ROTATION INVARIANT HISTOGRAM FEATURES FOR OBJECT DETECTION AND TRACKING IN AERIAL IMAGERY

Dissertation

Submitted to

The School of Engineering of the

UNIVERSITY OF DAYTON

In Partial Fulfillment of the Requirements for

The Degree of

Doctor of Philosophy in Electrical Engineering

By

Alex Mathew

UNIVERSITY OF DAYTON

Dayton, Ohio

May, 2014

ROTATION INVARIANT HISTOGRAM FEATURES FOR OBJECT DETECTION AND TRACKING IN AERIAL IMAGERY

Name: Mathew, Alex

APPROVED BY:

Vijayan K. Asari, Ph.D. Advisor Committee Chairman Professor, Department of Electrical and Computer Engineering	Keigo Hirakawa, Ph.D. Committee Member Assistant Professor, Department of Electrical and Computer Engineering
Raúl Ordóñez, Ph.D Committee Member Associate Professor, Department of Electrical and Computer Engineering	Youssef N. Raffoul, Ph.D Committee Member Professor, Department of Mathematics
John G. Weber, Ph.D. Associate Dean School of Engineering	Tony E. Saliba, Ph.D. Dean, School of Engineering & Wilke Distinguished Professor

ABSTRACT

ROTATION INVARIANT HISTOGRAM FEATURES FOR OBJECT DETECTION AND TRACKING IN AERIAL IMAGERY

Name: Mathew, Alex
University of Dayton

Advisor: Dr. Vijayan K. Asari

Object detection and tracking in imagery captured by aerial systems are becoming increasingly important in computer vision research. In aerial imagery, objects can appear in any orientation, varying sizes and in different lighting conditions. Due to the exponential growth in sensor technology, increasing size and resolution of aerial imagery are becoming a challenge for real-time computation. A rotation invariant feature extraction technique for detecting and tracking objects in aerial imagery is presented in this dissertation. Rotation invariance in the feature representation is addressed by considering concentric circular regions centered at visually salient locations of the object. The intensity distribution characteristics of the object region are used to represent an object effectively. A set of intensity-based features is derived from intensity histograms of the circular regions and they are inherently rotation invariant. An integral histogram computation approach is used to compute these features efficiently. To improve the representational strength of the feature set for rotation and illumination-invariant object detection, a gradient-based feature set is derived from normalized gradient orientation histograms of concentric regions. Rotation invariance is achieved by considering the magnitude of the Discrete Fourier Transform (DFT) of the gradient orientation histograms. A novel computational framework called Integral DFT is presented for fast and efficient extraction of gradient-based features in large imagery. A part-based model, which relies on a representation of an object as an aggregation of significant parts, using the gradient-based features is also presented in this dissertation. Integrating the features of significant parts gives robustness to partial occlusions and slight deformations, thus leading to a better object representation. The effectiveness of the intensity-based feature is demonstrated in tracking objects in Wide Area Motion Imagery (WAMI) data. The object detection capability of the gradient-based feature extraction technique is evaluated on three different types of targets in low altitude aerial imagery. It is observed that the speed of computation is several times faster than state-of-the-art methods while maintaining comparable detection accuracies. Research work is continuing to automatically identify significant parts of an object to build the part-based model. Another direction of this research is to use the gradient-based rotation invariant features for scene matching.

Dedicated to my Grandmother, Thankamma Alex

ACKNOWLEDGMENTS

I express my deep sense of gratitude to Dr. Vijayan K. Asari for suggesting this research problem and for his guidance throughout my years as a research student. It was his keen interest in this topic, constructive suggestions and constant encouragement that helped me complete this work. He has motivated me to deal with challenges in research and to look at every hurdle as a fresh opportunity. I thank him for the great drive and consistent support he provided me during the past five years. I express my sincere thanks to the committee members Dr. Keigo Hirakawa, Dr.Raul Ordonez and Dr.Youssef Raffoul for their help, valuable comments and suggestions during different stages of research.

From the first day at the University of Dayton, the Department of Electrical and Computer Engineering treated me as family. I express my sincere gratitude to the Department of Electrical and Computer Engineering, and to Dr. Guru Subramanyam, for funding my graduate studies and extending all facilities during my time at the University of Dayton.

I extend my heartfelt gratitude to Saibabu Arigela, Varun Santhasheelan, Binu M. Nair and all the members of the Vision Lab for the cooperation and support throughout my research. My sincere thanks to the Graduate School especially to Johanna Lantz for her quick response to all my graduation related queries and concerns. My beloved wife Ann was of immense help in completing this dissertation. I express my indebtedness to my wonderful parents and sisters, and all family members for their continuous encouragement and prayers.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER I

INTRODUCTION

Artificial intelligence is increasingly being used to reduce human labour. An essential component of an intelligent system is the ability to 'see' the world around it and understand visual imagery. Building intelligent machines with the power of visual perception opens up endless possibilities such as self-driving cars, robots that can cook, automatically tagging friends on a social networking website and building efficient visual search engines. Computer vision is the sub-field of artificial intelligence that is concerned with acquiring and processing visual imagery. The ultimate goal of computer vision is developing human-like vision and scene understanding. One of the most fundamental tasks in computer vision is object detection and recognition. It has several applications including image retrieval, robotics and surveillance. The difference between object detection and recognition is in the degree of specificity. The goal of object detection is identifying an object as belonging to a particular category. Object recognition, on the other hand, is identifying a particular object within a category of objects. For example, identifying an object as a car among a category of objects such as cars and humans is object detection, while identifying a particular car is object recognition. The challenges in object detection and recognition include background clutter, view point changes, object rotation, object deformation and noise. Object detection, in general, consists of two steps - feature extraction and classification. A feature is a distinctive attribute of an object represented in some mathematical form. For example, one of the features extracted from an object that always appears in the same color in all scenes, could be the intensity histogram extracted from the object region.

1.1 Properties of a good feature

As outlined in [1], a good feature should have the following properties -

- *Repeatability* - A high percentage of features should be matchable across images of objects captured under different viewing conditions.
- *Distinctiveness* - The feature should be such that the background and the object can be easily distinguished.
- *Locality* - The features should be local, so as to reduce the effect of occlusion and to allow approximations of global photometric and geometric variations of the object under different viewing conditions.
- *Accuracy* - The detected features should be accurately localized, in image location and scale.
- *Efficiency* - Feature detection should be fast enough for use in time-critical applications.

The following two properties are required for repeatability -

- *Invariance* - Objects typically undergo some form of geometric deformation, caused either by a change in viewpoint, or by a change in relative position of parts of objects. The common approach to model these deformations is to build a mathematical deformation model and develop feature extraction methods that do not depend on deformations.
- *Robustness* - In addition to geometric deformations, object appearance can be affected by image noise, lighting changes and blur introduced by viewing distance. The parameters in the

feature extraction technique can be adjusted such that these deformations are accommodated. Doing so may reduce the overall accuracy of the system slightly. Feature robustness tackles these small deformations.

Often times, it is hard to achieve all the desired properties simultaneously. The relative importance of each property depends on the application. Increasing the invariance level of the descriptor causes the distinctiveness of the descriptor to go down. Accuracy and efficiency are clearly two other conflicting requirements; a complex algorithm may be highly accurate, but computationally expensive. Of the myriad number of image processing techniques developed over the years, most methods fall short of efficiency. In this dissertation, we attempt to formulate a method that meets the conflicting requirements of accuracy and efficiency without sacrificing other requirements for a good feature.

1.2 General object detection vs. object detection in aerial imagery

Several hundreds of methods have been proposed over the last few decades for detecting specific categories of objects such as faces, pedestrians and cars. In most object detection problems, an orientation of the object of interest is assumed. For example, pedestrians are assumed to be upright and a car is not expected to be upside down in a scene. However, this assumption cannot be made in the case of aerial images, where there is equal probability of an object appearing in any orientation. These differences are illustrated in Figure 1.1. Although humans can easily identify an object as belonging to a particular category even if it is rotated, it is a surprisingly difficult task for a computer vision algorithm.

Aerial imagery is captured either by flying platforms such as unmanned aerial vehicles (UAVs), aircrafts or by satellites. High resolution satellite imagery, such as Google Earth, is now readily available to civilians. Reconnaissance, surveillance and traffic monitoring with UAVs are becoming

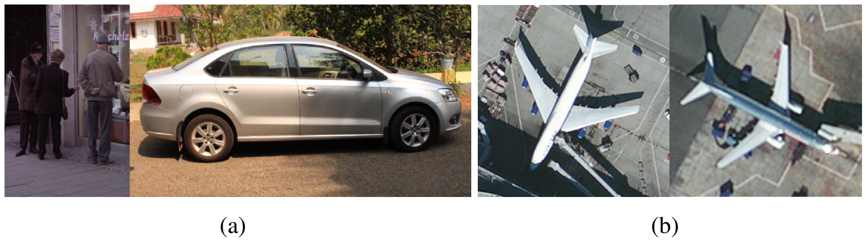


Figure 1.1: Difference between aerial imagery and images captured from a ground-based camera, (a) Natural upright orientation and (b) Aerial imagery with equal probability of any orientation.

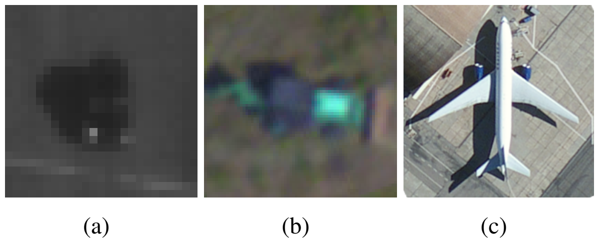


Figure 1.2: Challenges in aerial imagery, (a) Low-resolution, (b) Motion blur and (c) Self-shadow.

increasingly common. UAVs were originally developed for military applications, but recently their use has extended to civilian applications also. For example, Amazon.com recently announced plans to deploy UAVs for door delivery. Such applications involve some form of object detection. A single frame of aerial imagery typically corresponds to a large area. This poses an additional problem - large processing time. This requirement dictates that the algorithm is fast enough to achieve reasonably good detection speed. In addition to problems encountered in general object detection, object detection in aerial imagery also has to cope with motion blur and environmental aberrations such as fog and clouds. Some representative examples of objects in aerial imagery are shown in Figure 1.2. Large viewpoint changes are not usually encountered in aerial images - only slight variations of 'top-view' are encountered. A simple way to detect an object in different orientation is to rotate either the reference template or the scene and look for the object in multiple orientations.

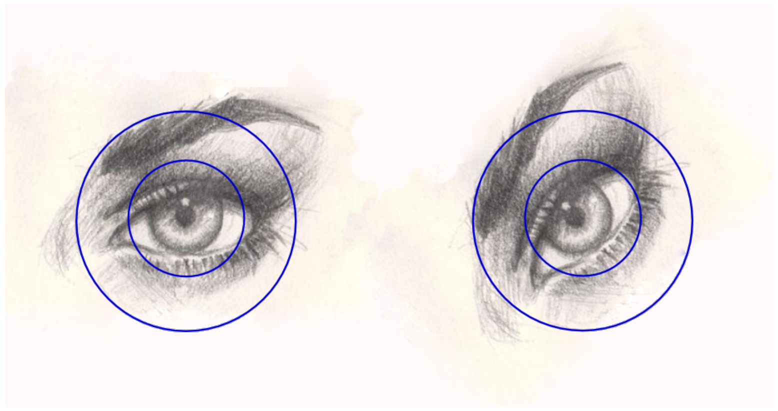


Figure 1.3: Concentric regions for feature extraction. A concatenation of rotation invariant features extracted from these regions is rotation invariant.

The fineness of rotation depends on the robustness of the feature and the classifier. However, this is a very costly process. This will also produce many more false positives as the classifier is evaluated several times at the same location. This problem can be solved by building a rotation invariant feature. Rotation invariant features can be built either from image intensity or gradients. In order to add a spatial constraint for target localization, the target is divided into circular regions as shown in Figure 1.3. A concatenation of rotation invariant features collected from circular regions is also rotation invariant. It can be seen that even if the target image is rotated, the feature extracted from a circular region remains the same. Formally we can define the rotation invariant feature $F(x, y)$ of an image patch centered at (x, y) as

$$F(x, y) = [f_1(x, y), f_2(x, y), f_3(x, y), \ldots f_r(x, y)] \tag{1.1}$$

where f_1, f_2, ... ,f_r are rotation invariant features collected from concentric regions $1, 2, \ldots, r$. Since $F(x, y)$ is a concatenation of features collected from several regions, it encodes spatial information in addition to intensity or gradient information.

A common feature of interest in computer vision is the distribution of gradients in an image. Objects are represented rather well by their silhouette or contour. The disadvantage of using contour is its sensitivity to noise. This problem can be tackled by forming a histogram of orientation of edges or gradients. Furthermore, features derived from edge information are relatively robust to illumination and color changes. Histogram of gradient orientations can be made very robust to illumination changes by normalizing the histogram [2][3][4]. The idea of using edge information for object detection dates back to as early as late 70s [5]. The importance of edges in object detection is also supported by psychological studies [6] and [7]. Intensity histograms are rotation invariant; the intensity histogram of an image and that of the rotated image are the same. However, this principle does not hold true for histogram of gradient orientations. As an object rotates, there is a cyclic shift in the histogram of gradient orientations. In Fourier domain, a cyclic shift corresponds to a change in phase. Therefore, the magnitudes of the Fourier coefficients of a gradient orientation histogram remain the same even if the object is rotated. In this work, two features, intensity-based and gradient-based, are presented. Gradient-based features work well except when resolution of the targets in the image is extremely low. For example, in high-altitude Wide Area Motion Imagery (WAMI) data, objects such as vehicles span only a few pixels.

