excellent performances which are demonstrated in Figure 6.4 (c). According to these performances, the new proposed HBDs have illustrated the superiority to *SHexBinary* in terms of rotation invariance. Moreover, adding gradient comparison information into the descriptor rather than only using intensity comparison information can improve the rotation invariance of the local feature descriptor.

6.4 Parameter Learning of HBD

Good feature descriptors always rely on the good collaboration amongst their critical parameters. For HBD (HexIDB and HexLDB in this chapter), the essential parameters are: the edge length of the basic hexagon, the standard deviation σ, and support size of the Gaussian sampling kernel. These parameters will be comprehensively investigated by matching local features, which is described in the following subsections.

6.4.1 Dataset

The experiment is performed on the well-known and publicly available image dataset from [Mikolajczyk et al., 2005] as shown in Figure 6.5. The images contained in this dataset include typical image disturbances occurring in real-world scenarios, such as:

- viewpoint changes: Graffiti and Wall;
- image blur: Bikes and Trees;
- compression artifacts: Jpg;
- illumination changes: Light.

For each sequence, the test is designed to match the first image to the 5 remaining ones to get 5 pairs of matching cases sorted in order of ascending difficulty. Therefore, pair $1|6$ is much harder to match than pair $1|2$ for each sequence.

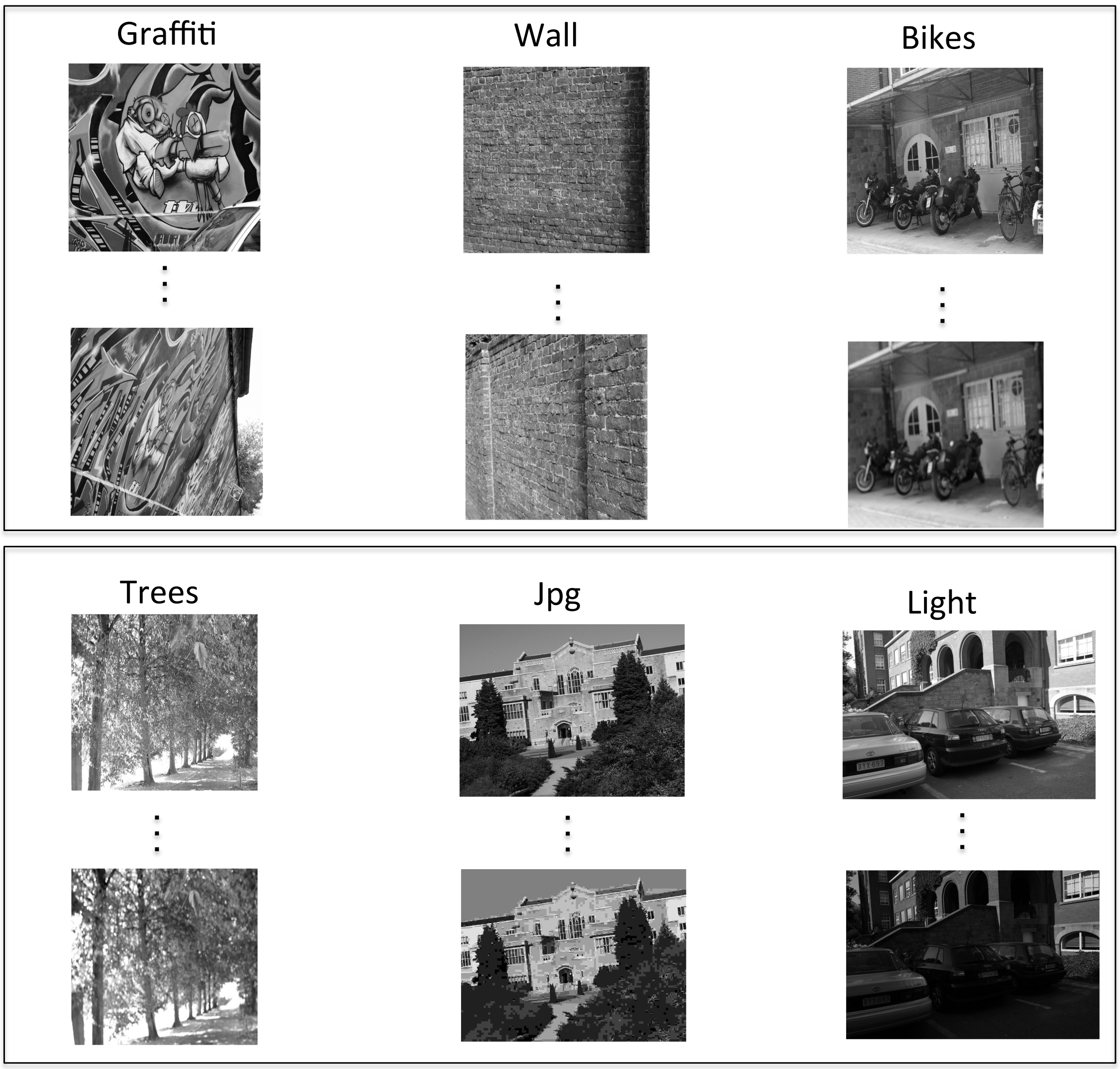


Figure 6.5: Image sequences: each sequence has 6 images, and only the first and the last images are illustrated. From the second to the sixth image, the difficulty in matching to the first image increases progressively.

6.4.2 Parameter Learning

The parameters of HBDs are evaluated by local feature matching. The measurement of the matching performance is still $RecognitionRate$ and the matching criterion is to find NN, which was introduced in Section 6.3. Any local feature detector can be employed to indicate where to extract the HBD. FAST is an efficient detector which is employed in this section. In the previous chapter, the edge length of the single hexagon is 3, and the Gaussian kernel size is 9×9 with $\sigma = 2$. When one of the parameters is tested, all the other parameters need to be fixed. For instance, when learning the Gaussian standard deviation σ, the edge length is set as 3, and the Gaussian kernel size is 9×9. The learning range of different parameters is shown in Table 6.2. All the parameter learning process is implemented on the third level descriptors, because higher level descriptors have more distinctiveness than the lower level descriptors.

Table 6.2: The learning range of different parameters

Parameters	Learning Range
$Sigma$	0.2−4.2
$Kernel\ size$	$13 \times 13 - 23 \times 23$ pixels
$Edge\ length$	1−8 pixels

For the new proposed HBDs, the Gaussian standard deviation *Sigma* is tested in the range of $[0.2, 4.2]$. The $RecognitionRate$ performances of the third level descriptors according to different sigmas are illustrated in Figure 6.7 and Figure 6.8. From the two figures we can see, for each pair matching, the $RecognitionRate$ is gradually improved with the sigma increasing, but it declines with the pair matching difficulty increasing for each image sequence. For both descriptors, the performance goes to be relatively constant when sigma reaches between [3, 4.2] for all the matching pairs. Therefore, sigma as 3.4 is chosen as a good compromise value to reduce the sensitivity to the noise while retaining the distinctive structure to achieve stable descriptors. Figure 6.9 demonstrates the performance of 3 level descriptors of *HexIDB* and *HexLDB* with *sigma*=3.4. It clearly shows that higher level descriptors perform better than lower level descriptors because of their bigger covering area including more distinctive information. For Graffiti, Wall, Light and Jpg sequences, *HexLDB* always performs superior to *HexIDB*. However, the two sequences with blurring issues: Bikes and Trees, give different results. With the descriptor level increasing, *HexLDB* gradually loses its advantage of including gradient comparison information and when the matching pair goes to be harder ones such as Bikes $1|6$, the gradient comparison information appears to disadvantage the *HexLDB* descriptor. This indicates that the gradient information in the image is greatly reduced when the image is significantly blurred, which results in gradient signals with a poor SNR.

According to the above analysis, in the later experiments for learning a proper Gaussian *kernel size*, *sigma* is defined as 3.4. All the learning results of the Gaussian kernel size are illustrated in Figure 6.10 and Figure 6.11. 10 different kernel sizes are tested on the six image sequences with *HexIDB3* and *HexLDB3* descriptors, respectively. From Figure 6.10 and Figure 6.11 we can see, the performances keep relatively stable when the *kernel size* changes between 13×13 and 23×23. Then *kernel size* 17×17 is selected for later parameter learning. Figure 6.12 shows the performance of 3 different levels of *HexIDB* and *HexLDB*, and illustrates consistent performances as in Figure 6.9.

After fixing the values of the Gaussian kernel size and the standard deviation, the experiments for learning the hexagon edge length are implemented and the results are shown in Figure 6.13 and Figure 6.14. It is observed in these two figures that, there is no significant difference when the *edge length* goes between [3, 8] for most images. In the cases of the two blurred images, Bikes and Trees, the larger edge length performs better than smaller edge length, particularly when the image pair is harder to match using more deeply blurred sequences. This may be because the deeply blurred images loose more high frequency information which makes the local point indistinct within a small area. For all the later experiments, an edge length of 3 pixels is employed to construct the hexagonal structure for local binary descriptors. With the learned parameters shown in Table 6.3, the performances of 3 different levels of *HexIDB* and *HexLDB* are illustrated in Figure 6.15 that shows consistent results as in Figure 6.9 and Figure 6.12.

Table 6.3: The learned values of different parameters for constructing the hexagonal structure.

Parameter	Sigma	Kernel Size	Edge Length
$Learned\ value$	3.4	17×17	3

6.4.3 Performance Evaluation

In order to evaluate the hypothesis that the new descriptors can provide competitive distinctiveness compared to the state-of-the-art descriptors, the performance of local feature matching by HBDs (including SHexBinary3, HexIDB3, HexLDB3 in this experiment) using the parameters in Table 6.3 is compared to the performance of the state-of-the-art descriptors, FREAK [Alahi et al., 2012] and SIFT [Lowe, 2004]. HBDs are claimed to have rotation invariance by directly constructing the sampling structure according to the local dominant orientation. There is no need to pre-rotate the local patch to align with the local dominant orientation, which is the conventional standard way to achieve rotation invariance. In order to have a fair comparison, scale and orientation are not considered in this test. FREAK, SIFT and HBD have all been coupled to the FAST detector for single scale experiments and termed as *U-desciptor*, which indicates that they do not normalise the descriptor orientation. Since SIFT is a multi-scale detected feature, for clarity, SIFT without rotation and scale-invariant property is termed as *USO-SIFT*.

Figure 6.16 illustrates the matching performance of each image pair with different descriptors. It is observed that on all the image sequence pairs except Graffiti, *U-HBD* and *USO-SIFT* both perform better than *U-FREAK*. *U-HexLDB3* always outperforms *U-HexIDB3* on image sequences of Graffiti and Wall. They achieve quite similar results on Light and Jpg image pairs, and also on the first three image pairs of Bikes and Trees sequences. For the harder-to-match pairs of Bikes and Trees, *U-HexLDB3* loses its advantage of utilising gradient comparison information. *U-SHexBinary3* gives inferior performance to *U-HexLDB3* and *U-HexIDB3* for almost all the image pairs, which confirms the improvement of distinctiveness for the new proposed hexagonal grouping structure. *USO-SIFT* achieves similar performance to *U-HexLDB3* and *U-HexIDB3* for most matching pairs comprising Light, Bikes, and Jpg sequences. For the remainder of the sequences, its performance is always inferior to that of *U-HexLDB3*.

6.5 Application to Pose Estimate

In this section, the new descriptors with the learned parameters are evaluated on the pose estimate application, which employs the same system as described in Section 4.3.1 in Chapter 4, with the hypothesis that the new proposed descriptors can achieve competitive performance in terms of pose estimate compared to the state-of-the-art descriptors, such as *SIFT*. Because the task is focused on the pose estimate rather than object detection, each test image with the object of interest in a cluttered background is directly matched to the corresponding reference image with the object of interest in pure black background. Therefore, one-to-one image matching is implemented to detect the object via the GHT, and a pose estimate is obtained by means of the RANSAC algorithm.

The pose estimate performance is evaluated by computing the mean and standard deviation (SD) of the pose errors of all the detected objects. The precise location of the edge contours of the reference object in the test image can be obtained according to the recorded ground-truth information, which specifies the rotation and translation used to embed the reference object pixels into a background image. Similarly, according to the recovered pose estimate using the system, the estimated object edge positions could be labelled by projecting the reference edge positions into the test image. The Euclidean distance between the estimated position and the ground-truth position of each reference edge point is computed to yield a pose estimate error for each matched edge location.

Five different features are tested, which are, standard SIFT (*SIFT*), SIFT without scale invariance (*US-SIFT*), *SHexBinary3*, *HexIDB3* and *HexLDB3*. Except *SIFT*, which employs its own feature detector to afford scale-invariant matching, all the other features are sampled at locations defined by key-points detected by FAST detector at a single scale. In addition, 5000 synthetic images are tested to match to the corresponding 1000 referent images, which is from ALOI dataset as used in the previous two chapters. The object detection and the pose estimate results are given in Table 6.4 and Table 6.5.