While feature $F(x, y)$ extracted using the isotropic image division described above may be enough to describe simple objects, the method may fail in certain cases. Specifically, $F(x, y)$ is enough to describe objects that are nearly radially symmetric, like circular or rectangular objects. An example is shown in Figure 1.4(a). In Figure 1.4(b), most of the object region is contained within region 1. A large area within region 2 and region 3 are not part of the object. The feature collected from these regions are, therefore, prone to background clutter. Similar methods such as Histogram of Oriented Gradients, collect samples from dense rectangular regions to form a concatenated feature vector [4]. Such methods are not rotation invariant. Another problem is that these methods do

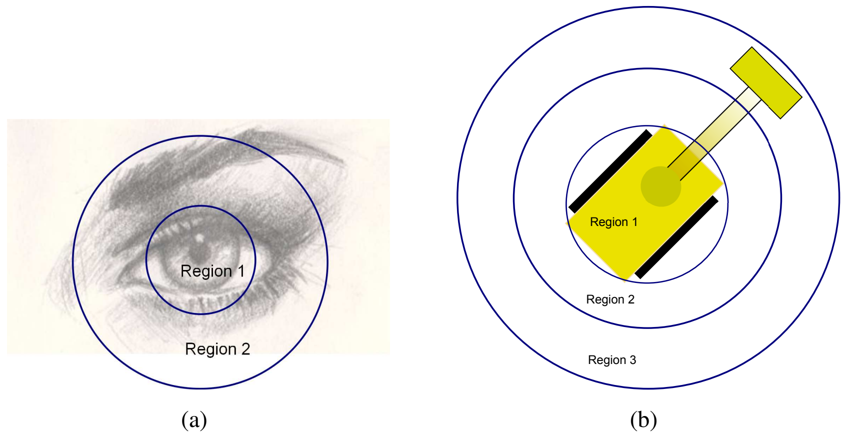


Figure 1.4: Concentric division of image region, (a) Case where concentric division performs well and (b) Case where concentric division may fail.

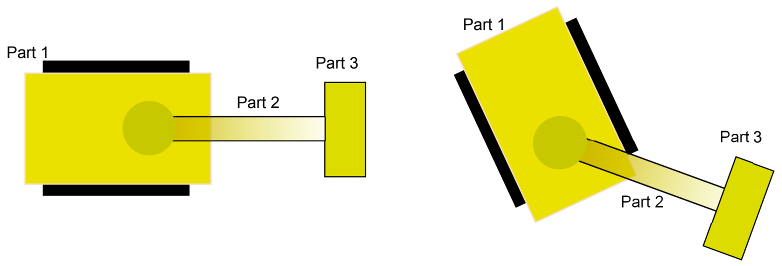


Figure 1.5: Object rotation and deformation.

not address deformation of objects and partial occlusion. In deformable or non-rigid objects, the overall geometry of the object can change, although the geometry of individual parts of the object stays the same. An example of a non-rigid object is shown in Figure 1.5, where, in addition to the overall rotation of the image, the relative positions of parts of the target can also change. When this kind of deformation occurs, most feature extraction methods fail, unless the method is part-based. To address object deformation, we apply the isotropic division on distinctive parts of an object. For example, for an aircraft, parts could be the wings and the mid-section of the fuselage. The idea of

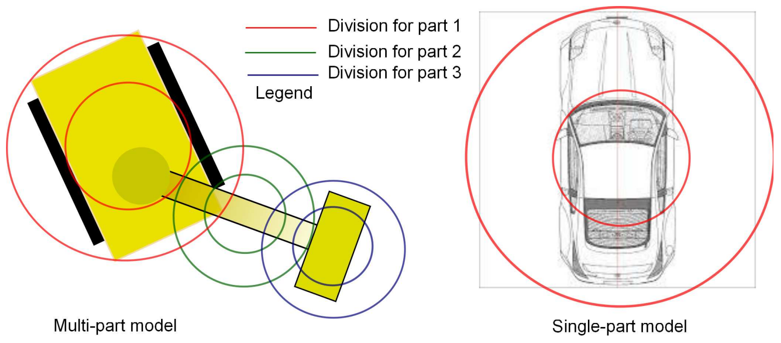


Figure 1.6: Model of an object can be multi-part or single part, depending on its complexity, rigidity and symmetry.

applying the descriptor on object parts is illustrated in Figure 1.6. If the object is rigid and approximately radially symmetric, it suffices to use a single part (the whole object) as shown in Figure 1.6.

1.3 Framework

Intensity histograms are widely used in image retrieval [8]. In this work, intensity-based features are used to track moving vehicles. The problem of target tracking is dealt with as a tracking-by-detection problem. From a list of possible target candidates, the actual target is selected as the one that has minimum distance to the reference target. Finding the right candidate in successive video frames can also be interpreted as an image retrieval problem. The basic idea presented in this work is shown in Figure 1.7. From a set of candidate features, the one that belongs to the actual target is taken as that which gives the minimum distance to the reference feature.

Gradient-based features are used for object detection. As in most object detection algorithms, the proposed method consists of a training phase, where the characteristics (features) of the target of interest are learnt by a classifier. The training phase of the algorithm is shown in Figure 1.8. The method consists of dividing each part into concentric regions and concatenating DFT magnitudes of

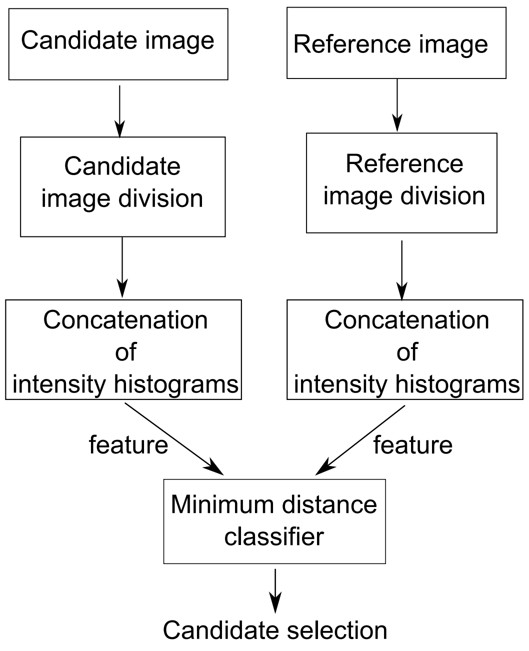


Figure 1.7: Candidate selection using intensity-based features.

gradient orientation histograms to form a rotation invariant gradient-based feature. These features are learnt by a classifier to build a model for each part. A specific rule (part configuration feature) is designed to model the relative location and sizes of parts. This rule is learnt by another classifier to build a part configuration model.

The testing phase of the algorithm, i.e., obtaining the decision when presented with a candidate image, is shown in Figure 1.9. For the candidate image to be the object (target), its part features and part configuration feature have to confirm to the learnt part feature and part configuration models respectively. If either of the features does not confirm to its corresponding model, the candidate image does not represent the target.

Integral formulation of feature extraction allows fast computation of feature vectors. There is performance advantage in moving as much computation as possible to integral computation.

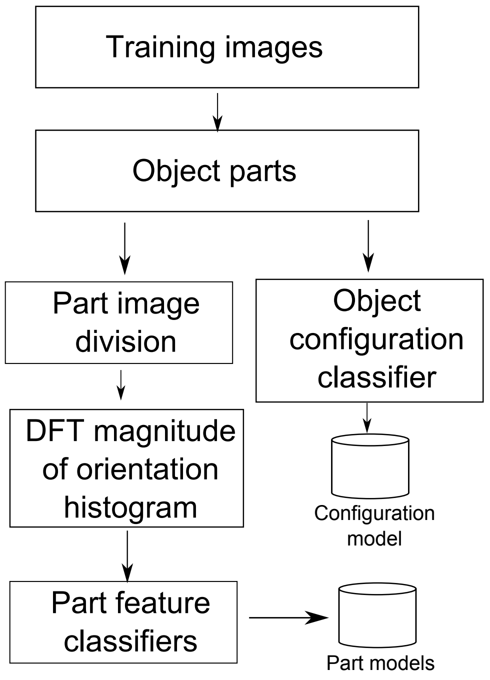


Figure 1.8: Training phase of the gradient-based approach.

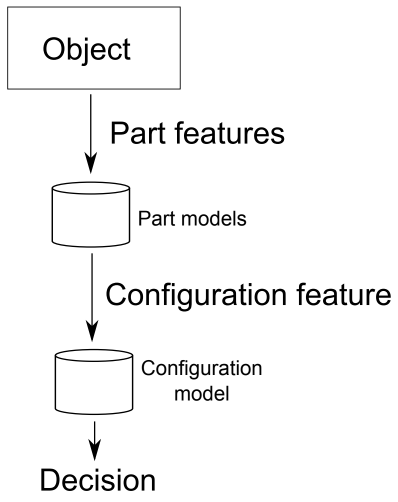


Figure 1.9: Testing phase of the gradient-based approach.

Intensity-based features are computed using integral histogram [9]. Gradient-based features are computed using a novel framework called Integral DFT.

There is a three-fold advantage in using the proposed part-based model.

- The method can handle partial occlusion. Even if a few parts are not detected because of occlusion, we can infer that the object is present based on the detected parts. For example, in Figure 1.5, if part 3 is not detected, we can be fairly confident that object is present if part 1 and part 2 are detected, provided the part-based model is built using part 1 and part 2.
- The method can handle object deformation to some extent. If all the detected parts confirm to a learnt part-geometry rule, we can infer the presence of the object.
- Since we use rotation invariant features for detecting parts, the part-based model is rotation invariant.

Once the parts of an object are determined, the parts can be assembled into the complete object based on the learnt rule modeling the relative size and position of individual parts. In this work, the object is modeled as a complete graph with nodes formed by object parts.

1.4 Specific objectives

The specific objectives of this dissertation are outlined below.

- To perform a detailed literature survey of research in object feature extraction and representation methodologies.
- To develop a feature for robust object representation by considering concentric regions enclosing the distinctive regions of the object.
- To represent the object feature by concatenating individual features of each concentric region.

- To characterize the properties of the surface of the enclosing circular regions by considering intensity distribution of the circular regions.
- To characterize the properties of the surface of the enclosing circular regions by considering the distribution of intensity gradients of the circular regions.
- To represent the gradient features by considering the normalized spectral magnitudes of the histogram of gradient orientations.
- To develop a methodology for efficient computation of the features using an integral computation framework.
- To classify the object in feature-based tracking by employing Earth Mover's Distance based nearest-neighbour search.
- To perform object detection by employing a Radial Basis Function (RBF) kernel SVM.
- To define the object as a set of distinctive parts for robust object representation.
- To represent object parts using the concatenated spectral magnitudes of gradient orientation histogram features.
- To develop a rule-based approach for combining the part features to infer the presence of the object in the scene.
- To evaluate the performance of the feature extraction methods in aerial imagery.
- To compare the performance of the proposed methods with state-of-the-art techniques.

1.5 Contributions

The main contributions of this dissertation can be summarized as follows:

- Exploiting the surface characteristics of concentric regions centered at visually salient locations of the object for feature representation.

- Concatenated intensity histograms of concentric regions representing surface characteristics of the object for defining the feature for robust object tracking.

- Concatenated spectral magnitudes of gradient orientation histograms of concentric regions representing variations in surface characteristics of the object for defining the feature for robust object detection.

- Unique way of representing surface variations employing normalized spectral coefficients of integral histograms for efficient feature extraction.

- Transformation of the feature to a uniform range to achieve illumination invariance.

- Feature representation without considering the phase component of the spectrum of histograms of concentric regions for achieving rotation invariance.

- Representation of an object as an aggregation of significant parts.

- A rule-based object detection strategy using concentric region surface features of various parts of an object for accurate object detection.

1.6 Dissertation outline

An overview of methods that are used in object detection are presented in Chapter II. Since this work is comprised of two components, feature-based tracking and feature-based detection, Chapter II includes point features used in feature tracking, and holistic approaches and part-based methods used in object detection. In Chapter III, the details of tracking by detection are presented. Gradient-based features, integral DFT framework and object modelling as parts are explained in detail in

Chapter IV. Detailed analysis of the choice of algorithm parameters, experimental backing of the components of the proposed method and qualitative and quantitative comparison of the proposed method with state-of-the art algorithms are presented in Chapter V. Finally in Chapter VI, conclusion and possible future directions of current research are presented.

CHAPTER II

LITERATURE REVIEW

In this chapter, an extensive review of research in object detection and recognition is presented. The methodologies presented are divided into three major categories - interest point detectors and point features, holistic approaches, and part-based methods. An interest point is a well-defined location on an object that can be located accurately under affine and photometric variations. Point features are local features extracted over interest points. In addition to other applications like general object recognition and stereo matching, they are used in object tracking [10]. In holistic approaches, a feature corresponding to the whole object is used, rather than features collected from interest points. Part-based approaches combine information from different parts, in the form of features, to make robust inferences about the presence and position of the whole object.