Table 6.4: Numeric distribution of images detected within a given pose error range (pixels) and the corresponding error ranges.

Error Range (pixel)	0-0.5	0.5-1	1-1.5	1.5-2	2-3	3-4	4-5	5-Inf
$SIFT$	270	355	**350**	**359**	**1348**	**108**	44	146
$US-SIFT$	**2787**	387	165	179	69	40	22	132
$SHexBinary3$	2767	**522**	182	85	83	52	21	**1163**
$HexIDB3$	2401	471	176	92	88	51	38	1036
$HexLDB3$	2570	465	175	84	113	50	33	607

Table 6.5: Number of images successfully detected with a pose error of less than 5 pixels, and their corresponding pose error Mean (*Mean*) and Standard Deviation (*SD*) in pixels.

Descriptor	SIFT	US-SIFT	SHexBinary3	HexIDB3	HexLDB3
$Number$	2834	3649	**3712**	3317	3490
$Mean$	1.9241	**0.4209**	0.4726	0.5295	0.5207
SD	0.9560	**0.6541**	0.6731	0.7489	0.7375

In Table 6.4, the number of detected images having the corresponding pose error is accumulated in different error ranges for each descriptor. Most of the detected images have the pose error less than 5 pixels for all the descriptors examined. Since it is a one-to-one image match, to better compare the performance of different descriptors, each detected image having pose error bigger than 5 pixels is defined as a failed detection. The *Number* of successfully detected images in Table 6.5 only accounts for the images with pose error smaller than 5 pixels, based on which, the *Mean* and *SD* of the pose error through the images are computed for each descriptor, respectively. Accordingly, the mean pose error result is also illustrated as a function of different descriptors in Figure 6.6, with error bars showing 95% confidence interval for the mean. It can be clearly seen from Figure 6.6 that, the standard *SIFT* performs statistically significantly worse than *US-SIFT* and three HBDs, while *US-SIFT* performs best with statistically significant difference from all the other descriptors under consideration.

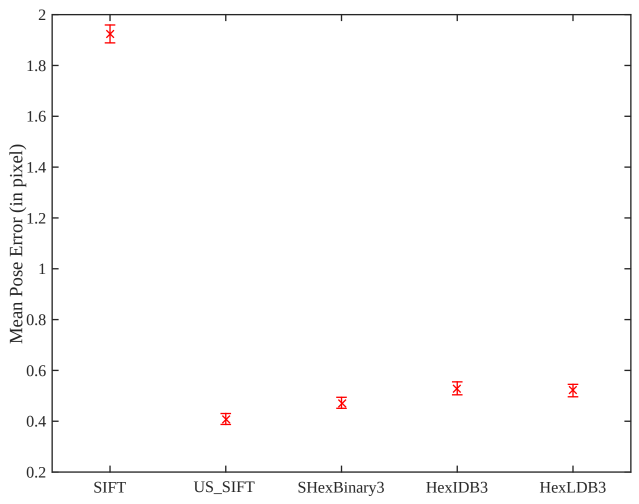


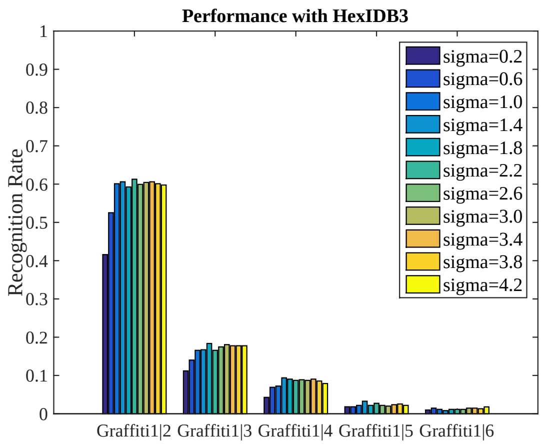
Figure 6.6: Pose estimate performance with error bars showing 95% confidence interval for the mean pose error based on the results given in Table 6.5.

In detail, it is clearly shown in Table 6.4 and Table 6.5 that, *US-SIFT* achieves the best performance in terms of the mean pose error and has the biggest number of images having pose error less than half pixel, while *SHexBinary3* has the biggest number of images successfully detected. *SHexBinary3* also has the least mean error and the biggest number of images having less than half pixel pose error among the three binary descriptors, however, it has the biggest number of detected images defined as failed examples. The test images do not have scale changes from the reference images, which might be the reason for *SIFT* exhibiting inferior results to all of the other descriptors. Due to multi-scale detection being applied in this case, the associated Hough parameter space needs one more dimension to be able to detect objects, compared to the Hough space generated for the other single scale descriptors, which leads to lower pose estimate accuracy. *HexLDB3* works slightly better than *HexIDB3* due to the extra comparison information from the gradient map. In summary, *US-SIFT* and HBDs have statistically significantly better performance than *SIFT* in terms of pose estimate, which can be clearly observed in Figure 6.6, exhibiting close results with an error of approximately half a pixel.

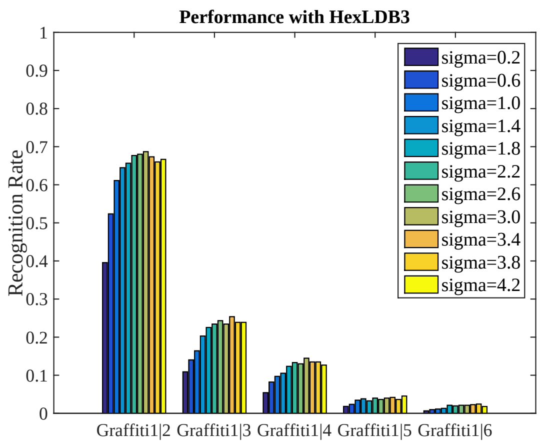
6.6 Summary and Discussion

This chapter introduces a new hexagonal grouping structure, which is designed primarily to reduce excessively redundant information during the hierarchical grouping progress, by decreasing the overlap frequency of sampled image regions. Based on the new structure, two new hexagon-based binary descriptors (HBDs): HexIDB and HexLDB, are generated. Moreover, the parameters used to construct HBDs are learned based on matching local features of the well-known image set, which is a standard choice for evaluating the performance of local feature descriptors. By using the learned parameters, the new proposed HBDs are then compared to the state-of-the-art descriptors based on the application of object pose estimate.

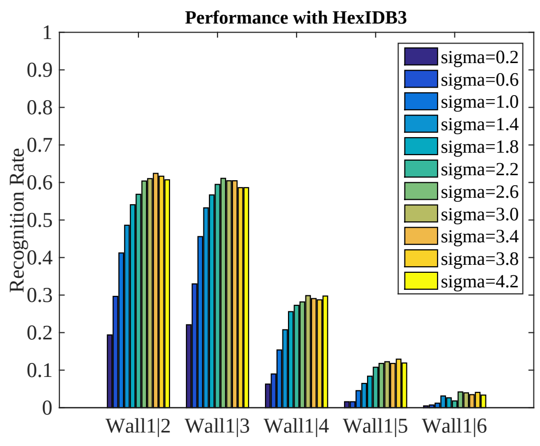
During the process of generating new HBDs, the new hexagonal grouping structure produces shorter feature vectors for the third level of the hierarchical feature descriptor with fewer repeatedly sampled regions, as compared to the structure of generating HexBinary descriptors. A gradient map is also employed to generate the descriptor binary bits in the same way as the intensity map is encoded. However, it is not a wise choice to use the gradient map when the gradient information representing image features has a low SNR. From the results of object pose estimate we can see, although the parameters used in this application are not learned from the training data, HBDs still produce much better performance than the standard *SIFT* and show competitive performance compared to *US-SIFT* with around a half-pixel mean pose error, without any subsequent pose refinement step. Based on this result, direct edge labelling can be implemented and also optimised by parameter learning in the future investigation.



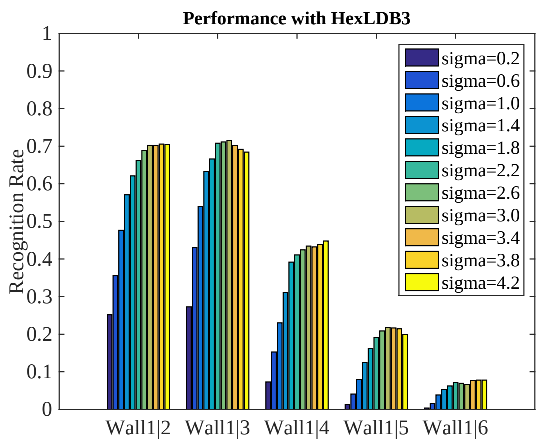
(a)



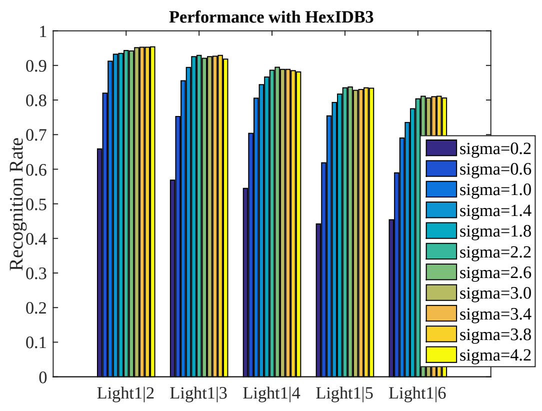
(b)



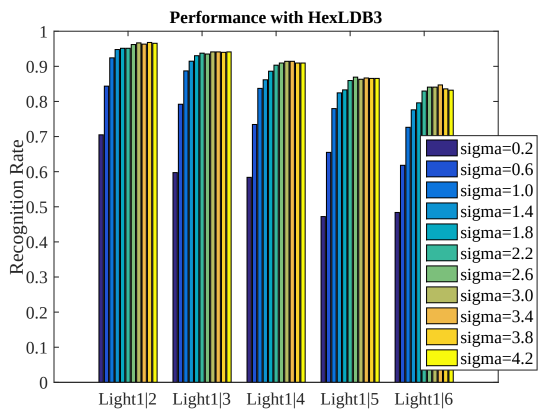
(c)



(d)

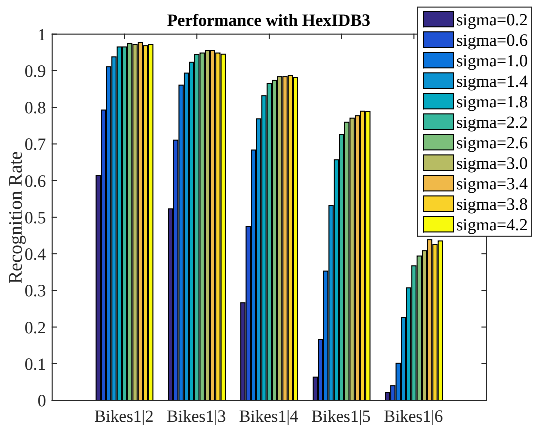


(e)

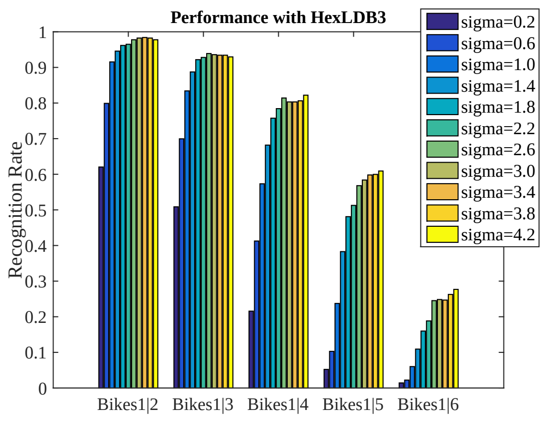


(f)

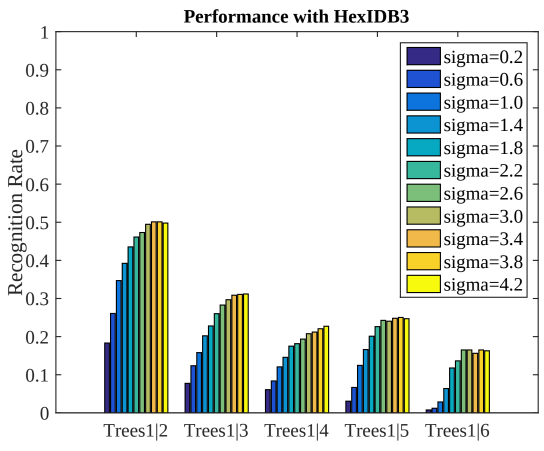
Figure 6.7: *RecognitionRate* performance of the third level descriptors: *HexIDB3* and *HexLDB3*, with the changes of the standard deviation of the Gaussian smoothing kernel. Each image pair has 11 colour bars to represent the correct matching rates with 11 different sigmas. The matching difficulty of the pairs is increasing from left to right of each figure. The left column shows the performances with *HexIDB3*. The right column shows the performances with *HexLDB3*.



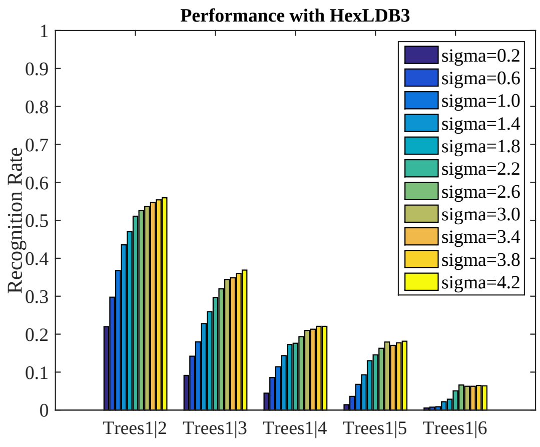
(a)



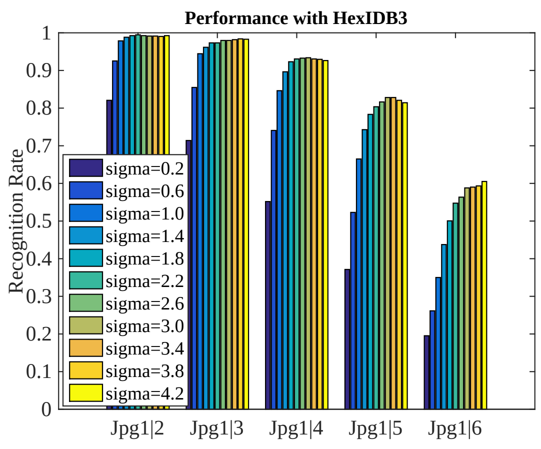
(b)



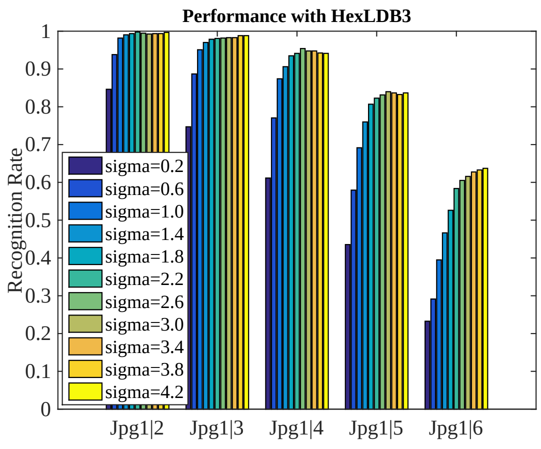
(c)



(d)

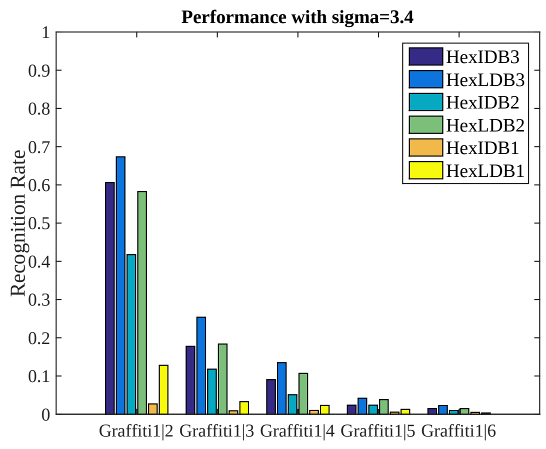


(e)

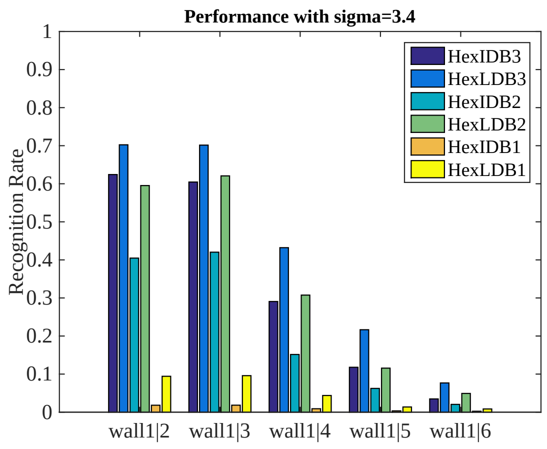


(f)

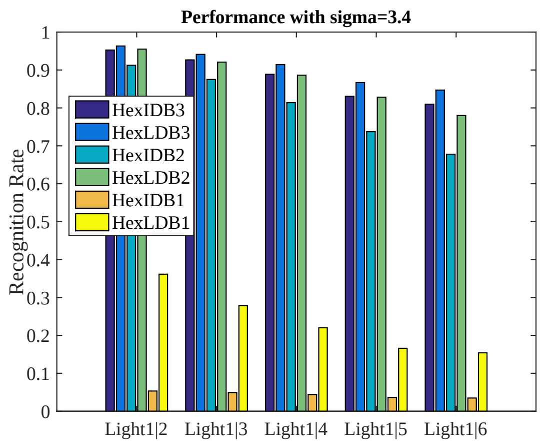
Figure 6.8: $RecognitionRate$ performance of the third level descriptors: *HexIDB3* and *HexLDB3*, with the changes of the standard deviation of the Gaussian smoothing kernel. Each image pair has 11 colour bars to represent the correct matching rates with 11 different sigmas. The matching difficulty of the pairs is increasing from left to right of each figure. The left column shows the performances with *HexIDB3*. The right column shows the performances with *HexLDB3*.



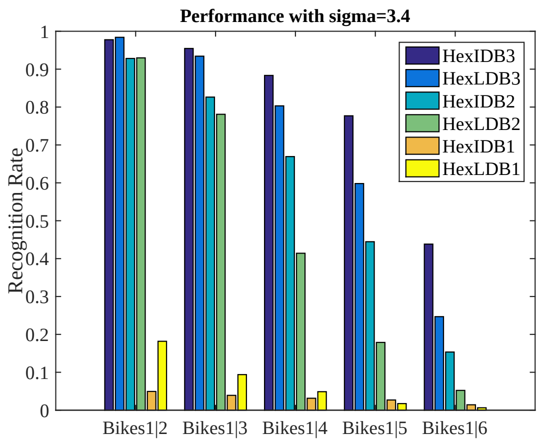
(a)



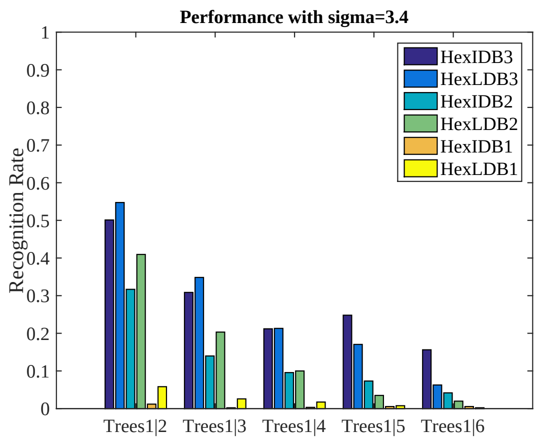
(b)



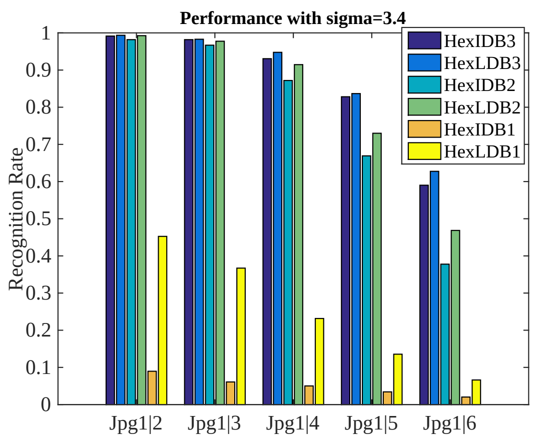
(c)



(d)

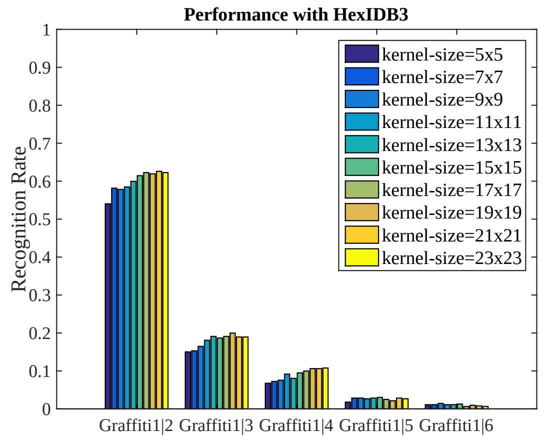


(e)

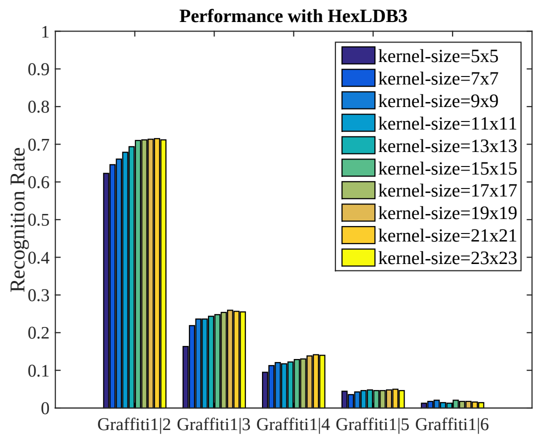


(f)

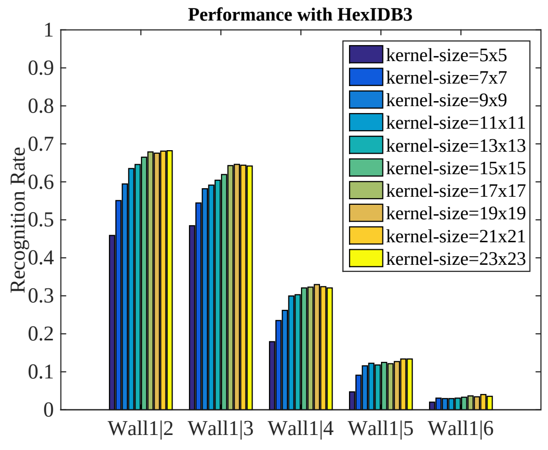
Figure 6.9: $RecognitionRate$ performance with Gaussian sigma=3.4. For each image pair, the performances of 3 different levels of *HexIDB* and *HexLDB* descriptors are illustrated.



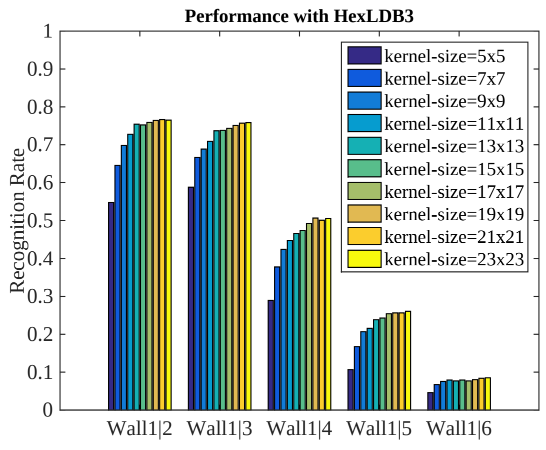
(a)



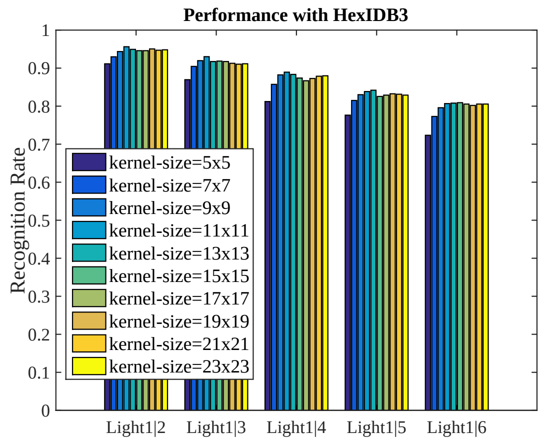
(b)



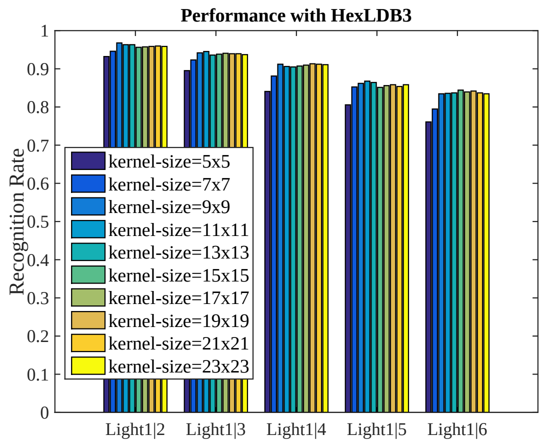
(c)