2.1 Interest point detectors and point features

In tracking applications, tracking every pixel on the object is computationally expensive. Instead of doing this, a set of interest points is first found. Corners are typically good interest points because of their well-defined positions and stability under affine and photometric variations. One of the most popular corner detectors is the Harris corner detector [11]. The 'cornerness' of a point is determined by analyzing the Eigenvalues of the structure tensor (second moment matrix) obtained from image gradients. For corners, both Eigenvalues have large values. To avoid explicit computation of Eigenvalues λ_1 and λ_2 of the structure tensor A, the quantity $det(A) - \kappa trace(A)^2$ is used. The value of the parameter κ is determined experimentally. Shi-Tomasi corner detector uses the same structure tensor, but the quantity $min(\lambda_1, \lambda_2)$ is used to determine whether a point is a corner or not [12]. A number of detectors have been proposed that do not rely on image gradient. Smallest Univalue Segment Assimilating Nucleus (SUSAN) corner detector uses the idea of Univalue Segment Assimilating Nucleus (USAN) [13]. A circular template of radius 3.4 is placed on a pixel location termed the nucleus. The number of pixels within a threshold of the intensity of nucleus, or the USAN area, depends on where the nucleus is located. The key idea is that USAN area is minimum when the nucleus is at a corner location. Thus, corners give Smallest USAN (SUSAN). The advantages of SUSAN over methods that rely on image gradient is that it is less sensitive to noise. Using a circle of radius 3.4 amounts to testing 37 pixel locations. Features from Accelerated Segment Test (FAST) is similar to SUSAN, but differs in two aspects - first, instead of testing all pixels in the circular mask, only the pixels along the boundary of the circle are tested and second, a machine learning approach is used to classify a point as a potential corner [14]. The nucleus is compared with 16 pixels along the boundary of the circle. The central pixel is considered a corner if at least S connected pixels are brighter or darker within a threshold of the central pixel. The value of S determines the angle of the corner. The original choice of S in the algorithm was 12 because it allowed for a high speed Accelerated Segment Test (AST). First, pixels 1 and 9 are examined; if these pixels are not within the threshold, the central pixel is not a corner. If the central pixel can be corner, pixels 5 and 13 are examined. For the central pixel to be a corner at least three of pixels 1, 5, 9 or 13 should be brighter or darker by the threshold. If this condition also proves to be true, all the remaining pixels are examined. As can be seen, the high speed test quickly rejects points that are not corners. It is obvious that for S $<$ 12, a large number of tests are required. To address this issue, a machine learning approach is used. Each of the 16 pixels on the circle can fall into one of the three categories - brighter, similar or darker. The ID3 decision tree learning algorithm is applied on all the 16 pixels to determine best order in which the pixels can be tested [15]. In most applications $S = 9$ is used. In an improved version of FAST, called FAST-er, instead of a circle of thickness 1 pixel, a thicker circle of 3 pixels is used. The disadvantage of FAST is its dependence on learning based on application domain. In recently introduced AST based detector called Adaptive and Generic Accelerated Segment Test (AGAST), learning is not based on a particular problem domain [16].

Although computationally expensive, Scale-Invariant Feature Transform (SIFT) is one of the most popular feature point detectors [2][3]. In the context of object detection, an object can be identified based on the number of robust feature point matches between the reference object and the target object. In SIFT algorithm, after a set of interest points called keypoints are found, a feature histogram is computed around it. Interests points are computed as follows. First the image is convolved with a Difference of Gaussian (DoG) filter, a filter constructed by finding the difference of Gaussian filters at two different scales. The DoG filter derived from Gaussians at successive scale σ and $k\sigma$ as

$$DoG(x, y, \sigma) = \Gamma(x, y, k\sigma) - \Gamma(x, y, \sigma) \tag{2.1}$$

where $\Gamma(x, y, \sigma) = \frac{1}{2\pi\sigma^2}e^{-\frac{x^2+y^2}{2\sigma^2}}$.

The convolution with DoG filter can be computed efficiently as the difference between the convolution of image with Gaussians at successive scales. After the convolutions for each scale is computed, local extrema in scale space are selected as keypoints. A local extrema is a point which is either smaller than or larger than its neighbors in space, and two adjacent scales above and below it, as shown in Figure 2.1. The interpolated location of the keypoint is found by fitting a quadratic function to the candidate keypoints. The true extrema are found by equating the derivative of the function to zero [17]. Some keypoints may represent edges. Keypoints along edges have a low

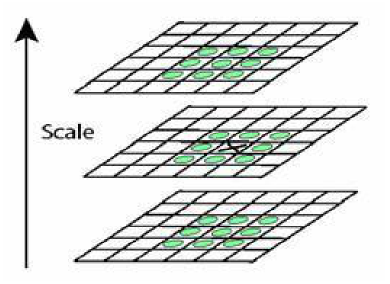


Figure 2.1: Scale space extrema in SIFT are those that are larger or smaller than all neigbhors in space and scale.

value for the ratio of principal curvature in direction perpendicular to the edge, to that in the direction along the edge. Keypoints with values of this ratio smaller than a threshold are eliminated. To achieve rotation invariance, the image patch around the keypoint is rotated such that the dominant orientation points upwards. SIFT feature is a 128 element vector built by computing 8 bin histograms of gradient orientations in a 4 x 4 patch region around the keypoints. Trilinear interpolation is used to vote into this spatial-orientation histogram. The histograms are normalized to achieve illumination invariance. Finally, a Gaussian is applied over the region to give higher weights to locations closest to the center. A more compact and accurate version of SIFT, called PCA-SIFT, is presented in [18]. Principal Component Analysis (PCA) is a popular technique for dimensionality reduction. In PCA-SIFT, instead of applying Gaussian weight on the image patch, PCA is applied to the spatial-orientation histogram around the keypoint to produce a more compact descriptor. Despite PCA-SIFT feature vector being much more compact than SIFT, it has been proven to be more accurate. In Fast Approximated SIFT, a speed-up of 8 times is achieved by approximating DoG as Difference of Mean (DoM) and using integral formulation [19].

Color information is discarded when computing SIFT descriptor. A number of methods have been developed to incorporate color information into SIFT [20]. Bosch et al. introduced HSV-SIFT [21], where the feature vector is a concatenation of SIFT feature computed in H, S and V spaces. In HueSIFT, SIFT feature vector is concatenated with hue histogram. Other schemes include computing the SIFT feature vector in opponent color space or RGB color space.

Yet another popular extension of SIFT is Gradient Location-Orientation Histogram (GLOH), designed to have higher robustness and distinctiveness [22]. This is achieved by computing the SIFT descriptor in a log-polar location grid, with three bins in radial direction and eight in angular direction. The central bin is not divided into angle bins. A 16 bin quantization is used for gradient histogram. This results in a 272 element descriptor. Finally, PCA is applied to reduce the size of the descriptor. The covariance matrix for PCA is computed based on 47,000 image patches collected from random examples. In GLOH the feature patch is rotated in the direction of dominant gradient, as in SIFT, to achieve rotation invariance. Finding dominant orientation can be ambiguous when there are multiple peaks with similar values in the orientation histogram. To avoid rotating the patch, all shifted versions of the descriptor are used to find the best match in [23]. In [24] binary hashing is used to convert descriptors obtained by rectangular grid binning as in SIFT and SURF or log-polar binning as in GLOH to short binary codes called Compact and Real-time Descriptors (CARD). Descriptors that are binary strings can easily be compared using Hamming distance.

Spin images achieve rotation invariance of image patches [25]. The method was developed by Johnson and Hebert [26] for 3D surface registration and object registration. The intensity domain spin image is a two-dimensional histogram of distance from the center and intensity values. To give more importance to the pixels closest to the bin centers, pixel intensities contributions are weighted according to the distance from the center. Lazebnik et al. have proposed a rotation invariant feature called Rotation Invariant Feature Transform (RIFT) [25]. The patch is a normalized circular region divided into concentric rings of equal spacing, and the histogram of gradient orientation is computed in each ring in the radial direction.

Rotation Invariant Fast Feature (RIFF) descriptor is similar to RIFT, but uses a binning technique that reduces the dimensionality of the histogram of gradient orientations [27]. RIFF is based on Radial Gradient Transform (RGT), in which a coordinate system consisting of radial and tangential directions is used. An approximate RGT, obtained by using only 8 directions is used to improve speed. FAST is used as the interest point around which the descriptor is computed. RIFF-Polar is a newer variant of RIFF [28]. In RIFF-Polar, to improve the expressiveness of annular binning in RIFF, an angular and radial binning pattern is used as in GLOH. It is obvious that this kind of binning removes the rotation invariance property of RIFF. To retain rotation invariance property of the descriptor, all the cyclic shifts of the descriptor are considered. The distance $D(p, q)$ between two descriptors p and q can then be expressed as

$$D(p, q) = \min_{\theta}\{D(p, q_\theta)\} \tag{2.2}$$

where q_θ represents rotations of feature q by discrete angles.

Speeded-Up Robust Features (SURF) is similar to SIFT but is several times faster [29]. To compute keypoints, SURF relies on an approximation of Hessian matrix rather than on DoG. The Hessian matrix $H(x, y, \sigma)$ at scale σ is

$$\mathcal{H}(x, y, \sigma) = \begin{bmatrix} L_{xx}(x, y, \sigma) & L_{xy}(x, y, \sigma) \\ L_{xy}(x, y, \sigma) & L_{yy}(x, y, \sigma) \end{bmatrix} \tag{2.3}$$

where $L_{xx}(x, y, \sigma)$, $L_{xy}(x, y, \sigma)$ and $L_{yy}(x, y, \sigma)$ are the convolution of the image with the second derivatives of Gaussian $G(x, y, \sigma)$. The computational efficiency of SURF arises from approximating the second derivatives of Gaussian as 9×9 box filters as shown in Figure 2.2. The convolution of these filters with the image can be computed very fast using integral images. With this approximation, the approximate determinant of Hessian is computed as

$$det(\mathcal{H}_{appox}) = D_{xx}D_{yy} - (0.9D_{xy})^2 \tag{2.4}$$

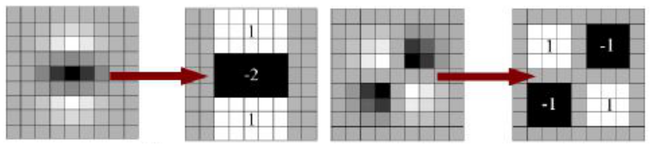


Figure 2.2: Approximating Gaussian derivatives as 9×9 box filters.

where D_{xx}, D_{yy} and D_{xy} are box filter approximations. Keypoints are those at which the determinant of Hessian becomes maximal. A dominant orientation is assigned to each keypoint. To find the dominant orientation, the response to Haar wavelets in x and y direction are first computed. In a sliding angular window of angle $60°$, a vector is computed by summing the responses in x and y directions. The direction of the longest of such vectors is taken as the dominant orientation. To compute the SURF feature vector, a 4×4 patch around the region is considered. In each region, the vector $[\sum d_x, \sum d_y, \sum |d_x|, \sum |d_y|]$ is computed, where d_x and d_y are the responses of the Haar wavelets in x and y directions. The final feature vector is formed by concatenating these vectors collected from each subregion, resulting in a feature length of $4 \times 4 \times 4 = 64$.

Keypoints in Central Surround Extrema Feature (CenSurE) are based on Bilevel Laplacian of Gaussian (BLoG) approximation of DoG [30] [31]. The BLoG filter, shaped like an annular region, has only two levels, 1 and -1. Despite its simplicity, the response to this kind of filter is hard to compute because of its geometry. To circumvent this difficulty, CenSure approximates BLoG as octagonal, hexagonal or square ring-shaped regions. Square ring shaped region is the coarsest approximation of annular region. The best trade-off between accuracy and speed is achieved for octagonal region. Responses to octagonal ring shaped filters are computed as the difference of responses to two concentric octagonal filters. The response to each octagonal region is computed by dividing the region into two trapezoids and a rectangular region. The sum of intensities in trapezoidal region is computed using *slated integral histograms*. The descriptor centered around the keypoint is built in a manner similar to SURF. DAISY is a descriptor inspired by SIFT and GLOH [32]. Instead of using the square binning pattern as in SIFT and SURF, DAISY uses a circular grid. In this aspect, it is closer to GLOH than SIFT. Instead of computing the histogram in a circular bin, DAISY computes a smoothed orientation map ${G_o}^{\Sigma} = G_{\Sigma} * \left(\frac{\partial I}{\partial o}\right)^+$ where G_{Σ} is a Gaussian kernel of standard deviation Σ, $\left(\frac{\partial I}{\partial o}\right)^+$ is the gradient operator in direction o and $(.)^+$ is the operator such that $(a)^+ = max(a, 0)$. The quantity $h_{\Sigma}(u, v) = \left[{G_1}^{\Sigma}(u, v), \ldots, {G_H}^{\Sigma}(u, v)\right]$ is the equivalent of histogram in SIFT, with H representing the number of bins. The final descriptor is a concatenation of normalized $h_{\Sigma}(u, v)$ collected from each circular region. Circular binning allows for some robustness to rotation, but not full invariance to rotation. Binary Robust Independent Elementary Features (BRIEF) is a binary descriptor similar to CARD [33]. While CARD builds a large descriptor first, and then reduces the size of the descriptor, BRIEF computes a binary descriptor directly. The basis of BRIEF computation is a test τ defined on a patch p of size $S \times S$

$$\tau(p; \boldsymbol{x}, \boldsymbol{y}) = \begin{cases} 1, & \text{if} p(\boldsymbol{x}) < p(\boldsymbol{y}) \\ 0, & \text{otherwise} \end{cases} \tag{2.5}$$

where $p(\boldsymbol{x})$ and $p(\boldsymbol{y})$ are the pixel intensities at point $\boldsymbol{x}$ and $\boldsymbol{y}$ in the smoothed version of patch p. Choosing n_d unique location pairs, the binary feature is

$$f_{n_d}(p) = \sum_{1 \leq i \leq n_d} 2^{i-1} \tau(p; \boldsymbol{x_i}, \boldsymbol{y_i}) \tag{2.6}$$

Hamming distance can be used to compare BRIEF descriptors. It may be noted that BRIEF is not a rotation invariant descriptor. Sensitivity of BRIEF to rotations is addressed in Oriented FAST and Rotated BRIEF (ORB) [34].