(d)

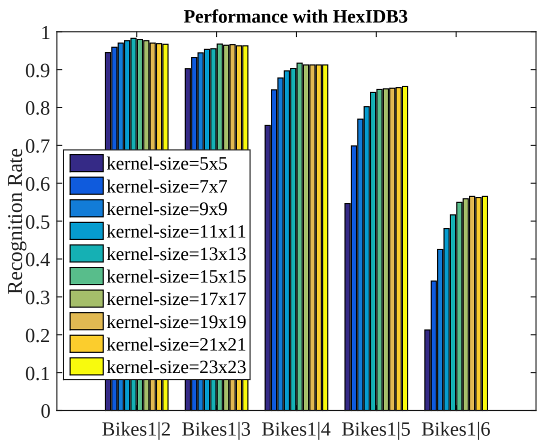


(e)

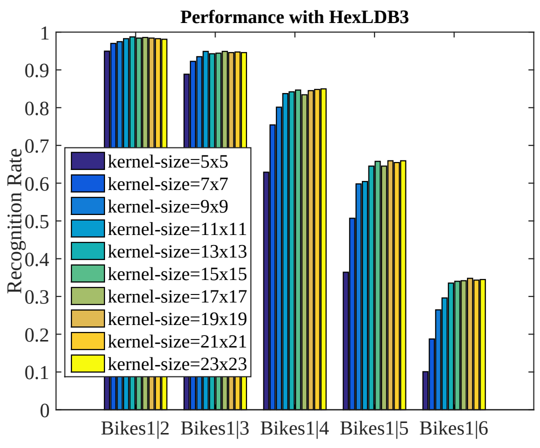


(f)

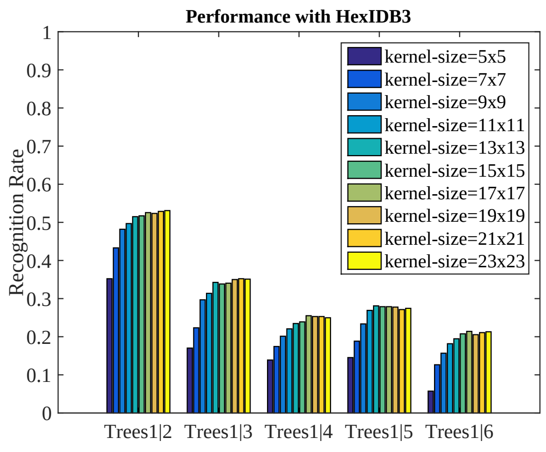
Figure 6.10: $RecognitionRate$ performance of the third level descriptors: *HexIDB3* and *HexLDB3*, with $\sigma = 3.4$ and the kernel size varying. Each image pair has 10 colour bars to represent the correct matching rates with 10 different kernel sizes. The matching difficulty of the pairs is increasing from left to right of each figure. The left column shows the performances with *HexIDB3*. The right column shows the performances with *HexLDB3*.



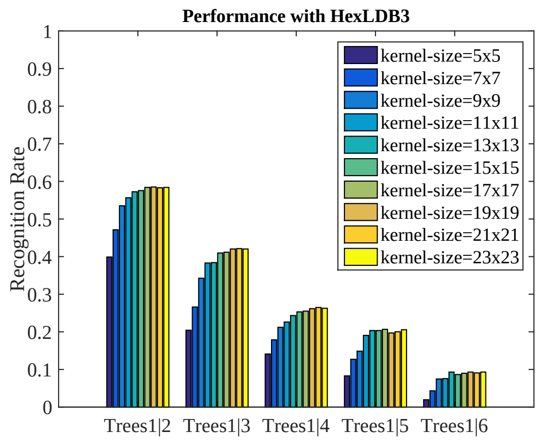
(a)



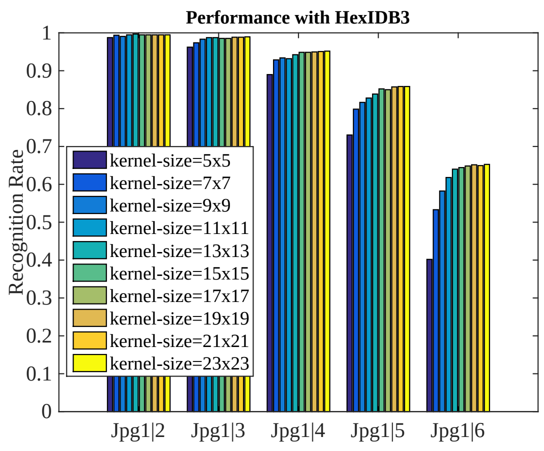
(b)



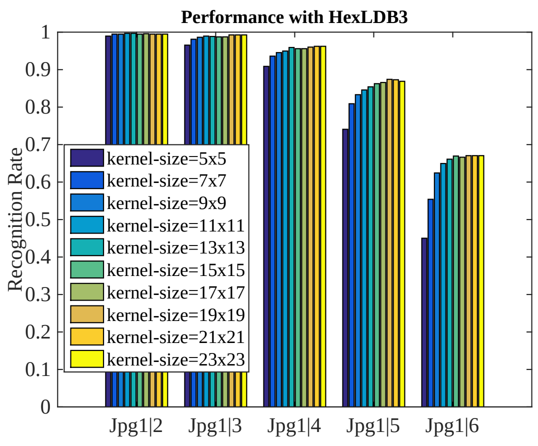
(c)



(d)

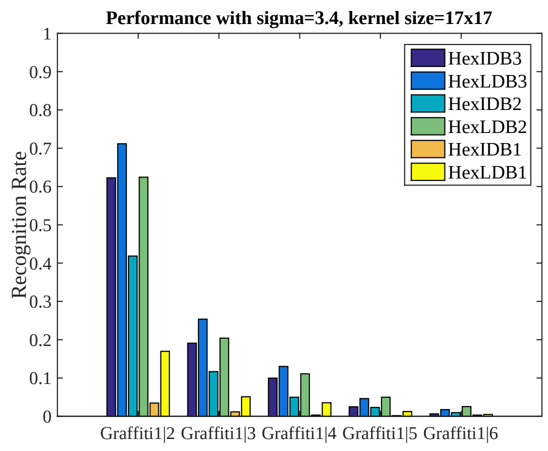


(e)

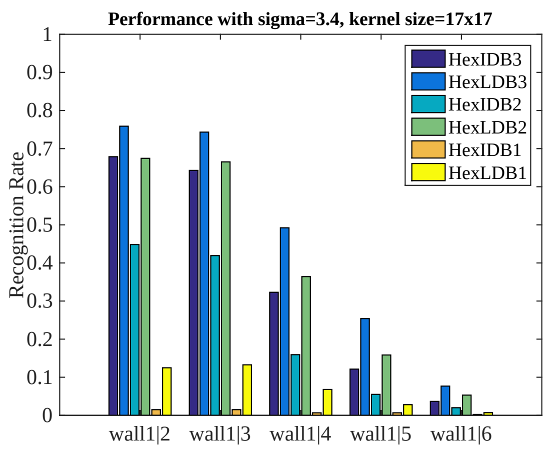


(f)

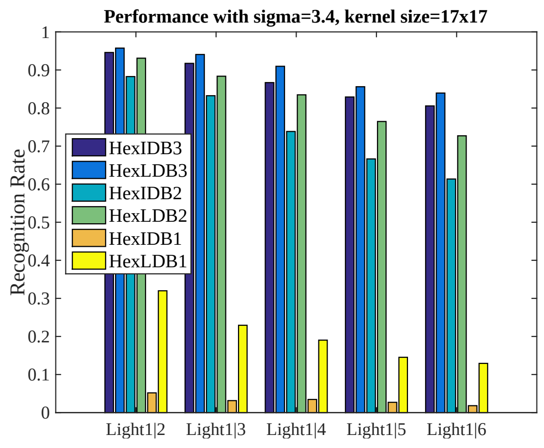
Figure 6.11: $RecognitionRate$ performance of the third level descriptors: *HexIDB3* and *HexLDB3*, with $\sigma = 3.4$ and the kernel size varying. Each image pair has 10 colour bars to represent the correct matching rates with 10 different kernel sizes. The matching difficulty of the pairs is increasing from left to right of each figure. The left column shows the performances with *HexIDB3*. The right column shows the performances with *HexLDB3*.



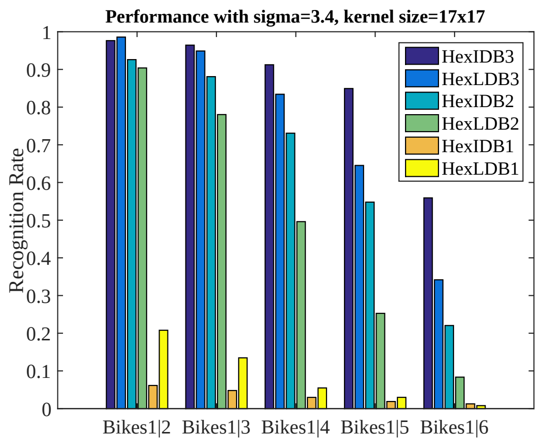
(a)



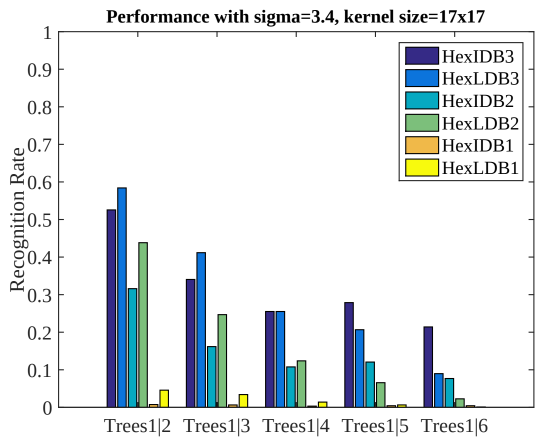
(b)



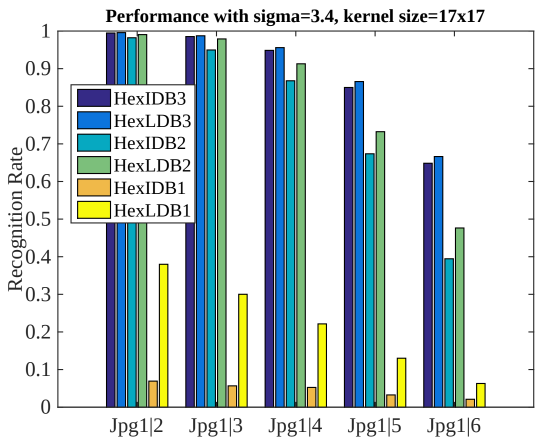
(c)



(d)

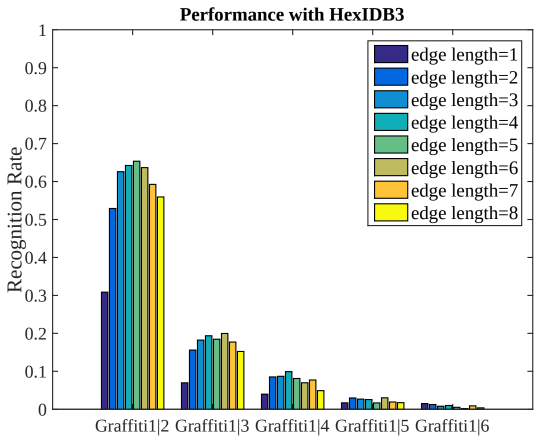


(e)

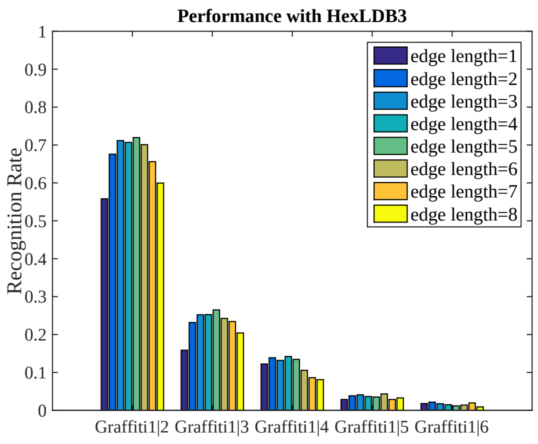


(f)

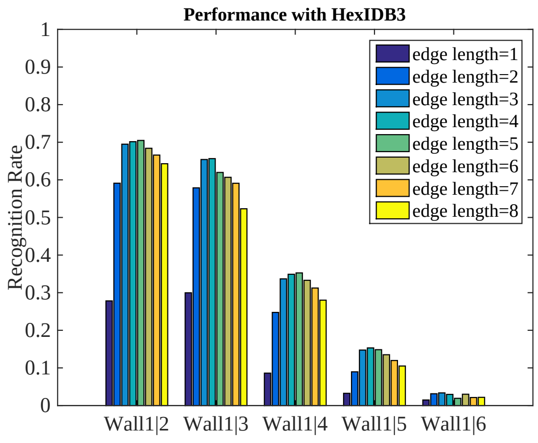
Figure 6.12: $RecognitionRate$ performance with Gaussian sigma=3.4, kernel size=17×17. For each image pair, the performances of 3 different levels of *HexIDB* and *HexLDB* descriptors are illustrated.