2.2 Holistic approaches

Holistic methods work best for detecting objects that are nearly rigid. Viola-Jones object detection framework is one of the seminal works in real-time object detection. It uses a set of Haar

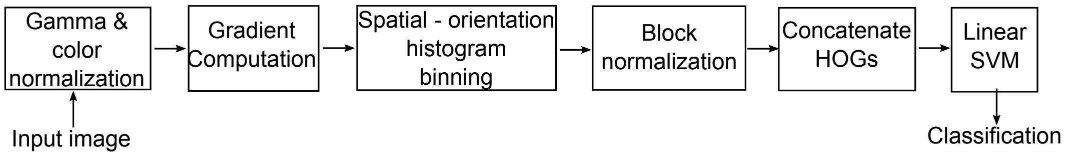


Figure 2.3: HOG computation flow.

features and an AdaBoost cascade to achieve real-time performance. In the recent work by Leitloff et al., Haar-like features in a boosting framework are used to detect vehicles in satellite imagery [35]. Based on the observation that humans identify an object by analyzing their general shape rather than pixel intensities, Belongie et al. proposed 'Shape Contexts' [36]. Shape context is based on the distribution of edge pixels. These edge pixels are not landmark points such as local extrema. For each point on a shape with n points, the histogram of relative positions of the remaining $n - 1$ points, is computed. The histograms are computed on a log-polar grid. A method to deal with illumination changes, called GradientFaces, is presented in [37]. In their method, the image I is first smoothened using a Gaussian kernel. The smoothened image is differentiated in x and y directions by convolving with the differential of Gaussian in x and y directions, to get I_x and I_y respectively. GradientFaces is defined as $tan^{-1}(\frac{I_y}{I_x})$. Many state-of-the-art holistic approaches are based on Histogram of Oriented Gradients (HOG) [4], Local Binary Patterns (LBP) [38] or combination of both [39][40]. The HOG descriptor was originally developed for pedestrian detection, although it can be applied in most object detection problems. HOG can be thought of as a denser version of SIFT. It is essentially a feature vector built from histograms of orientations of gradients collected from overlapping spatial regions in the region of interest. It consists of the following steps - gamma and color normalization, gradient computation, spatial and orientation histogram binning and contrast normalization. The system architecture for general object detection using HOG is shown in Figure 2.3. Gamma correction is a technique to improve contrast in an image. The basic form of gamma correction is given by $I_{out} = \alpha I_{in}^{\gamma}$ where I_{in} is the input image and I_{out}, the output image. The parameter α is a scaling factor and γ is the value of gamma applied. In color images such as RGB images, gamma correction is applied to each channel.

Before computing the gradients, a Gaussian smoothing can be performed to remove high frequency noise. There are several schemes for computing gradients. The gradient of an image (G) is composed of horizontal gradient (G_x) and vertical gradient (G_y). They are computed by convolving the image I with a kernel as shown in (2.7) and (2.8).

$$G_x = K_x * I \tag{2.7}$$

$$G_y = K_y * I \tag{2.8}$$

In (2.7) and (2.8), K_x and K_y are the horizontal and vertical kernels respectively. The kernels used are usually [1 -1], [-1 0 1], [1 -8 0 8 -1], 3 $\times$ 3 Sobel masks and 2 $\times$ 2 diagonal masks. From the horizontal gradient G_x and vertical gradient G_y, the gradient magnitude G and gradient orientation θ_G are computed as

$$G = \sqrt{G_x^2 + G_y^2} \tag{2.9}$$

$$\theta_G = tan^{-1}\left(\frac{G_y}{G_x}\right) \tag{2.10}$$

Fundamental to the computation of HOG are the terms *cells* and *blocks*. A cell is a local spatial region, while a block is a group of cells. The most commonly used configuration is called R-HOG, in which cells are square-shaped. The number of pixels in a cell (η) and the number of cells in a block (ς) are parameters in HOG computation. The choice of η and ς depends on the application. As a rule of thumb, the number of pixels in a cell should correspond to the size of parts of the object. For example, this can be the size of limbs in case of pedestrian detection and the size of windshield for vehicle detection. The idea of cells and blocks is shown in Figure 2.4. The pixels near the edges of a block are down-weighted by applying a Gaussian with standard deviation σ on it. A β bin

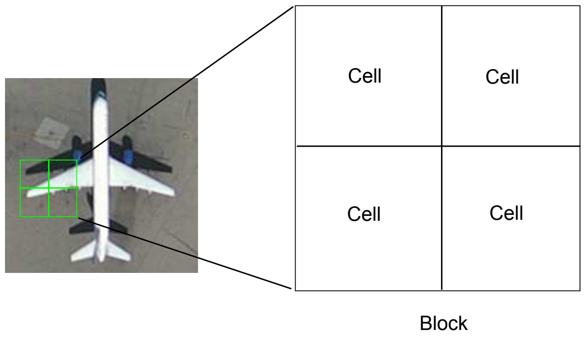


Figure 2.4: Cells and blocks in HOG.

histogram is built corresponding to each of the $\varsigma \times \varsigma$ spatial bins. Hence, a histogram in a block is a $\varsigma \times \varsigma \times \beta$ spatial-orientation histogram. Orientations can be evenly spaced over $0-180°$ or $0-360°$. When the bin orientations are spaced over $0 - 180°$, the sign of the gradient orientation is ignored. In general, accuracy increases with increasing the number of angle bins β upto 9. For pedestrian detection, the best results are given when unsigned gradients are used. However, there is significant improvement when signed gradients are used for vehicle or motorbike detection. The gradient value at a particular spatial location can make contributions to any of the four neighboring spatial bins and any of the two neighboring angle bins as illustrated in Figure 2.5. The contributions made to these bins can be calculated using trilinear interpolation. Let b_x, b_y and b_θ be the bandwidths along x, y and θ. Bandwidth of a histogram is the distance between neighboring bins. The gradient magnitude G at location (x, y, θ) is distributed to the neighboring bins in histogram H based on distances to the bin centers. An example of histogram update is

$$H(x_1, y_1, \theta_1) \leftarrow H(x_1, y_1, \theta_1) + \left(1 - \frac{x - x_1}{b_x}\right)\left(1 - \frac{y - y_1}{b_y}\right)\left(1 - \frac{\theta - \theta_1}{b_\theta}\right) G \qquad (2.11)$$

where (x_1, y_1, θ_1) is a spatial-orientation bin. The bin update expressions for other bins can be formulated in a similar manner. Beside the R-HOG configuration, there can be several other geometries for the HOG. Blocks can be vertical (e.g., 2×1 cells) or horizontal (e.g., 1×2 cells).

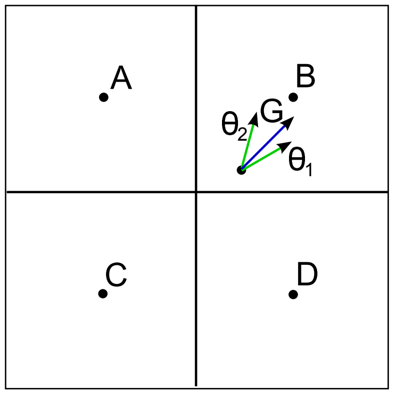


Figure 2.5: Building the spatial-orientation histogram in HOG. A point votes into four spatial bins A, B, C and D, and two angle bins θ_1 or θ_2.

Dalal and Triggs have also proposed the C-HOG, which has a 'circular' configuration. This configuration is similar to GLOH. Another variant of C-HOG is one in which the region is divided into angular sectors. To reduce the effect of variations in illumination, the histogram in each block is normalized. It can be done in several different ways. Commonly used schemes are shown below.

$$f = \frac{v}{\sqrt{\|v\|_2^2 + \epsilon}} \tag{2.12}$$

$$f = \frac{v}{\|v\|_1 + \epsilon} \tag{2.13}$$

$$f = \sqrt{\frac{v}{\|v\|_1 + \epsilon}} \tag{2.14}$$

In (2.12), (2.13) and (2.14), f is the normalized vector, v the vector to be normalized, and $\|v\|_k$, the L_k norm of v. ϵ is a small value used to avoid divide-by-zero error. Clipped L_2 norm is another normalization scheme related to L_2 normalization. It is calculated by finding the L_2 norm, clipping the maximum value to 0.2 and then renormalizing the clipped histogram. The normalized histograms collected from all the blocks in the detection window are concatenated to form the final descriptor. Blocks typically overlap. This gives a certain degree of translation invariance to the descriptor. Dalal and Triggs have suggested using the highest gradient among all color channels when color information is available. However in [41], it is demonstrated that this is not the best strategy to incorporate color information. They have shown that HOG feature vector formed by concatenating HOG computed in Y, Cb and Cr components of YCbCr space performs significantly better than HOG computed with highest gradients. The method was tested in traffic sign recognition. Although this method may work reasonably well for detecting traffic symbols in a particular country, it may not be generalizable enough to detect any type of traffic symbol. Binary tests similar to those in FAST and BRIEF are done in HOG space to train a Random Fern classifier for pose estimation in [42]. Once the pose is estimated, the classifier corresponding to the pose can be used to detect a potential target. Another popular feature is the Local Binary Pattern (LBP). The original LBP operates on 8 neighbors of a pixel by thresholding them with the value of the pixel. If a pixel intensity is greater than or equal to the intensity of the central pixel, it is assigned a value '1'. A value of '0' is assigned otherwise. The assigned values are read in a counter-clockwise fashion starting with the pixel on the left of the central pixel. The LBP value associated with the pixel is the decimal value corresponding to the binary pattern. A histogram of LBP is constructed by binning the number of possible types of LBPs. Formally, LBP operator can be defined as

$$LBP(x,y) = \sum_{0}^{7} s(i_n - i_{x,y})2^n \tag{2.15}$$

where i_n are the intensities of the neighbors of (x,y). $i_{x,y}$ is the intensity of the pixel at (x,y). The function $s(.)$ is defined as $s(x) = \begin{cases} 1, & \text{if } x \geq 0 \\ 0, & \text{otherwise} \end{cases}$.

LBP is more sensitive to illumination variations than HOG. In [43], color information is integrated into LBP to address this issue. With this approach, LBP is successfully used to recognize objects in Pascal Visual Object Challenge (VOC) 2007 images. As an image rotates, its LBP histogram undergoes a cyclic shift. A method that combines this idea with Fourier transform to achieve rotation invariance is presented in [44]. A similar approach is used to achieve rotation invariance in Equivariant Histogram of Oriented Features (EHOF) descriptor [45]. In this method, the 2D DFT of a spatial-orientation histogram matrix is taken to achieve rotation invariance. Computation of 2D DFT and soft binning with trilinear interpolation causes the descriptor computation to be slow. Other methods that use DFT include [46] and [47]. The circular geometry of the grid pattern in these methods makes admissibility into integral formulation hard. Using a single rotation sensitive feature like HOG and multiple classifiers corresponding to multiple orientations for aerial object detection is prohibitively expensive. A common approach in this domain is to combine several features and high level information such as context. For example, in [48], Maximally Stable Extremal Regions (MSER) [49] are used to extract salient regions. Once these regions are extracted, a symmetry detection technique using Directed Chamfer Matching (DCM) [50] is used to detect airplanes.

2.3 Part-based methods

Part-based models are motivated by the fact that most objects are formed by a set of parts. For example, a vehicle can be described by the appearance and relative position of its doors, hood, wind shields, windows and tires. The concept of part-based model dates back to early 70s in the highly influential work by Fischler and Elschlager [51]. They used a pictorial structure to represent an object as a collection of object parts. The appearance and spatial integrity of an object was modeled as parts connected by a set of springs as shown in Figure 2.6.

Some approaches model the parts, but not the relationship between parts. The most popular among this class of methods is the bag-of-visual-words (BoV) model. The commonly used steps in BoV are keypoint extraction, extracting patches centered around keypoint and clustering to form a visual vocabulary [52]. In [53], BoV built with SIFT points clustered using *k*-means clustering is employed to classify objects such as buildings, crop fields and water bodies in aerial imagery. In contrast to BoV model, constellation model accounts for the spatial relationship between every part of the object. Constellation model is a generative probabilistic model. Part locations, part

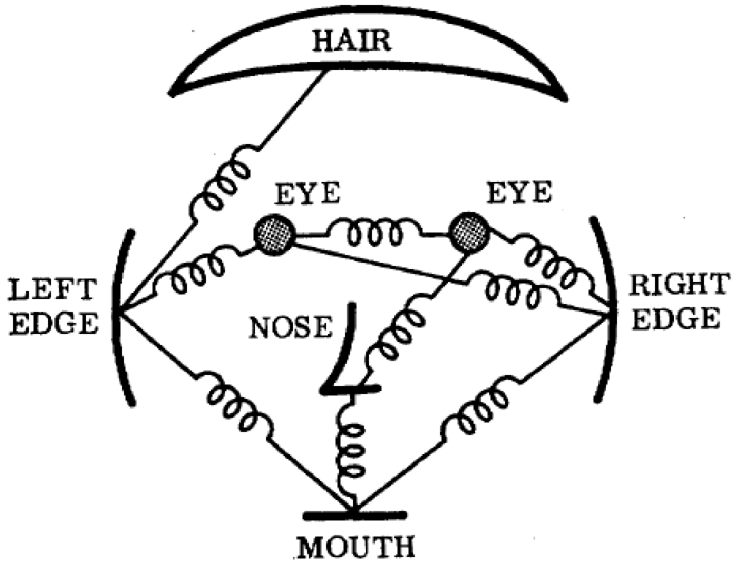


Figure 2.6: Part-based model introduced by Fischler and Elschlager [51].

appearances or both are modeled as joint probability distribution functions. The first constellation model was introduced by Burl et al. [54]. In their method, the parts are manually labelled to build the probability distribution model describing part relationships. A correlation-based method is used as part detectors. The tediousness of hand-labeling training data can be solved by using an unsupervised learning mechanism. In the constellation model introduced by Weber et al., unsupervised learning is achieved by considering patches extracted around interest points as object parts [55]. Interest points are extracted using Förstner corner detection [56]. Image patch features used in the approach are DoG filtered versions of the patches. The large set of keypoints is clustered together using a clustering method to reduce the number of parts. Fei-Fei et al. further improved the Weber's method by using Bayesian estimation and Maximum Likelihood learning [57].

Recently, Felzenswalb et al. described a star-shaped model in which a root is connected to parts [58]. The model is learnt using a latent SVM. A rotation invariant part based model is presented in [59]. SIFT features computed along edges is used to build a codebook of features. Rotation invariance is achieved by using polar coordinates to specify the position of the features.

A part-based framework to detect aircrafts in aerial imagery is presented in [60]. The basic idea in the algorithm is finding the SIFT feature vector of an object part. The histogram of gradient orientation is computed at a point for a square-shaped image region centered at a manually selected point. The highest peak in the histogram corresponds to the dominant orientation. The image patch is rotated such that the dominant orientation points in a canonical direction (e.g., upwards). Histogram of Oriented Gradients is then computed on the rotated patch. This feature is termed Rotation Invariant Histogram of Oriented Gradients (RIHOG). A rotation invariant part based model is realized by representing the object as an $(n+1)$-tuple $(P_0, P_1, \ldots, P_n)$ with P_0 representing the whole object and P_i representing the i^{th} part. Each entry in the representation $(P_0, P_1, \ldots, P_n)$ is a tuple defined as

$$P_k = (\varphi(p_k), p_i, \theta_k) \tag{2.16}$$

where $\varphi(p_k)$ and θ_k are the RIHOG feature and dominant orientation respectively of the k^{th} part centered at position p_k.

The position p_k is

$$p_k = \begin{cases} (x_0, y_0), & \text{if } k = 0 \\ (r_k, \alpha_k), & \text{otherwise} \end{cases} \tag{2.17}$$

where (x_0, y_0) is the location of the whole object. The locations of parts (r_k, α_k) are the polar coordinates of the parts with respect to the center of the whole object. The score of a detection window is calculated as

$$score(p_0, p_1, \ldots, p_n) = \sum_{i=0}^{n} f_i.\varphi(p_i) - \sum_{i=2}^{n} d_i.\varphi_d(dr_i, d\alpha_i) - \sum_{i=1}^{n} e_i.\varphi_e(d\theta_i) \tag{2.18}$$

In (2.18), f_i represents the weights for the RIHOG features of the whole object and object parts, d_i the weights of deformation cost $\varphi_d(dr_i, d\alpha_i)$ and e_i, the weights of rotation cost $\varphi_e(d\theta_i)$. This formulation is very similar to that used in Felzelswalb's discriminatively trained parts based model described earlier. The deformation of the model relative to the whole object is normalized to the first part (landmark part) as in (2.19).

$$(dr_i, d\alpha_i) = (\frac{r_i}{r_1}, \alpha_i - \alpha_1) \tag{2.19}$$

r_i and α_i are the distance to the i^{th} part and the angle that the i^{th} part makes to a fixed reference. $\alpha_i - \alpha_1$ is therefore the angle between the line joining the i^{th} part to the center and the first part to the center.

The deformation term is

$$\varphi_d(dr_i, d\alpha_i) = (dr_i, d\alpha_i, dr_i^2, d\alpha_i^2) \tag{2.20}$$

Normalizing the dominant orientation of the i^{th} part with respect to the whole object as

$$d\theta_i = \theta_i - \theta_0 \; where \; i = 1, 2, \ldots, n \tag{2.21}$$

the rotation cost of the i^{th} part is

$$\varphi_e(d\theta_i) = (d\theta_i, d\theta_i^2) \tag{2.22}$$

The score in (2.18) can be expressed as a dot product between model parameters and model feature.

$$score(\gamma) = \beta.\Phi(\gamma) \tag{2.23}$$

$$\beta = (f_0, f_1, \ldots, f_n, d_2, \ldots, d_n, e_1, \ldots, e_n) \tag{2.24}$$

$$\Phi(\gamma) = (\varphi(p_0), \ldots, \varphi(p_n), -\varphi_d(dr_2, d\alpha_2), \ldots, -\varphi_d(dr_n, d\alpha_n), -\varphi_e(d\theta_1), \ldots, -\varphi_e(d\theta_n)) \tag{2.25}$$

$$\gamma = (p_0, p_1, \ldots, p_n) \tag{2.26}$$

(2.23) establishes the connection between this model and linear classifiers like SVMs. To detect an object of interest in a scene, a sliding window detector is used to search for the whole object using feature $\varphi(p_0)$. If the whole object is detected, a second sliding window is used in the detection window to search for all parts. If all parts are detected, a linear SVM is used to test the feature $\Phi(\gamma)$.

2.4 Summary

In this chapter, several features used in object detection and tracking were discussed. Those methods with which the proposed method are compared, were discussed in detail. In the domain of tracking in low resolution aerial imagery, many point detectors fail as they are based on some form of gradient or intensity comparisons. Those methods that are highly accurate are computationally expensive. Several methods that rely on annular division of a local neighborhood were also presented. These methods do not allow admissibility into a fast computation framework such as integral histogram. In the domain of object detection in aerial imagery, it is imperative that the method is highly computationally efficient because of the size of imagery. The traditional way is to use methods that work on ground-shot imagery, but with classifiers trained for different orientations. Obviously this is also computationally expensive. The method presented in this dissertation is reminiscent of some of the methods discussed in this chapter, but is significantly better in terms of computational efficiency.

CHAPTER III

INTENSITY-BASED ROTATION INVARIANT FEATURES FOR OBJECT TRACKING

Intensity-based rotation invariant features are used in applications like tracking by detection. In this chapter, we show how rotation invariant intensity features can be used for tracking in Columbus Large Image Format (CLIF) Wide Area Motion Imagery (WAMI) [61]. The resolution of the imagery is extremely low, with targets spanning only 10 to 20 pixels. At this resolution, gradient-based features do not work well, as will be shown in the results section. In tracking applications, where targets are tracked by detecting the reference target taken from the first frame, the system cannot be trained for two reasons. First, training process is expensive; it involves manual selection of training examples and a time-consuming decision boundary computation process. Second, only a single positive example is available. In such applications the target is found using a distance measure.

In general, any available cue is of interest in object detection. There are several cues that are of interest in tracking.

They include :-

Physical characteristics: Features such as color, texture, physical structure, location, orientation and depth map [62] [63] [64] are not available in low resolution high-altitude imagery.

Behavioral patterns: This includes behavioral patterns such as the gait of a person, the trajectory, acceleration and speed of a vehicle etc. This can be used for targets with predictable movements like vehicles.

Environment context: These are the features of physical environment that co-occur with a target feature. For example, a car is likely to be on a road or a parking spot rather than on the top of a building. Such contexts are very useful for effective search and detection of objects. However, in WAMI data, areas such as road cannot be reliably extracted since there is very little texture information available.

Points: An object can be represented as its centroid or as a set of points on it. This is suitable for objects that span a small region of the image [65][66]. However, point features and interest points such as SIFT, SURF and Harris corners are not effective in identifying points in the objects in high-altitude low resolution data.

Tracking aims at generating the trajectory of a moving object over time, by locating its position in consecutive frames. The following steps need to be done to do this − image registration, search space identification and sliding window detection.

3.1 Image registration

Image registration reduces or eliminates global camera motion by transforming the data into a particular coordinate system. Descriptors such as SURF and SIFT can be used for image registration. The computation of SIFT descriptor is relatively expensive. SURF, on the other hand, is several times faster than SIFT. Interest points are located in the reference frame and the frame to be registered. The frame to be registered is transformed in such a way that the corresponding interest points are matched in spatial location. We use SURF keypoints for image registration. Figure 3.1 shows two frames registered with respect to one another.

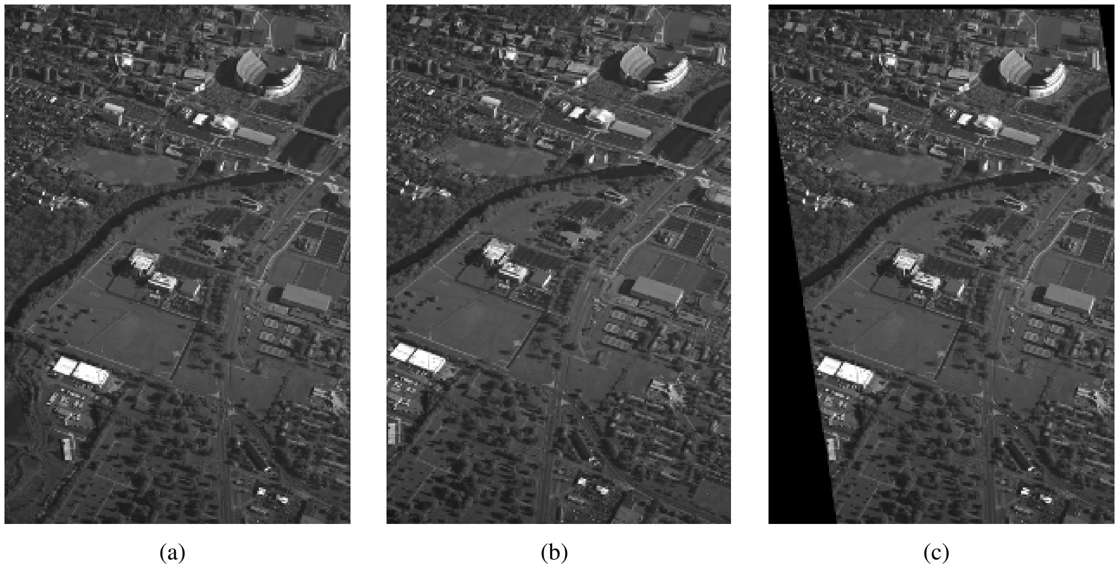


Figure 3.1: Example of image registration in CLIF WAMI data [61], (a) Reference image, (b) Frame to be registered and (c) Registered frame.

3.2 Search space identification

The search region is a rectangle that extends three times the target size to the left and the right, and two times the displacement in the forward direction and two times the target size in the backward direction. The position of the target, along with its velocity, is predicted in each frame based on the previous observation of position and velocity. The current velocity of the target is proportional to the difference between the current position and that in the previous frame. Reducing the search space to a smaller area, instead of the entire frame, allows us to eliminate false positives significantly and reduce computation time.

3.3 Sliding window detection

Sliding windows are commonly used in object detection algorithms to search for an object of interest in an image. The approach involves scanning the image with a fixed-size window, extracting

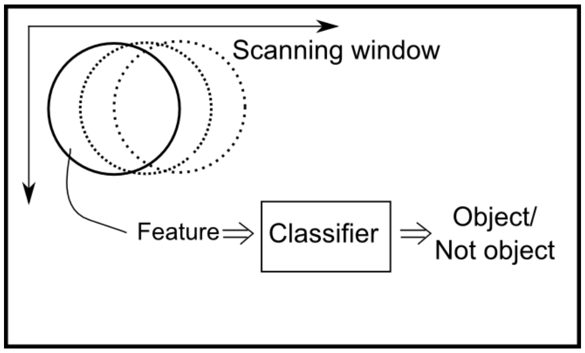


Figure 3.2: A scanning window is moved over the image in raster order. Based on the feature extracted from the window, a classifier determines whether the window contains an object or not.

features from each window and determining whether the feature corresponds to the object or not. In the proposed algorithm, the geometry of sliding windows is circular. The idea is illustrated in Figure 3.2. As will be discussed in section 3.4, circles are approximated as squares for speeding-up the algorithm. This can potentially result in some loss of accuracy. The intensity feature $F(x, y) = [f_1(x, y), f_2(x, y), f_3(x, y), \ldots, f_r(x, y)]$ extracted from the target of interest in the first frame is taken as the reference feature. A sliding window is used to search for the target in the identified search space. Integral histogram is used to compute intensity-based features in each sliding window [9]. The distance

$$D(x, y) = d(f_1(x, y), {f'}_1(x, y)) + d(f_2(x, y), {f'}_2(x, y)) + \ldots + d(f_r(x, y), {f'}_r(x, y))) \quad (3.1)$$

between the feature $F'(x, y) = [f'_1(x, y), f'_2(x, y), f'_3(x, y), \ldots f'_r(x, y)]$ extracted from each sliding window at position (x, y) and the reference feature $F(x, y)$ are recorded. The notation $d(.)$ represents some distance measure between two intensity histograms. The position of the target is taken as the location (x, y) that gives the smallest value of $D(x, y)$. This idea is illustrated in Figure 3.3. The distance measure $d(.)$ can be any histogram comparison metric such as

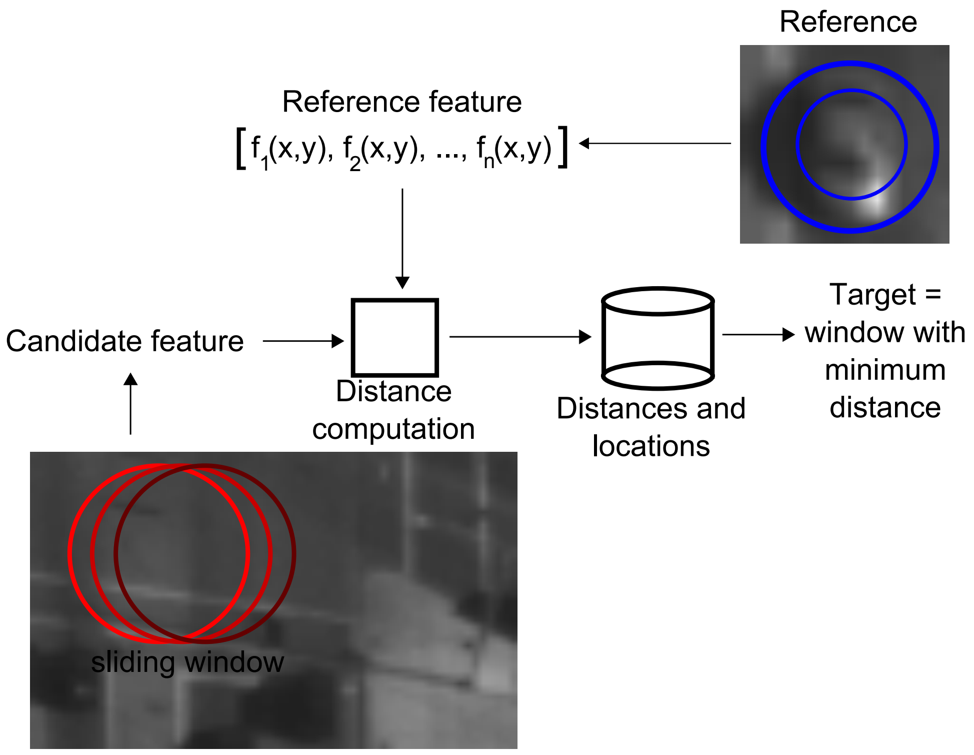


Figure 3.3: Tracking with rotation invariant intensity-based features.