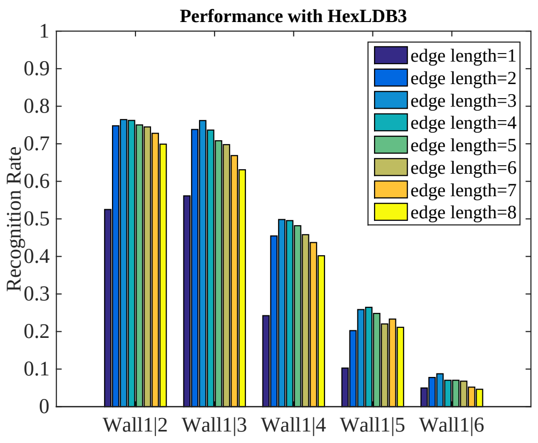
(a)



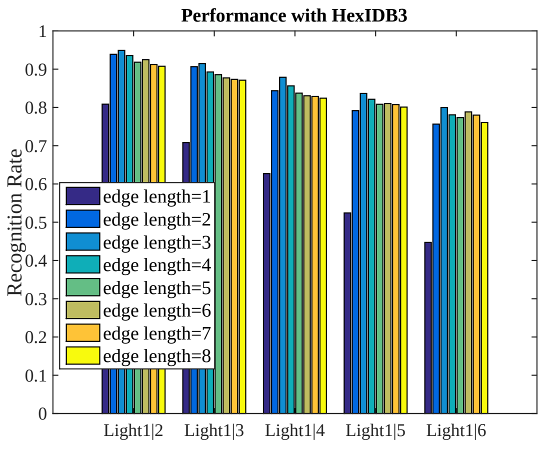
(b)



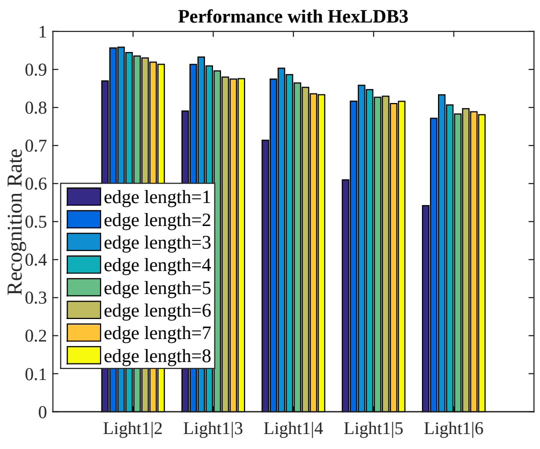
(c)



(d)



(e)



(f)

Figure 6.13: $RecognitionRate$ performance of the third level descriptors: *HexIDB3* and *HexLDB3*, with $\sigma = 3.4$ and the kernel size $= 17 \times 17$. There are 8 different colour bars representing the hexagon edge length changing in the range of [1,8]. The matching difficulty of the pairs is increasing from left to right of each figure. The left column shows the performances with *HexIDB3*. The right column shows the performances with *HexLDB3*.

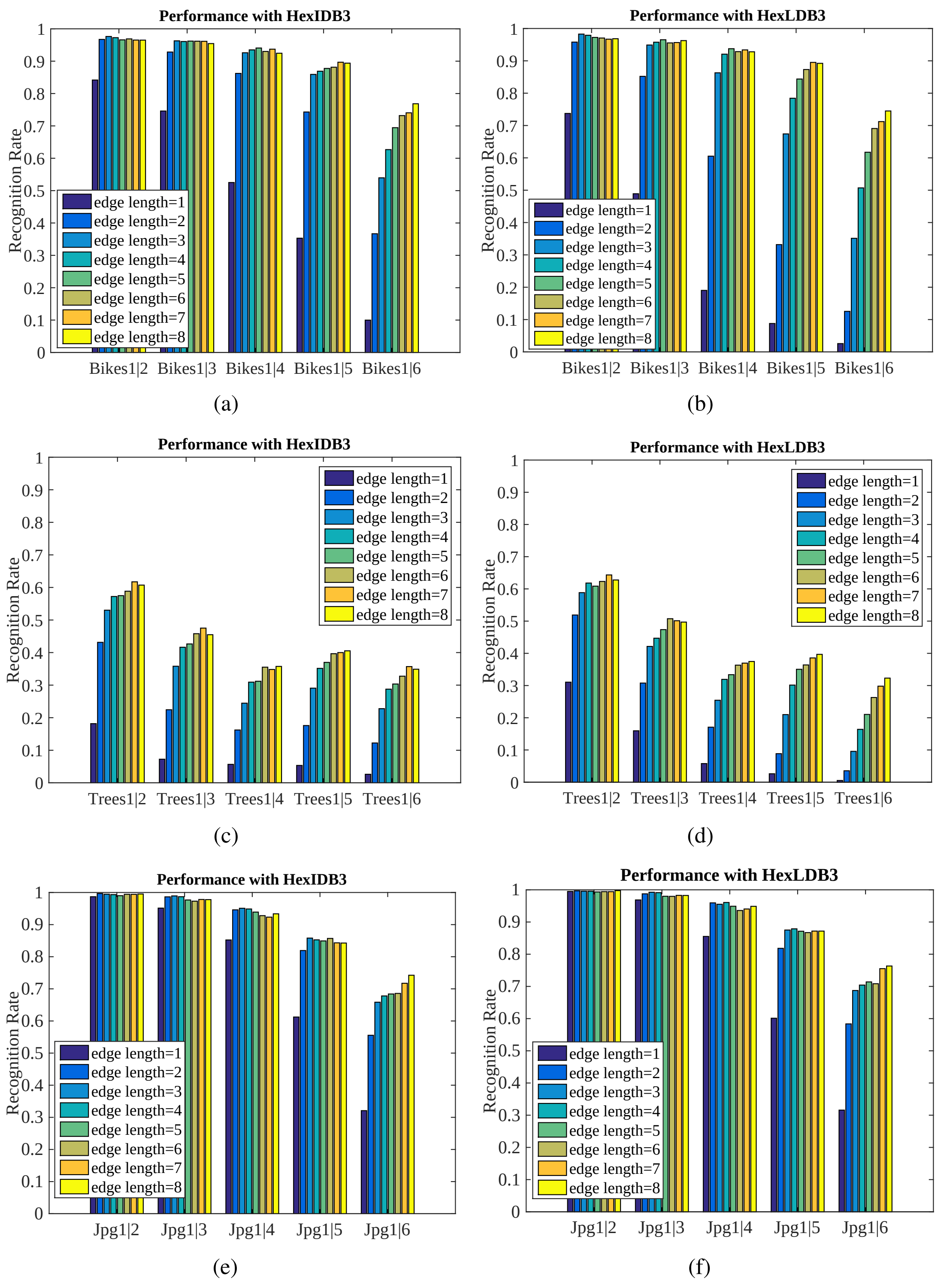
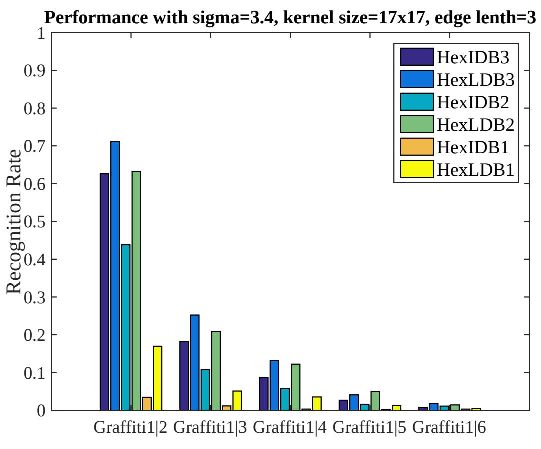
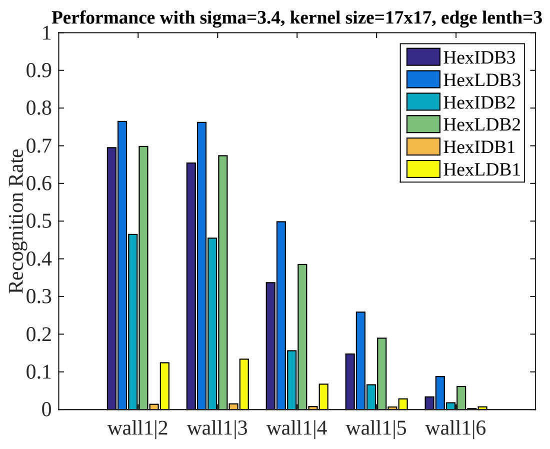


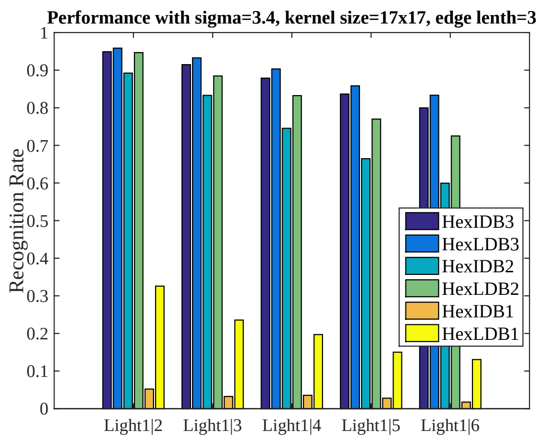
Figure 6.14: $RecognitionRate$ performance of the third level descriptors: *HexIDB3* and *HexLDB3*, with $\sigma = 3.4$ and the kernel size $= 17 \times 17$. There are 8 different colour bars representing the hexagon edge length changing in the range of [1,8]. The matching difficulty of the pairs is increasing from left to right of each figure. The left column shows the performances with *HexIDB3*. The right column shows the performances with *HexLDB3*.



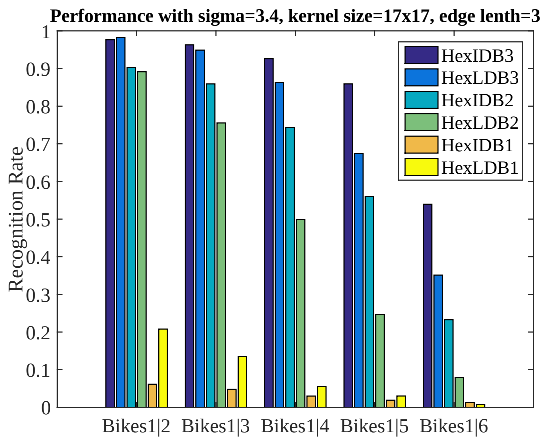
(a)



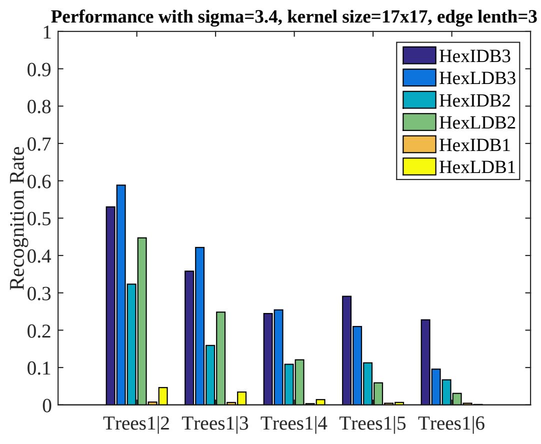
(b)



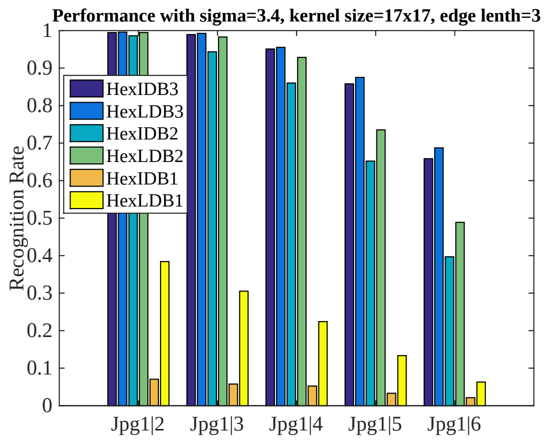
(c)



(d)

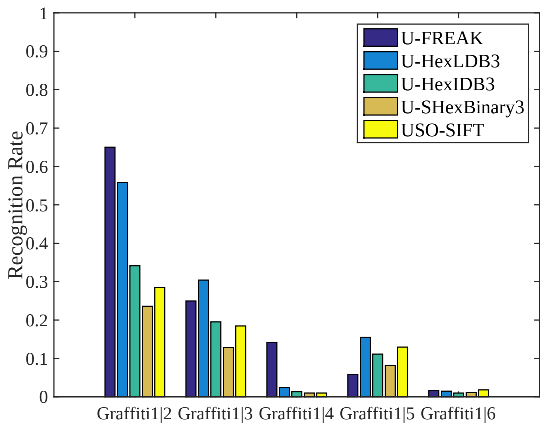


(e)

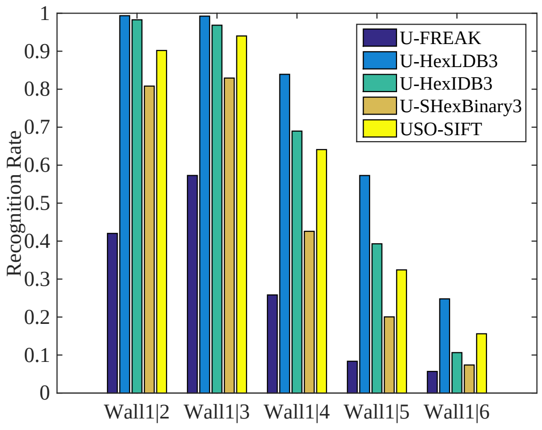


(f)

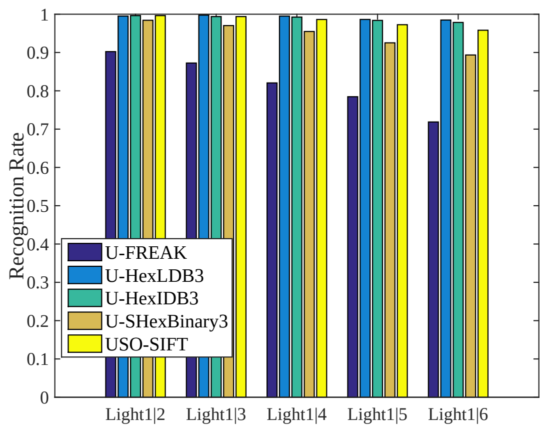
Figure 6.15: $RecognitionRate$ performance with Gaussian sigma=3.4, kernel size=17×17, hexagon edge length=3. For each image pair, the performances of 3 different levels of *HexIDB* and *HexLDB* descriptors are illustrated.



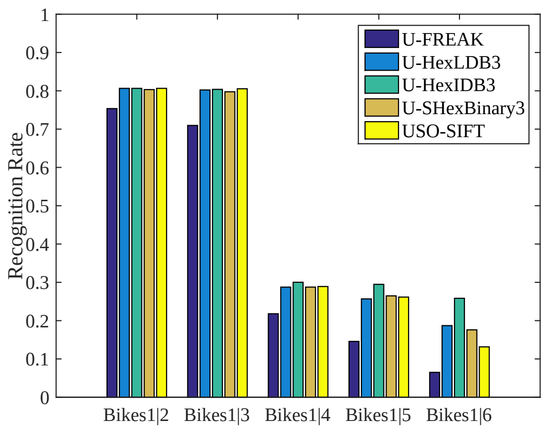
(a)



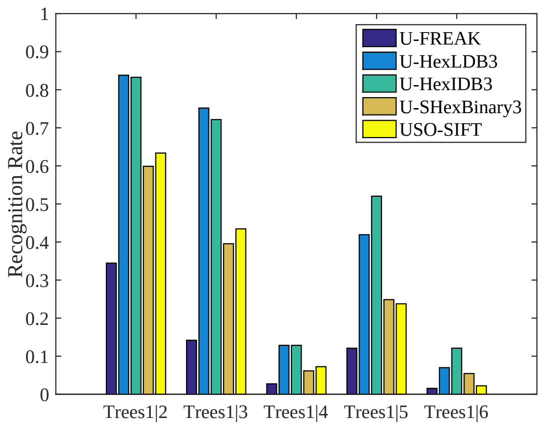
(b)



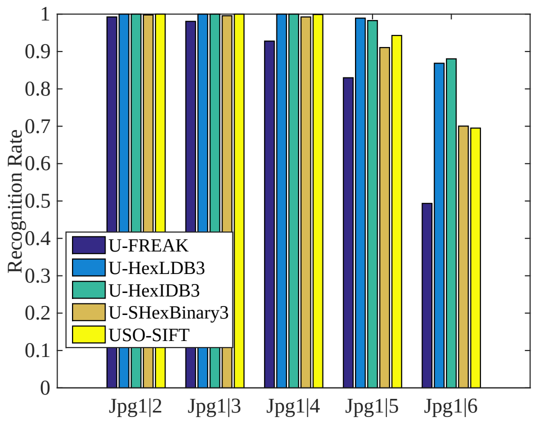
(c)



(d)



(e)



(f)

Figure 6.16: The $RecognitionRate$ performance for each image pair with utilising different local feature descriptors.

Chapter 7

Conclusion and Future Work

In this chapter, the research in this thesis is brought to a conclusion with the achievements and limitations addressed. In addition, future work based on this thesis is also presented.

7.1 Thesis Objectives

The main objective of this thesis is to investigate the method to localise the object in the image with the edge contours labelled. Though matching linear edge contour features provides a straightforward method to label the detected object, linear edge contour detection is a difficult task especially for cluttered natural images. To avoid the influence of linear edge contour detection and reduce the interruption from the image background, edge contour points rather than linear edge contours can be used for localising objects, as an alternative method to utilise the diagnostic feature of the object shape. Edge detection is a non-trivial problem in digital image analysis, however, it is simpler to detect than linear edge contours. Accordingly, new feature descriptors that are rotation-invariant and fast to compute of edge contour points must also be investigated. As a consequence, the main objective of this thesis is expanded into the following objectives:

- Extract local feature descriptors which can be fast to compute and robust to represent local edge features;
- Construct an object localisation system with labelling the edge contour points of the detected object in the image.

Inspired from the successful applications of hierarchical feature extraction and hexagonal structure sampling for image analysis, advantages of rotational symmetry and sampling efficiency of the hexagonal structure are utilised in this thesis to generate hierarchical feature descriptors with distinctiveness and rotation invariance. Moreover, a 2D object localisation system based on matching features of edge contour points is also proposed in this thesis.

7.2 Contributions

Based on the work to achieve the thesis objectives, the main contributions of this thesis can be summarised as follows:

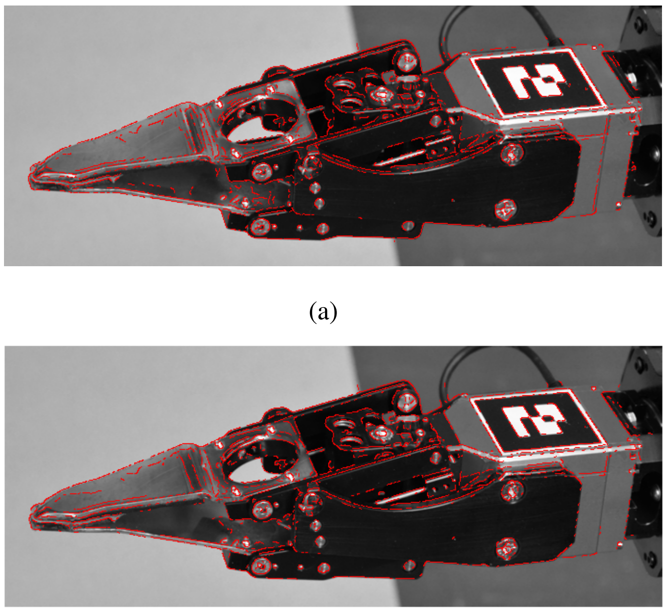
- Two different kinds of hierarchical grouping structure are proposed by sampling local patches in the hexagonal structure, which can be combined with different coding methods to generate appropriate hierarchical descriptors;
- Two different feature coding methods: histogram of oriented gradients and binary string, are employed to combine with the two proposed hexagonal structures, and generate a set of rotation-invariant hierarchical feature descriptors that can be applied at both edges and corners;
- A new method to achieve rotation invariance for local feature descriptors is proposed, by taking advantage of rotational symmetry of the hexagonal structure rather than prerotating and resampling the feature receptive field, which improves the computing efficiency of generating rotation-invariant feature descriptors;
- A 2D object localisation system is introduced by matching features of edge contour points rather than shape contours in a constrained search area, with a fine-to-coarse matching mechanism, which can decrease the false matching rate associated with directly matching edge points from the model to the unknown images.

7.3 Future Work

According to the work presented in this thesis, there are still some challenges left that can be further investigated. The limitations and potential extensions of the work are summarised as follows:

7.3.1 Out-of-Plane Application

The 2D object localisation system is currently limited for the in-plane application. It can also be extended to localise the edge contours of objects having out-of-plane transformations. An initial investigation of the robot's gripper localisation is implemented based on the system introduced in this thesis, and the results are illustrated in Figure 7.1. The two images show the results of edge labelling for the gripper under 3° and 5° of out-of-plane rotation, respectively. Although the proposed system has not been specifically designed to be invariant to out-of-plane rotations, it is still able to make a reasonable attempt at matching and localising the grippers edge contours when the appearance of the gripper has been deformed within a small range of pixels. In order to deal with better the out-of-plane applications, more of the view-sphere of the target object should be sampled to construct an appropriate set of training data representing the set of expected prior poses. How densely the view-sphere needs to be sampled is dependent on the base-line between sampled views and the discriminability of the employed features. If the projective 3D pose transformation between views can be estimated accurately, then this can serve to constrain the match search area as in the in-plane search currently employed in this thesis.



(a)

(b)

Figure 7.1: Edge contour localisation for the out-of-plane application: (a) a robot gripper under 3° of out-of-plane rotation ; (b) a robot gripper with 5° of out-of-plane rotation.

7.3.2 2D Object Localisation Using Features of Pure Edge Contours

The 2D object localisation system can also be implemented by employing features of pure edge contours. The proposed descriptors in this thesis are actually region-based and are inevitably distracted by background information when they are sampled at the object boundaries. The straightforward method to reduce the interference from background is to use features only containing pure edge information, though this should also rely on the detection result of pure edge contours.

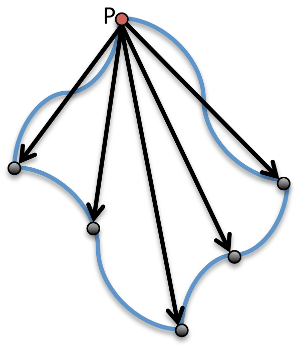


Figure 7.2: Compute a shape context: pick out all the edge points of a contour, and then for the edge point p on the contour, compute its relative distance and angle to all the other points to generate a histogram, which is defined as the shape context of point p.

To give an initial investigation of representing local features by using pure edge information, *shape context* [Belongie et al., 2002] is employed in the localisation system to represent the features extracted at edge contour points. The main idea of computing a shape context descriptor is concluded in Figure 7.2 and the localisation results are illustrated in Figure 7.3. *HexLDB3* (introduced in Chapter 6) descriptor is used to estimate the pose transformation, which is good enough to give a projection of the reference object edge map into the test image. The projection is very close to the true target object silhouette as Figure 7.3 (d) shows. Consequently, the *shape context* descriptors are computed at the edge points in both the projected edge map and the test edge map. Only edge contour samples containing more than 7 edge points connecting to the sampling centre are considered to compute a *shape context* descriptor. After computing the *shape context* descriptor, the same pipeline for edge matching within a local search range is implemented as described in Section 5.3.2. If a matched point is defined as an object edge point, the corresponding contour fragment used to compute the *shape context* descriptor of that point is also defined as the contour of the object. The final labelling results are demonstrated in Figure 7.3 (e) (f). Though *shape context* used as a local descriptor tends to lose distinctiveness because it was originally designed to represent the global shape contour of an object, the performance of the edge labelling can still indicate that the proposed 2D object localisation system is sufficiently flexible to be integrated with different kinds of feature descriptors.

7.3.3 Develop A Semantic Contour Detector

In the example of Figure 7.3, one of the limitations to compute a stronger *shape context* descriptor is the contour length. Longer contours can provide better distinctiveness by computing more relative information between points. However, for natural images, it is challenging to achieve smoothly semantic contours. Though certain morphological operations, such as dilation and erosion, could fill breaks and connect contours, they cannot determine whether the new connected contour is a semantically meaningful contour from the target object or from a noisy background, which might decrease the detection accuracy if the connected contour comprises contours partly from the background and partly from the target object. Therefore, developing a semantic contour detector can provide a much better condition to generate distinctive linear contour features, which can be significant for object localisation.