Histogram intersection -

$$d(r,s) = \sum_{i=1}^{N} min(r_i, s_i) \tag{3.2}$$

Log-likelihood statistic

$$d(r,s) = -\sum_{i=1}^{N} r_i log(s_i) \tag{3.3}$$

Chi-square statistic

$$d(r,s) = \sum_{i=1}^{N} \frac{(r_i - s_i)^2}{r_i + s_i} \tag{3.4}$$

Match distance

$$d(r,s) = \sum_{i=1}^{N} |R_i - S_i| \tag{3.5}$$

where $R_i = \sum_{j=1}^{i} r_j$ and $S_i = \sum_{j=1}^{i} s_j$ are the i^{th} bin of the cumulative histograms corresponding to N-bin histograms $r = \{r_i\}_{i=1,2,\ldots,N}$ and $s = \{s_i\}_{i=1,2,\ldots,N}$ respectively. Histogram intersection, log-likelihood statistic and chi-square statistic are bin-bin similarity metrics, meaning they consider only the corresponding bins. If there are slight shifts in intensity histogram due to illumination changes, bin-bin similarity measures may give inaccurate results. For example, consider an 8 bin histogram $r = [1\ 0\ 0\ 0\ 0\ 0\ 0\ 0]$. Let $r' = [0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0]$ be the histogram r shifted by one bin. The distance between $s = [0\ 0\ 0\ 0\ 0\ 1\ 0\ 0]$ and r is the same as the distance between r and r'. Since r' is the histogram of the image produced by a slight change in illumination of the image whose histogram is r, we expect the distance between r and r' to be much smaller than the distance between r and s. This problem is partly solved by using cross-bin similarity metrics. Match distance is an example of a cross-bin similarity metric. It is a special case of Earth Mover's Distance (EMD), originally used for image retrieval [67]. EMD reflects the minimum cost that must be paid to transform one distribution to another. Let $P = \{(x_i, w_i)\}_{i=1,2\ldots m}$ and $Q = \{(y_i, u_i)\}_{i=1,2\ldots n}$ be two feature clusters with x_i and y_i representing the mean of i^{th} cluster, and w_i and u_i representing their weights. The cost (W) of transforming one feature cluster to the other is

$$W = \sum_{i=1}^{m} \sum_{j=1}^{n} d_{ij} f_{ij} \tag{3.6}$$

where d_{ij} is some distance measure between x_i and y_j and f_{ij} is the flow or the amount of mass moved from the i^{th} cluster to the j^{th} cluster. The flow f_{ij} is subject to the following constraints.

$$f_{ij} \geq 0 \quad ; \quad 1 \leq i \leq m, \quad 1 \leq j \leq n \tag{3.7}$$

$$\sum_{j=1}^{n} f_{ij} \leq w_i \quad ; \quad 1 \leq i \leq m \tag{3.8}$$

$$\sum_{i=1}^{m} f_{ij} \leq u_j \quad ; \quad 1 \leq j \leq n \tag{3.9}$$

$$\sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij} = \min\left(\sum_{i=1}^{m} w_i, \sum_{j=1}^{n} u_j\right) \tag{3.10}$$

The Earth Mover's Distance (EMD) between P and Q is defined as

$$EMD\left(P, Q\right) = \frac{\sum_{i=1}^{m} \sum_{j=1}^{n} d_{ij} f_{ij}}{\sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij}} \tag{3.11}$$

3.4 Approximating circular regions

An integral part of our algorithm is computation of histograms within regions defined by circles. A circle can be thought of as a polygon with infinite number of sides. However, this kind of approximation does not fit into the integral histogram framework. Another method is to use a finite number of sampling points on the circle. The point may not fall at the center of a pixel location. This will require costly interpolation. Yet another way is to have a set of 'corrugated' edges as shown in Figure 3.4.

The number of 'ridges' is limited since images are quantized as pixels. The representation in Figure 3.4(a) is the most accurate approximation of the circle among the four shown in the figure.

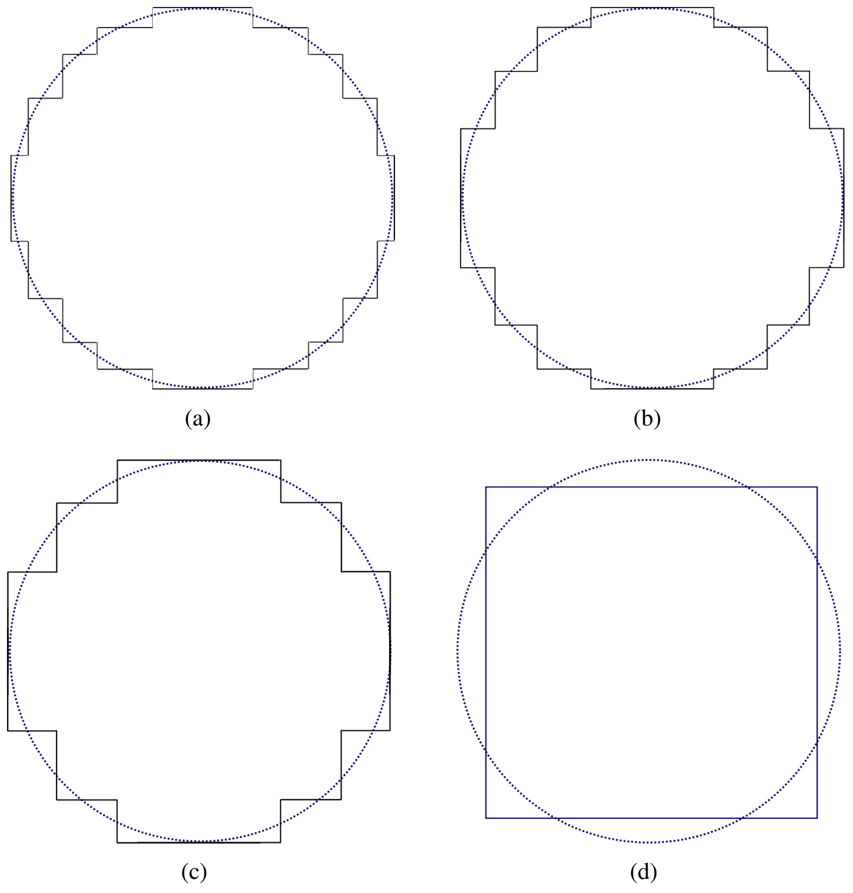

(a) (b) (c) (d)

Figure 3.4: Approximations of circle where (a) is the most accurate while (d) is the least accurate.

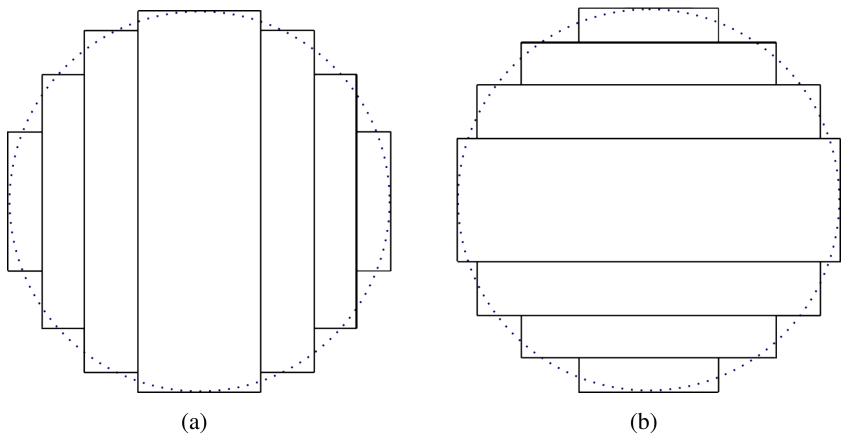


Figure 3.5: Fitting a circle approximation into the integral histogram framework.

The approximations shown in Figure 3.4 can be evaluated using integral histograms by dividing them into rectangular regions horizontally or vertically as shown in Figure 3.5.

To compute the histogram of a rectangular region using integral histogram, 3 summations are required. Since there are 7 rectangular regions in Figure 3.5, there are $3 \times 7 = 21$ summations. The number of summations required is 27, 21, 15 and 3 in Figure 3.4(a) , Figure 3.4(b), Figure 3.4(c) and Figure 3.4(d) respectively. A square is a very crude approximation of the circle, but requires only few computations to calculate the region histogram. It is possible to have a target part fully contained in a square region such that even if it is rotated, the histogram associated with the target part does not change. This idea is illustrated in Figure 3.6.

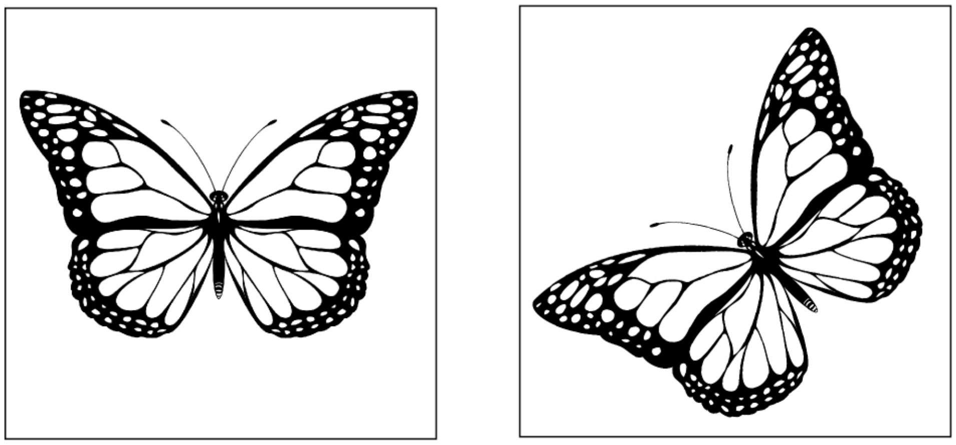


Figure 3.6: Fully enclosed target in sliding window.

3.5 Summary

In this chapter, we presented a method to detect objects in low-resolution high-altitude data using intensity-based rotation invariant features. The speed of the method arises from using integral histogram in a small search space. We also showed that circles could be approximated as squares to allow fast feature computation using integral histogram. The experimental results supporting this method are presented in Chapter V.

CHAPTER IV

GRADIENT-BASED ROTATION INVARIANT FEATURES FOR OBJECT DETECTION

Intensity-based rotation invariant features presented in Chapter III are not expressive enough to be used in object detection. As mentioned previously, a better approach is to use features derived from gradients. An overview of gradient-based features was presented in Chapter I. In this chapter we discuss the method in detail.

A detection algorithm usually consists of a training process. In the training process, a classifier is trained to learn a decision boundary separating positive and negative examples. Once a model is obtained, a sliding window is used to search for possible location of object parts. The detection stage consists of feature extraction in a sliding window, feature classification, detection clustering and part assembly (for a multi-part model). The system architecture of the method is shown in Figure 4.1.

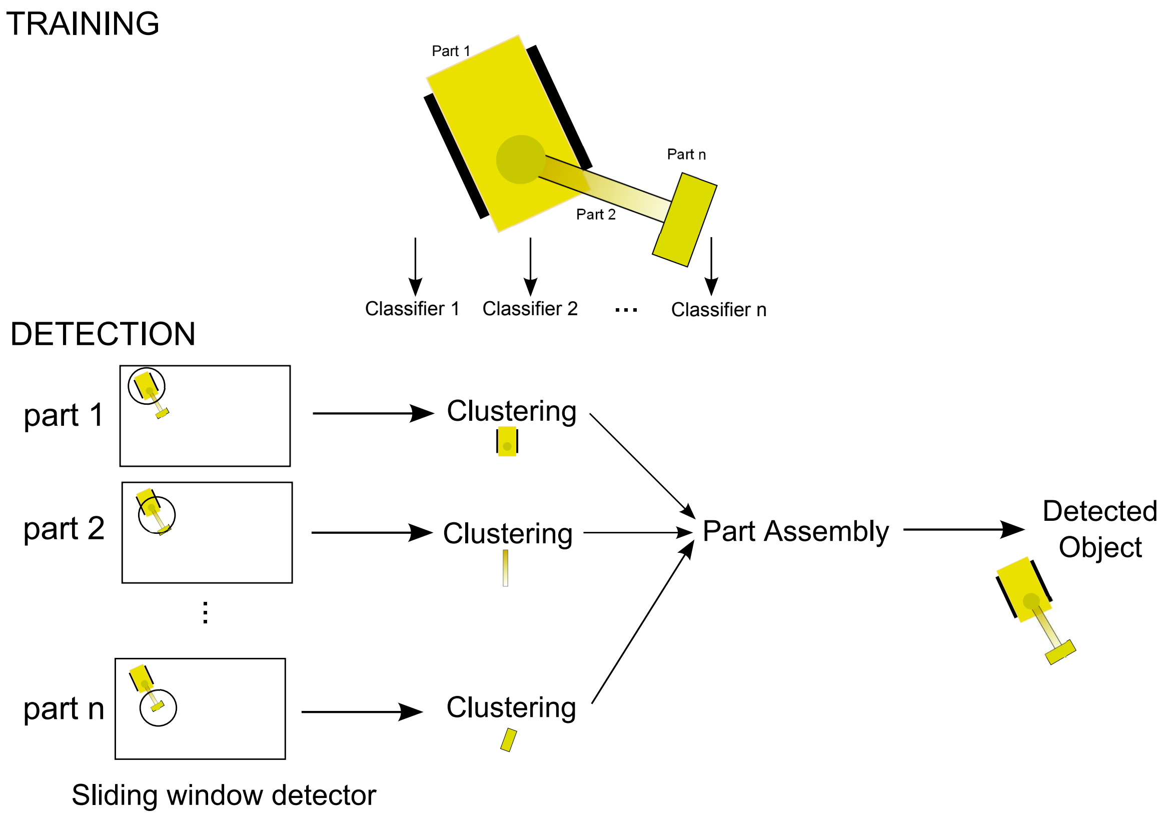


Figure 4.1: Detection based on rotation invariant gradient-based features: system architecture.