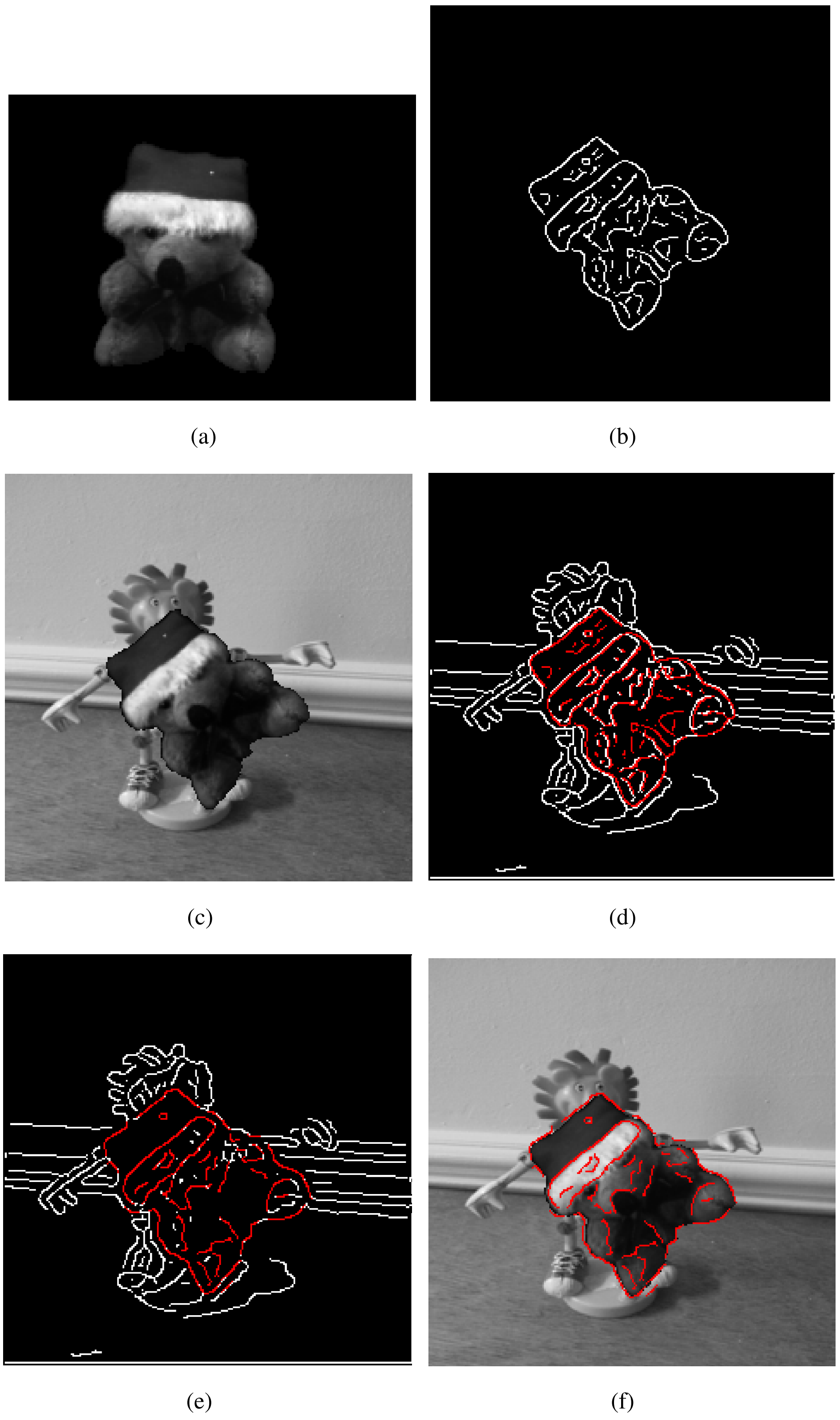


Figure 7.3: An example of object localisation based on *shape context* feature matching: (a) is the reference image. (b) is the reference edge map projected into the test image coordinate according to the pose estimate result. (c) is the test image, (d) is the test edge map, where the red points are from (b). (e) and (f) are the labelling results shown in the test edge map and the test image, respectively, where the red points are the labelled edge points of the target object.

7.3.4 Develop A Hierarchically Grouped Descriptor Based on Pure Edge Contours

As discussed previously, to better represent the edge contour structure and reduce the background disturbance at the object boundaries, a descriptor computed from the pure edge information is necessary. A hierarchical structure is still preferred to represent the object contours better from locally to globally, which is then potentially able to deal with object occlusions. Generating hierarchical descriptors based on grouping mechanism of local regions has the potential to make the descriptor more distinctive, as has been demonstrated in computer vision community. A simple example is shown in Figure 7.4. Therefore, a hierarchical grouping descriptor extracted using pure edge contour information could provide a powerful representation for object edge features, though this will be dependent on the capability of detecting semantic contours.

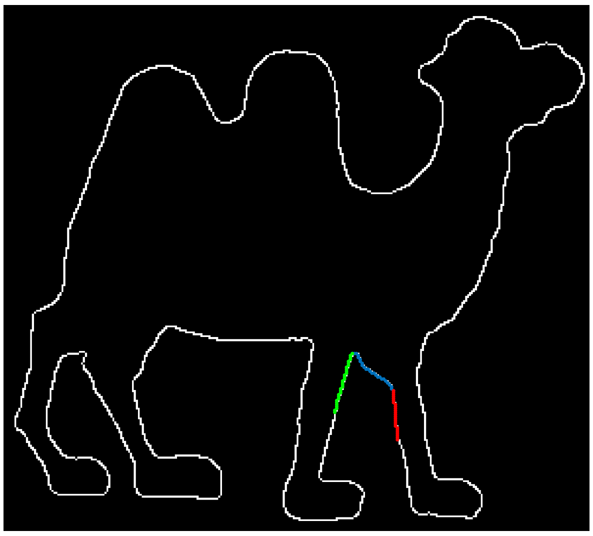


Figure 7.4: A simple example of edge contour grouping: the grouping of the red, blue, and green contours can make a more distinctive feature descriptor than any of these contours considered individually.

7.3.5 Collect A Data Set of Natural Images with Scale Changes

The reference dataset used in this thesis for evaluation is derived from ALOI. The test dataset is generated by manually compositing the reference images with background images. In these synthetic images, the target object boundary is affected by artefact edges generated during the compositing process, which can affect the edge detection results, and consequently influence the performance of object localisation. Therefore, a new set of natural images without artefact edges is preferred to evaluate the object localisation system. Moreover, no scale issue is considered in this thesis. However, in the real world, objects are composed of different structures at different scales. Extending the proposed object localisation system in multi-scale space would open more practical applications.

7.3.6 Reduce the Feature Dimension

The proposed hexagonally structured hierarchical descriptors (HHDs) are generated by means of a hierarchically overlapped grouping mechanism, which leads to better robustness, but longer feature vectors and redundant information. To achieve shorter descriptors for lower storage requirement and faster matching, the proposed HHDs can employ a dimensionality reduction method such as PCA, or the bit selection methods introduced in LDB [Yang and Cheng, 2012]: random bit selection and entropy-based method, which results in the final feature vectors formed by a set of least correlated values with better feature distinctiveness and matching efficiency.

7.3.7 Develop A Ternary Descriptor

There are three semantically significant states in an image for derivative measurement: positive gradient, negative gradient and no gradient. However, binary coding method does not encode the ”no gradient” state. When there is no significant difference in a local region of an image, binary coding based on two bit values: 0 and 1, can not distinguish this region from other regions that have significant difference. Therefore, if there is a threshold T that sets the level of change perception, then the regions having positive gradient, negative gradient and no gradient can be respectively encoded with values 1, -1, and 0. Therefore, developing a ternary descriptor based on the hierarchical grouping mechanism is a potential extension of the current work.

Bibliography

Agarwal, A. and Triggs, B. (2006). Hyperfeatures–multilevel local coding for visual recognition. In *Computer Vision–ECCV 2006*, pages 30–43. Springer.

Agrawal, M., Konolige, K., and Blas, M. R. (2008). Censure: Center surround extremas for realtime feature detection and matching. In *Computer Vision–ECCV 2008*, pages 102–115. Springer.

Alahi, A., Ortiz, R., and Vandergheynst, P. (2012). Freak: Fast retina keypoint. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 510–517. IEEE.

Alcantarilla, P. F., Bartoli, A., and Davison, A. J. (2012). Kaze features. In *Computer Vision–ECCV 2012*, pages 214–227. Springer.

Alcantarilla, P. F., Nuevo, J., and Bartoli, A. (2013). Fast explicit diffusion for accelerated features in nonlinear scale spaces. In *British Machine Vision Conf. (BMVC)*.

Arbelaez, P., Maire, M., Fowlkes, C., and Malik, J. (2011). Contour detection and hierarchical image segmentation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 33(5):898–916.

Ballard, D. H. (1981). Generalizing the hough transform to detect arbitrary shapes. *Pattern recognition*, 13(2):111–122.

Bay, H., Tuytelaars, T., and Van Gool, L. (2006). Surf: Speeded up robust features. In *Computer Vision–ECCV 2006*, pages 404–417. Springer.

Belongie, S., Malik, J., and Puzicha, J. (2002). Shape matching and object recognition using shape contexts. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(4):509–522.

Blaschko, M. B. and Lampert, C. H. (2008). Learning to localize objects with structured output regression. In *Computer Vision–ECCV 2008*, pages 2–15. Springer.

Bo, L., Lai, K., Ren, X., and Fox, D. (2011). Object recognition with hierarchical kernel descriptors. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 1729–1736. IEEE.

Bo, L., Ren, X., and Fox, D. (2010). Kernel descriptors for visual recognition. In *Advances in Neural Information Processing Systems*, pages 244–252.

Borenstein, E. and Ullman, S. (2002). Class-specific, top-down segmentation. pages 109–122. Springer.

Brown, M., Hua, G., and Winder, S. (2011). Discriminative learning of local image descriptors. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 33(1):43–57.

Calonder, M., Lepetit, V., Strecha, C., and Fua, P. (2010). Brief: binary robust independent elementary features. In *Computer Vision–ECCV 2010*, pages 778–792. Springer.

Chandrasekhar, V., Takacs, G., Chen, D., Tsai, S., Grzeszczuk, R., and Girod, B. (2009). Chog: Compressed histogram of gradients a low bit-rate feature descriptor. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 2504–2511. IEEE.

Cootes, T. F., Edwards, G. J., and Taylor, C. J. (2001). Active appearance models. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 23(6):681–685.

Curcio, C. A., Sloan, K. R., Packer, O., Hendrickson, A. E., and Kalina, R. E. (1987). Distribution of cones in human and monkey retina: individual variability and radial asymmetry. *Science*, 236(4801):579–582.

Dai, Q. and Hoiem, D. (2012). Learning to localize detected objects. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3322–3329. IEEE.

Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893. IEEE.

Deselaers, T., Keysers, D., and Ney, H. (2008). Features for image retrieval: an experimental comparison. *Information Retrieval*, 11(2):77–107.

Dollar, P., Wojek, C., Schiele, B., and Perona, P. (2012). Pedestrian detection: An evaluation of the state of the art. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(4):743–761.

Epshtein, B. and Ullman, S. (2005). Feature hierarchies for object classification. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 1, pages 220–227. IEEE.

Epshtein, B. and Ullman, S. (2007). Semantic hierarchies for recognizing objects and parts. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE.

Felzenszwalb, P. F., Girshick, R. B., McAllester, D., and Ramanan, D. (2010). Object detection with discriminatively trained part-based models. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(9):1627–1645.

Fergus, R., Perona, P., and Zisserman, A. (2003). Object class recognition by unsupervised scale-invariant learning. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 2, pages II–264. IEEE.

Ferrari, V., Fevrier, L., Jurie, F., and Schmid, C. (2008). Groups of adjacent contour segments for object detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(1):36–51.

Ferrari, V., Jurie, F., and Schmid, C. (2010). From images to shape models for object detection. *International Journal of Computer Vision*, 87(3):284–303.

Feyereisl, J., Kwak, S., Son, J., and Han, B. (2014). Object localization based on structural svm using privileged information. In *Advances in Neural Information Processing Systems*, pages 208–216.

Finckh, M., Dammertz, H., and Lensch, H. P. (2014). On near optimal lattice quantization of multi-dimensional data points. In *Computer Graphics Forum*, volume 33, pages 271–281. Wiley Online Library.

Fischler, M. A. and Bolles, R. C. (1981). Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395.

Fitz, A. and Green, R. (1996). Fingerprint classification using a hexagonal fast fourier transform. *Pattern recognition*, 29(10):1587–1597.

Friedman, M. (1940). A comparison of alternative tests of significance for the problem of m rankings. *The Annals of Mathematical Statistics*, 11(1):86–92.