4.1 Gradient orientation histogram computation

The methodology for computing gradient orientation histogram is similar to computing the histogram of oriented gradients (HOG) in Dalal-Triggs algorithm. The image is first converted to grayscale. It may be recalled that each part is represented by a feature $F(x, y)$. If the feature is gradient-based, each element $f_i(x, y)$ of the feature is derived from gradient orientation histogram computed in the region. Each region is analogous to a cell in Dalal-Triggs HOG algorithm. The process of gradient computation enhances noise within the region. For best results, the gradient kernel should compute the gradient and at the same time, alleviate the problem of enhancing noise component. This can be done by convolving a gradient kernel such as $[-1, 0, 1]$ with a Gaussian filter to obtain a noise reducing kernel. An example of such a kernel is
$[-0.1286, -0.2310, -0.1523, 0, 0.1523, 0.2310, 0.1286]$.

The gradient of the region in x and y directions, $G_x(x, y)$ and $G_y(x, y)$ respectively, are computed using a noise reducing differentiating kernel described above. The gradient magnitude $G(x, y)$ at location (x, y) is computed as $\sqrt{G_x(x, y)^2 + G_y(x, y)^2}$. The gradient orientation $\theta(x, y)$ at location (x, y) is computed using the four quadrant tangent function

$$\theta(x,y) = \begin{cases} tan^{-1}\left(\frac{G_y(x,y)}{G_x(x,y)}\right), & G_x(x,y) > 0. \\ tan^{-1}\left(\frac{G_y(x,y)}{G_x(x,y)}\right) + \pi, & G_x(x,y) < 0, G_y(x,y) \geq 0. \\ tan^{-1}\left(\frac{G_y(x,y)}{G_x(x,y)}\right) - \pi, & G_x(x,y) < 0, G_y(x,y) < 0. \\ \frac{\pi}{2}, & G_x(x,y) = 0, G_y(x,y) > 0. \\ -\frac{\pi}{2}, & G_x(x,y) = 0, G_y(x,y) < 0. \\ Not\ defined, & G_x(x,y) = 0, G_y(x,y) = 0. \end{cases} \tag{4.1}$$

For a window W of size $(2w + 1) \times (2w + 1)$ centered at location (x, y), an N-bin gradient orientation histogram is constructed by accumulating votes $V(x, y, b)$ from each pixel, as shown in (4.2).

$$h(x,y,b) = \sum_{m=-w}^{w} \sum_{n=-w}^{w} V(x+m, y+n, b)\ where\ 0 \leq b \leq N-1 \tag{4.2}$$

Although this is not a true histogram, we call this a gradient orientation histogram for convenience. Since the angle bins are discretized, the gradient orientation at a pixel may not fall on the center of an angle bin. As an example, let the range $(-\pi, +\pi)$ be divided into 4 bins. The bin centers are, therefore, at $-3\pi/4$, $-\pi/4$, $+\pi/4$ and $+3\pi/4$. The gradient magnitude corresponding to value at angle 0° can vote into the bin centered at $-\pi/4$ or $+\pi/4$. When this kind of situation arises, that is to say, when the value does not fall on a bin center, bilinear interpolation is used to vote into two neighboring bins. Within the window W, let $\theta_1(x, y)$ and $\theta_2(x, y)$ be the bin centers of the bins closest to $\theta(x, y)$. Let $b_1(x, y)$ and $b_2(x, y)$ be the indexes corresponding to bin centers $\theta_1(x, y)$ and $\theta_2(x, y)$ respectively, in the gradient orientation histogram $h(x, y, b)$. When bilinear interpolation is used, the vote $V(x, y, b)$ at the location (x, y) consists of contributions to two bins $b_1(x, y)$ and $b_2(x, y)$ as in (4.3).

$$V(x,y,b) = V(x,y,b_1(x,y))\delta(b - b_1(x,y)) + V(x,y,b_2(x,y))\delta(b - b_2(x,y)) \tag{4.3}$$

where $\delta(.)$ represents the discrete time impulse function. In (4.3), the contributions $V(x, y, b_1(x, y))$ and $V(x, y, b_2(x, y))$ are calculated as

$$V(x,y,b_1(x,y)) = \left(1 - \frac{\theta(x,y) - \theta_1(x,y)}{B}\right) G(x,y) \tag{4.4}$$

$$V(x,y,b_2(x,y)) = \left(\frac{\theta(x,y) - \theta_1(x,y)}{B}\right) G(x,y) \tag{4.5}$$

where

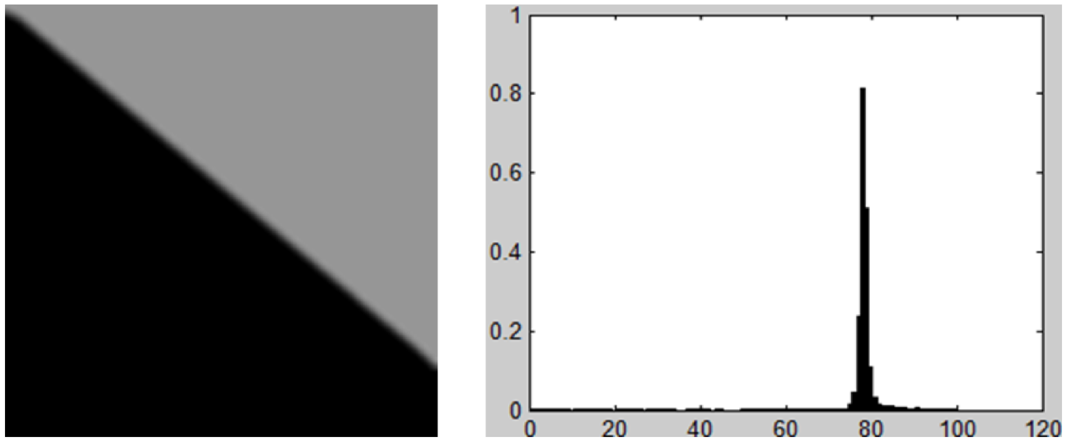
$$b_1(x,y) = \left\lfloor \frac{\theta(x,y) - \frac{B}{2}}{B} \right\rfloor + 1 \tag{4.6}$$

$$b_2(m,n) = \begin{cases} 0, & b_1(x,y) = N - 1. \\ b_1(x,y) + 1, & \text{otherwise.} \end{cases} \tag{4.7}$$

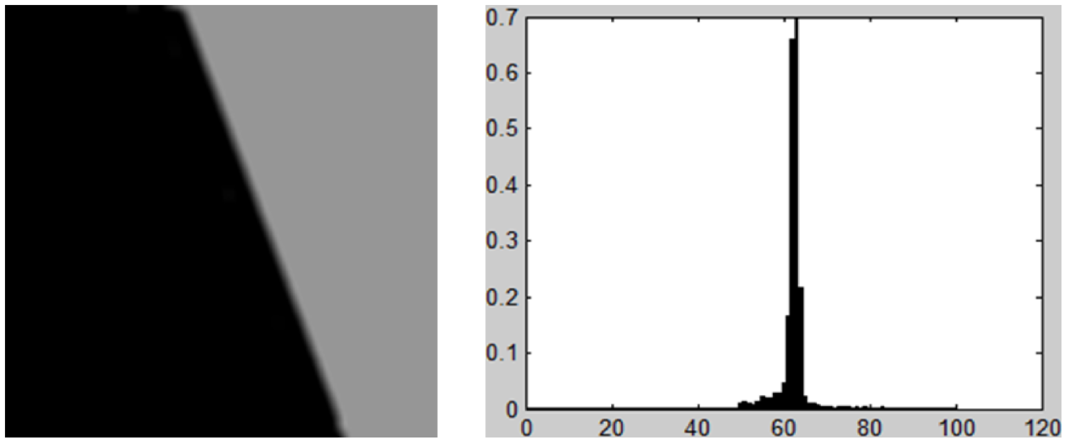
$$\theta_1(x,y) = (b_1(x,y) - 0.5)B \tag{4.8}$$

$$B = \frac{2\pi}{N} \tag{4.9}$$

B is called the histogram bandwidth. The operator $\lfloor . \rfloor$ represents the flooring function.



(a)



(b)

Figure 4.2: Two synthetic images and corresponding gradient orientation histograms. The image in (b) is the rotated version of that in (a). As can be seen, the histogram undergoes a shift.

4.2 Rotation invariance

Gradient orientation histogram of a region is not rotation invariant. As the region rotates, the histogram undergoes a cyclic shift. Two synthetic images and the corresponding histograms are shown in Figure 4.2. In the figure, the second image is the rotated version of the first. If a vector undergoes a cyclic shift, there is a phase change in Fourier domain. Let $v(n)$ be an N-dimensional vector and $v'(n)$ be the vector produced by the cyclic shift of $v(n)$ by m elements, i.e.,

$$v'(n) = v((n-m) mod N) \tag{4.10}$$

where mod represents the modulo operator. The Discrete Fourier Transform (DFT) of the vector $v'(n)$ is

$$\mathcal{V}'(k) = \mathcal{V}(k)e^{-j2\pi\frac{km}{N}} \ where \ k = 0, 1, 2, \ldots, N-1 \tag{4.11}$$

where $\mathcal{V}(k)$ is the DFT of $v(n)$. The magnitude of $\mathcal{V}'(k)$ is

$$|\mathcal{V}'(k)| = |\mathcal{V}(k)e^{-j2\pi\frac{km}{N}}| = |\mathcal{V}(k)||e^{-j2\pi\frac{km}{N}}| = |\mathcal{V}(k)| \tag{4.12}$$

Therefore, the magnitude of DFT of the vector remains unchanged when it undergoes a cyclic shift. Applying this idea to gradient orientation histograms, the magnitude of DFT of gradient histogram remains unchanged when the image region is rotated. This allows us to derive a rotation invariant feature from gradient orientation histogram.

4.3 Deriving the feature from gradient orientation histogram

Based on the analysis in the previous section, each element $f_i(x,y)$ of the feature $F(x,y)$ in (1.1) is

$$f_i(x,y) = \left[|\widetilde{H}_i(x,y,0)|, |\widetilde{H}_i(x,y,1)| \ldots |\widetilde{H}_i(x,y,N-1)|\right] \tag{4.13}$$

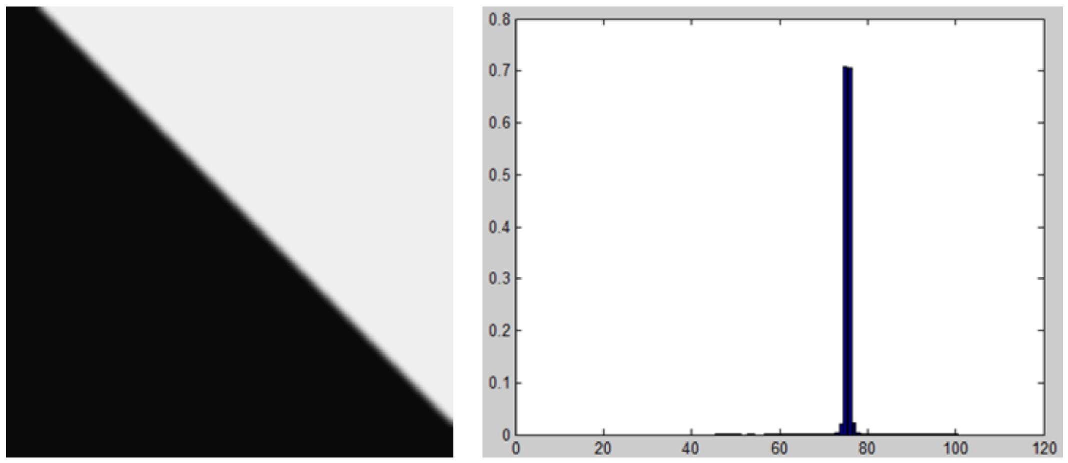
where

$$\begin{aligned}\widetilde{H}_i(x,y,k) &= \mathcal{F}\left\{\tilde{h}_i(x,y,b)\right\}(k) \\ &= \sum_{b=0}^{N-1} \tilde{h}_i(x,y,b)e^{-j2\pi\frac{kb}{N}} \ where \ k = 0, 1, 2, \ldots, N-1\end{aligned} \tag{4.14}$$

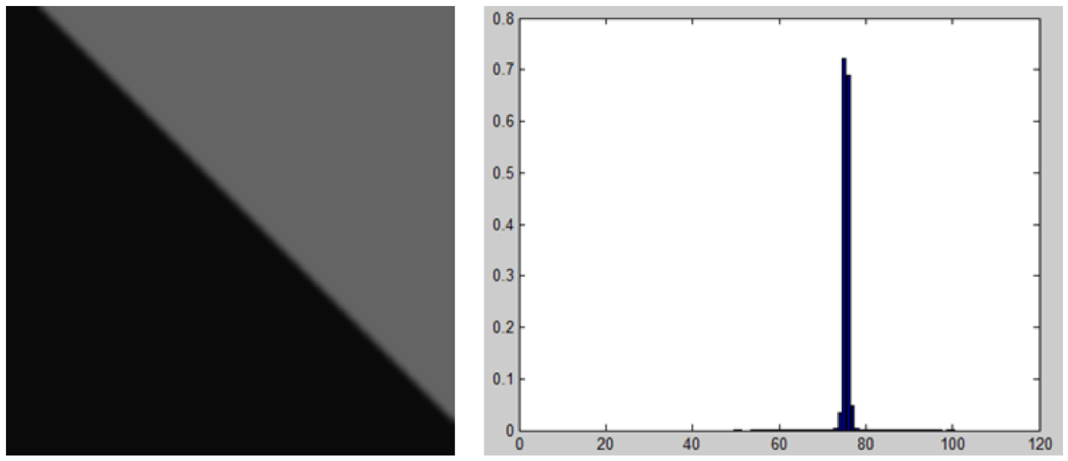
$$\tilde{h}_i(x,y,b) = \frac{h_i(x,y,b)}{\|h_i(x,y,:)\|_2} \tag{4.15}$$