Gardiner, B., Coleman, S., and Scotney, B. (2008). Multi-scale feature extraction in a sub-pixel virtual hexagonal environment. In *Machine Vision and Image Processing Conference, 2008. IMVIP'08. International*, pages 111–116. IEEE.

Geusebroek, J.-M., Burghouts, G. J., and Smeulders, A. W. (2005). The amsterdam library of object images. *International Journal of Computer Vision*, 61(1):103–112.

Grabner, H., Nguyen, T. T., Gruber, B., and Bischof, H. (2008). On-line boosting-based car detection from aerial images. *ISPRS Journal of Photogrammetry and Remote Sensing*, 63(3):382–396.

Gu, C., Lim, J. J., Arbeláez, P., and Malik, J. (2009). Recognition using regions. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 1030–1037. IEEE.

Hariharan, B., Arbeláez, P., Girshick, R., and Malik, J. (2014). Simultaneous detection and segmentation. In *Computer Vision–ECCV 2014*, pages 297–312. Springer.

Harris, C. and Stephens, M. (1988). A combined corner and edge detector. In *Alvey vision conference*, volume 15, page 50. Manchester, UK.

Hartman, N. P. and Tanimoto, S. L. (1984). A hexagonal pyramid data structure for image processing. *Systems, Man and Cybernetics, IEEE Transactions on*, (2):247–256.

He, X. and Jia, W. (2005). Hexagonal structure for intelligent vision. In *Information and Communication Technologies, 2005. ICICT 2005. First International Conference on*, pages 52–64. IEEE.

He, X., Li, J., Chen, Y., Wu, Q., and Jia, W. (2007). Local binary patterns for human detection on hexagonal structure. In *Multimedia, 2007. ISM 2007. Ninth IEEE International Symposium on*, pages 65–71. IEEE.

Heitz, G., Elidan, G., Packer, B., and Koller, D. (2009). Shape-based object localization for descriptive classification. *International journal of computer vision*, 84(1):40–62.

Her, I. and Yuan, C.-T. (1994). Resampling on a pseudohexagonal grid. *CVGIP: Graphical Models and Image Processing*, 56(4):336–347.

Hinton, G., Osindero, S., and Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554.

Hubel, D. H., Wensveen, J., and Wick, B. (1995). *Eye, brain, and vision*. Scientific American Library New York.

Ikizler-Cinbis, N. and Sclaroff, S. (2010). Object recognition and localization via spatial instance embedding. In *Pattern Recognition (ICPR), 2010 20th International Conference on*, pages 452–455. IEEE.

Kamgar-Parsi, B., Sander, W., et al. (1989). Quantization error in spatial sampling: comparison between square and hexagonal pixels. In *Computer Vision and Pattern Recognition, 1989. Proceedings CVPR'89., IEEE Computer Society Conference on*, pages 604–611. IEEE.

Ke, Y. and Sukthankar, R. (2004). Pca-sift: A more distinctive representation for local image descriptors. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 2, pages II–506. IEEE.

Kokkinos, I. and Yuille, A. (2009). Hop: Hierarchical object parsing. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 802–809. IEEE.

Lampert, C. H., Blaschko, M. B., and Hofmann, T. (2008). Beyond sliding windows: Object localization by efficient subwindow search. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE.

Lee, H., Ekanadham, C., and Ng, A. Y. (2008). Sparse deep belief net model for visual area v2. In *Advances in neural information processing systems*, pages 873–880.

Lee, H., Grosse, R., Ranganath, R., and Ng, A. Y. (2009). Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 609–616. ACM.

Lee, H., Grosse, R., Ranganath, R., and Ng, A. Y. (2011). Unsupervised learning of hierarchical representations with convolutional deep belief networks. *Communications of the ACM*, 54(10):95–103.

Leibe, B., Leonardis, A., and Schiele, B. (2004). Combined object categorization and segmentation with an implicit shape model. In *Workshop on statistical learning in computer vision, ECCV*, volume 2, page 7.

Leibe, B., Leonardis, A., and Schiele, B. (2008). Robust object detection with interleaved categorization and segmentation. *International journal of computer vision*, 77(1-3):259–289.

Leonardis, A. and Fidler, S. (2011). *Learning hierarchical representations of object categories for robot vision*. Springer.

Leutenegger, S., Chli, M., and Siegwart, R. Y. (2011). Brisk: Binary robust invariant scalable keypoints. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2548–2555. IEEE.

Liu, Y. and Siebert, J. P. (2014). Contour localization based on matching dense hexhog descriptors. In *International Conference on Computer Vision Theory and Applications (VISAPP 2014)*, pages 656–666.

Lowe, D. G. (1999). Object recognition from local scale-invariant features. In *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, volume 2, pages 1150–1157. Ieee.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110.

Mersereau, R. M. (1979). The processing of hexagonally sampled two-dimensional signals. *Proceedings of the IEEE*, 67(6):930–949.

M.H.Pirenne (1967). *Vision and the Eye*. Chapman and Hall.

Middleton, L. and Sivaswamy, J. (2006). *Hexagonal image processing: A practical approach*. Springer Science & Business Media.

Mikolajczyk, K. and Schmid, C. (2005). A performance evaluation of local descriptors. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(10):1615–1630.

Mikolajczyk, K., Tuytelaars, T., Schmid, C., Zisserman, A., Matas, J., Schaffalitzky, F., Kadir, T., and Van Gool, L. (2005). A comparison of affine region detectors. *International journal of computer vision*, 65(1-2):43–72.

Mohan, A., Papageorgiou, C., and Poggio, T. (2001). Example-based object detection in images by components. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 23(4):349–361.

Okada, R. (2009). Discriminative generalized hough transform for object dectection. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 2000–2005. IEEE.

Opelt, A., Pinz, A., and Zisserman, A. (2006). A boundary-fragment-model for object detection. In *Computer Vision–ECCV 2006*, pages 575–588. Springer.

Overington, I. (1992). *Computer Vision: A unified, biologically-inspired approach*. Elsevier Science Inc.

Papageorgiou, C. and Poggio, T. (2000). A trainable system for object detection. *International Journal of Computer Vision*, 38(1):15–33.

Parkhi, O. M., Vedaldi, A., Jawahar, C., and Zisserman, A. (2011). The truth about cats and dogs. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 1427–1434. IEEE.

Perrett, D. I. and Oram, M. W. (1993). Neurophysiology of shape processing. *Image and Vision Computing*, 11(6):317–333.

Piccinini, P., Prati, A., and Cucchiara, R. (2012). Real-time object detection and localization with sift-based clustering. *Image and Vision Computing*, 30(8):573–587.

Porway, J., Wang, Q., and Zhu, S. C. (2010). A hierarchical and contextual model for aerial image parsing. *International journal of computer vision*, 88(2):254–283.

Riesenhuber, M. and Poggio, T. (1999). Hierarchical models of object recognition in cortex. *Nature neuroscience*, 2(11):1019–1025.

Rosin, P. L. (1999). Measuring corner properties. *Computer Vision and Image Understanding*, 73(2):291–307.

Rosten, E. and Drummond, T. (2005). Fusing points and lines for high performance tracking. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 2, pages 1508–1515. IEEE.

Rosten, E. and Drummond, T. (2006). Machine learning for high-speed corner detection. In *Computer Vision–ECCV 2006*, pages 430–443. Springer.

Rosten, E., Porter, R., and Drummond, T. (2010). Faster and better: A machine learning approach to corner detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(1):105–119.

Rowley, H., Baluja, S., Kanade, T., et al. (1998). Neural network-based face detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 20(1):23–38.

Rowley, H. A., Baluja, S., Kanade, T., et al. (1995). *Human face detection in visual scenes*. Carnegie-Mellon University. Department of Computer Science.

Rublee, E., Rabaud, V., Konolige, K., and Bradski, G. (2011). Orb: an efficient alternative to sift or surf. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2564–2571. IEEE.

Schlecht, J. and Ommer, B. (2011). Contour-based object detection. In *Proceedings of the British Machine Vision Conference. BVA Press*.

Schneiderman, H. and Kanade, T. (2000). A statistical method for 3d object detection applied to faces and cars. In *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, volume 1, pages 746–751. IEEE.

Serrano, J., Larlus, D., et al. (2013). Predicting an object location using a global image representation. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 1729–1736. IEEE.

Serre, T., Wolf, L., and Poggio, T. (2005). Object recognition with features inspired by visual cortex. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 2, pages 994–1000. IEEE.

Shao, W., Yang, W., Liu, G., and Liu, J. (2012). Car detection from high-resolution aerial imagery using multiple features. In *Geoscience and Remote Sensing Symposium (IGARSS), 2012 IEEE International*, pages 4379–4382. IEEE.

Shima, T., Sugimoto, S., and Okutomi, M. (2010). Comparison of image alignment on hexagonal and square lattices. In *Image Processing (ICIP), 2010 17th IEEE International Conference on*, pages 141–144. IEEE.

Shotton, J., Blake, A., and Cipolla, R. (2005). Contour-based learning for object detection. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 1, pages 503–510. IEEE.

Siegel, S. (1956). Nonparametric statistics for the behavioral sciences.

Strecha, C., Bronstein, A. M., Bronstein, M. M., and Fua, P. (2012). Ldahash: Improved matching with smaller descriptors. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(1):66–78.

Tola, E., Lepetit, V., and Fua, P. (2008). A fast local descriptor for dense matching. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE.

Tola, E., Lepetit, V., and Fua, P. (2010). Daisy: An efficient dense descriptor applied to wide-baseline stereo. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(5):815–830.

Torr, P. H. and Murray, D. W. (1997). The development and comparison of robust methods for estimating the fundamental matrix. *International journal of computer vision*, 24(3):271–300.

Trzcinski, T. and Lepetit, V. (2012). Efficient discriminative projections for compact binary descriptors. In *Computer Vision–ECCV 2012*, pages 228–242. Springer.

Van De Ville, D., Blu, T., Unser, M., Philips, W., Lemahieu, I., and Van de Walle, R. (2004). Hex-splines: a novel spline family for hexagonal lattices. *Image Processing, IEEE Transactions on*, 13(6):758–772.

Van De Ville, D., Van de Walle, R., Philips, W., and Lemahieu, I. (2002). Image resampling between orthogonal and hexagonal lattices. In *Image Processing. 2002. Proceedings. 2002 International Conference on*, volume 3, pages III–389. IEEE.

VC, H. P. (1962). Method and means for recognizing complex patterns. US Patent 3,069,654.

Vidya, P., Veni, S., and Narayanankutty, K. (2009). Performance analysis of edge detection methods on hexagonal sampling grid. *International Journal of Electronic Engineering Research*, 1(4):313–328.

Viola, P. and Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–511. IEEE.

Viola, P. and Jones, M. J. (2004). Robust real-time face detection. *International journal of computer vision*, 57(2):137–154.

Watson, A. B. and Ahumada Jr, A. J. (1989). A hexagonal orthogonal-oriented pyramid as a model of image representation in visual cortex. *Biomedical Engineering, IEEE Transactions on*, 36(1):97–106.

Winder, S. A. and Brown, M. (2007). Learning local image descriptors. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE.

Wold, S., Esbensen, K., and Geladi, P. (1987). Principal component analysis. *Chemometrics and intelligent laboratory systems*, 2(1):37–52.

Yang, J. and Yang, M.-H. (2011). Learning hierarchical image representation with sparsity, saliency and locality. In *BMVC*, pages 1–11.

Yang, J., Yu, K., Gong, Y., and Huang, T. (2009). Linear spatial pyramid matching using sparse coding for image classification. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 1794–1801. IEEE.

Yang, M., Kpalma, K., and Ronsin, J. (2008). A survey of shape feature extraction techniques. *Pattern recognition*, pages 43–90.

Yang, X. and Cheng, K.-T. (2012). Ldb: An ultra-fast feature for scalable augmented reality on mobile devices. In *Mixed and Augmented Reality (ISMAR), 2012 IEEE International Symposium on*, pages 49–57. IEEE.

Yang, X., Liu, H., and Latecki, L. J. (2012). Contour-based object detection as dominant set computation. *Pattern Recognition*, 45(5):1927–1936.

Yeh, T.-H., Lee, J. J., and Darrell, T. (2009). Fast concurrent object localization and recognition. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 280–287. IEEE.

Yu, S. and Shi, J. (2003). Object-specific figure-ground segregation. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 2, pages II–39. IEEE.

Zhang, C. and Zhang, Z. (2010). A survey of recent advances in face detection. Technical report, Tech. rep., Microsoft Research.

Zhou, H., Yuan, Y., and Shi, C. (2009). Object tracking using sift features and mean shift. *Computer vision and image understanding*, 113(3):345–352.

Zhu, Q., Wang, L., Wu, Y., and Shi, J. (2008). Contour context selection for object detection: A set-to-set contour matching approach. In *Computer Vision–ECCV 2008*, pages 774–787. Springer.