In (4.14), $\mathcal{F}$ denotes the Discrete Fourier Transform (DFT) with b representing spatial domain and k, the frequency domain. The gradient orientation histogram $h_i(x,y,b)$ in region i is normalized as

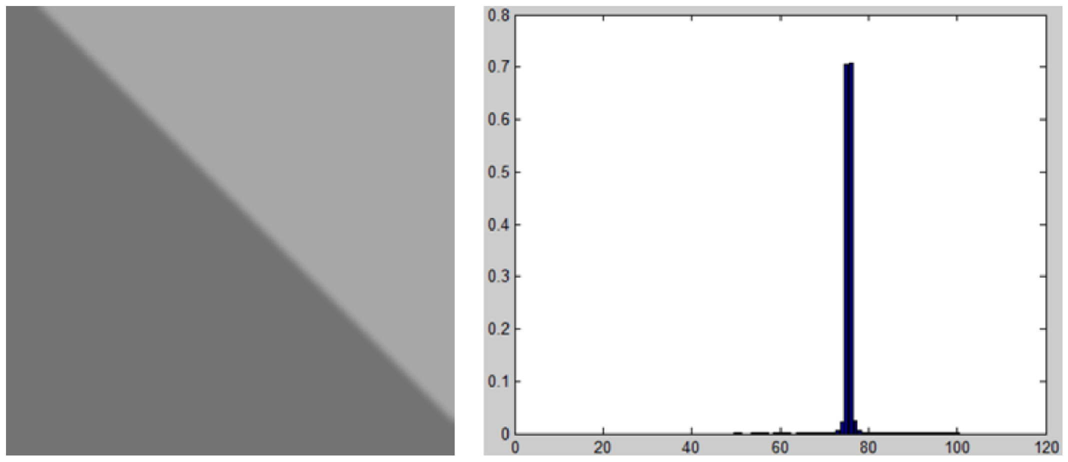
$\frac{h_i(x,y,b)}{\|h_i(x,y,:)\|_2} = \tilde{h}_i(x,y,b)$ to make it robust to illumination and color changes. This can be illustrated with an example. Figure 4.3 shows three images and the corresponding normalized histograms. It can be seen that the histograms corresponding to the three images remain almost the same. An alternative normalization scheme to achieve illumination invariance is to use L_1 norm in (4.15). Using L_1 norm instead of L_2 significantly speeds up computation. However, this may result in a slightly lower accuracy; for example, the highest-cross validation accuracy for the mid-section of aircraft is 98.46% with L_2 norm and 98.27% with L_1 norm. The loss of accuracy is also reported in [4]. For real signals, the DFT magnitude is symmetric. Since histograms are real signals, this idea can be applied to reduce the size of the feature. The DFT magnitude of a 32 bin gradient orientation histogram is shown in Figure 4.4. As can be seen from the figure, barring the DC part (shown in blue), the feature is symmetric. Therefore, for a feature of length N, only $N/2+1$ coefficients need to be used to build the feature.



(a)



(b)



(c)

Figure 4.3: Effect of normalizing gradient orientation histogram - images and corresponding gradient orientation histograms, (a) Image with gray levels 10 and 240, (b) Image with gray levels 10 and 100 and (c) Image with gray levels 115 and 167.

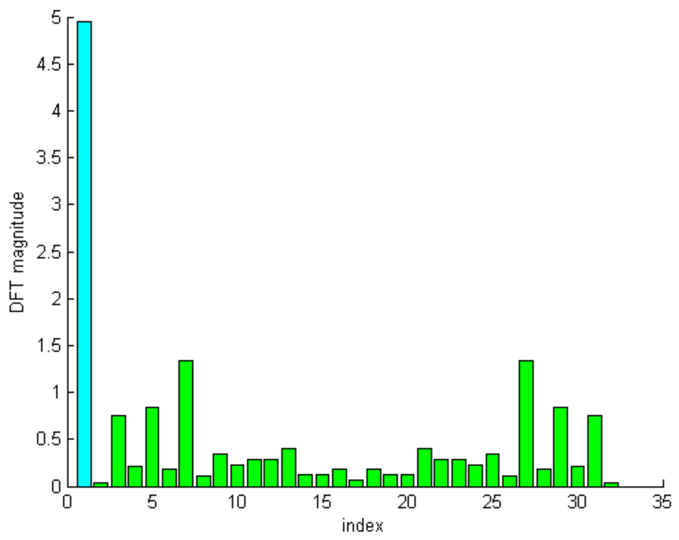


Figure 4.4: Symmetry of DFT coefficients.

4.4 Fast feature computation with Integral DFT

Using the linearity property of the Fourier transform, $f_i(x,y)$ in (4.13) can also be written as

$$f_i(x,y) = \left[\frac{|H_i(x,y,0)|}{\|h_i(x,y,:)\|_2}, \frac{|H_i(x,y,1)|}{\|h_i(x,y,:)\|_2} \cdots \frac{|H_i(x,y,N-1)|}{\|h_i(x,y,:)\|_2} \right] \tag{4.16}$$

where

$$H_i(x,y,k) = \mathcal{F}\left\{h_i(x,y,b)\right\}(k) \; where \; k = 0,1,2,\ldots,N-1 \tag{4.17}$$

It can be shown that the vector

$$\widetilde{f_i}(x,y) = \left[\frac{|H_i(x,y,0)|}{\|H_i(x,y,:)\|_2}, \frac{|H_i(x,y,1)|}{\|H_i(x,y,:)\|_2} \cdots \frac{|H_i(x,y,N-1)|}{\|H_i(x,y,:)\|_2} \right] \tag{4.18}$$

representing normalized magnitudes of Fourier coefficients is proportional to $f_i(x,y)$ and is therefore illumination invariant. To establish the relationship between $\widetilde{f_i}(x,y)$ and $f_i(x,y)$, we start with the Parseval's theorem

$$\sum_{b=0}^{N-1} |h_i(x,y,b)|^2 = \frac{1}{N} \sum_{k=0}^{N-1} |H_i(x,y,k)|^2 \tag{4.19}$$

Using the definition of L_2 norm, Parseval's theorem can also be written as

$$\|h_i(x,y,:)\|_2 = \frac{1}{\sqrt{N}}\|H_i(x,y,:)\|_2 \tag{4.20}$$

Substituting $\|H_i(x,y,:)\|_2 = \sqrt{N}\|h_i(x,y,:)\|_2$ from (4.20) into (4.18) and comparing with (4.16), we have

$$\widetilde{f}_i(x,y) = \frac{1}{\sqrt{N}} f_i(x,y) \tag{4.21}$$

Integral histogram $\psi(x,y,b)$ at a point (x,y) in an image is defined as the histogram collected from the rectangular region with diagonally opposite corners at the point (x,y) and the top-left of the image (origin) [9]. If this representation is available, the histogram $h_i(x,y,b)$ of any region with corners at $(x+w,y+w)$, $(x-w,y+w)$, $(x+w,y-w)$, $(x-w,y-w)$ and centered at (x,y) can be calculated as

$$h_i(x,y,b) = \psi(x{+}w,y{+}w,b) - \psi(x{-}w,y{+}w,b) - \psi(x{+}w,y{-}w,b) + \psi(x{-}w,y{-}w,b) \tag{4.22}$$

where

$$\psi(x,y,b) = \sum_{m=0}^{x}\sum_{n=0}^{y} V(m,n,b) \tag{4.23}$$

This idea is shown in Figure 4.5.

Taking Fourier transform of (4.22),

$$\begin{aligned} H_i(x,y,k) = {} & \mathcal{F}\{\psi(x+w,y+w,b)\}(k) - \mathcal{F}\{\psi(x-w,y+w,b)\}(k) \\ & - \mathcal{F}\{\psi(x+w,y-w,b)\}(k) + \mathcal{F}\{\psi(x-w,y-w,b)\}(k) \end{aligned} \tag{4.24}$$

We define the Integral DFT at position (x,y) as

$$\Psi(x,y,k) = \mathcal{F}\{\psi(x,y,b)\}(k) \; where \; k = 0,1,2,\ldots,N-1 \tag{4.25}$$

Applying the definition of Integral DFT on (4.24),

$$\begin{aligned} H_i(x,y,k) = {} & \Psi(x+w,y+w,k) - \Psi(x-w,y+w,k) \\ & - \Psi(x+w,y-w,k) + \Psi(x-1,y-w,k) \end{aligned} \tag{4.26}$$

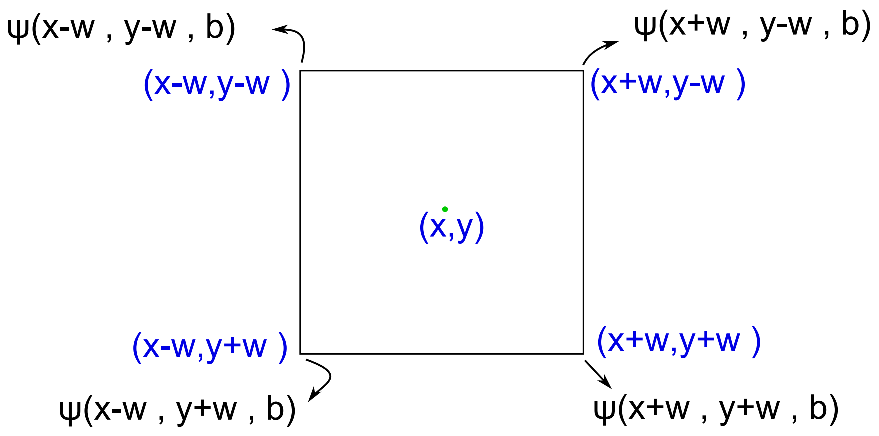


Figure 4.5: Integral histogram representation to compute the histogram of a $(2w+1) \times (2w+1)$ window centered at (x, y).

It may be noted that (4.26) has the same form as (4.22). Therefore, if the Integral DFT at every point is available, $H_i(x, y, k)$ in (4.18), corresponding to a window centered at (x, y) and with corners at $(x+w, y+w)$, $(x-w, y+w)$, $(x+w, y-w)$, $(x-w, y-w)$ can be computed in the same manner as computing integral histogram.

Using the definition of integral histogram in (4.23)

$$\begin{aligned} \Psi(x, y, k) &= \mathcal{F}\left\{\sum_{m=0}^{x}\sum_{n=0}^{y} V(m, n, b)\right\}(k) \\ &= \sum_{m=0}^{x}\sum_{n=0}^{y} \mathcal{F}\left\{V(m, n, b)\right\}(k) \\ &= \sum_{m=0}^{x}\sum_{n=0}^{y} \mathcal{F}\{V(m, n, b_1(x, y))\delta(b - b_1(m, n)) \\ &\qquad\qquad + V(m, n, b_2(m, n))\delta(b - b_2(m, n))\}(k) \\ &= \sum_{m=0}^{x}\sum_{n=0}^{y} g\left\{(m, n, b)\right\}(k) \end{aligned} \tag{4.27}$$

Since $\mathcal{F}$ is linear,

$$\begin{aligned} g\left\{(m, n, b)\right\}(k) &= V(m, n, b_1(m, n))\mathcal{F}\left\{\delta(b - b_1(m, n))\right\}(k) \\ &\quad + V(m, n, b_2(m, n))\mathcal{F}\left\{\delta(b - b_2(m, n))\right\}(k) \end{aligned} \tag{4.28}$$

The terms $\mathcal{F}\left\{\delta(b-b_1(m,n))\right\}(k)$ and $\mathcal{F}\left\{\delta(b-b_2(m,n))\right\}(k)$ can be computed without using FFT, since $\delta(b-b_1(m,n))$ and $\delta(b-b_2(m,n))$ are sparse signals. Using the definition of Fourier transform, (4.28) can be simplified as

$$g\left\{(m,n,b)\right\}(k) = V(m,n,b_1(m,n))e^{-j2\pi\frac{kb_1(m,n)}{N}} + V(m,n,b_2(m,n))e^{-j2\pi\frac{kb_2(m,n)}{N}} \tag{4.29}$$

The space complexity of FFT is $O(NlogN)$, while the space complexity with this method of computation is $O(N)$.

Integral DFT can be computed recursively. For the current pixel position (x,y), the integral DFT is updated as

$$\Psi(x,y,b) \leftarrow \Psi(x-1,y,b) + \Psi(x,y-1,b) - \Psi(x-1,y-1,b) + g\left\{(x,y,b)\right\}(:) \tag{4.30}$$

where $g\left\{(x,y,b)\right\}(:) = [g\left\{(x,y,b)\right\}(0), g\left\{(x,y,b)\right\}(1), \ldots, g\left\{(x,y,b)\right\}(N-1)]$. The method is shown in Figure 4.6.

Several techniques can be employed to speed-up the computation of $\Psi(x,y,b)$ in (4.30). In (4.29), b_1, b_2 and k can only take a value in the range $[0, N-1]$. Therefore the exponentials in (4.29) can be computed using a Look-Up Table (LUT). The summations in (4.30) can be done using Intel Streaming SIMD Extensions (SSE). These can be used on computing platforms with processors that support x86 instruction set, including AMD. SSE uses 128-bit XMM registers to perform operations in parallel.

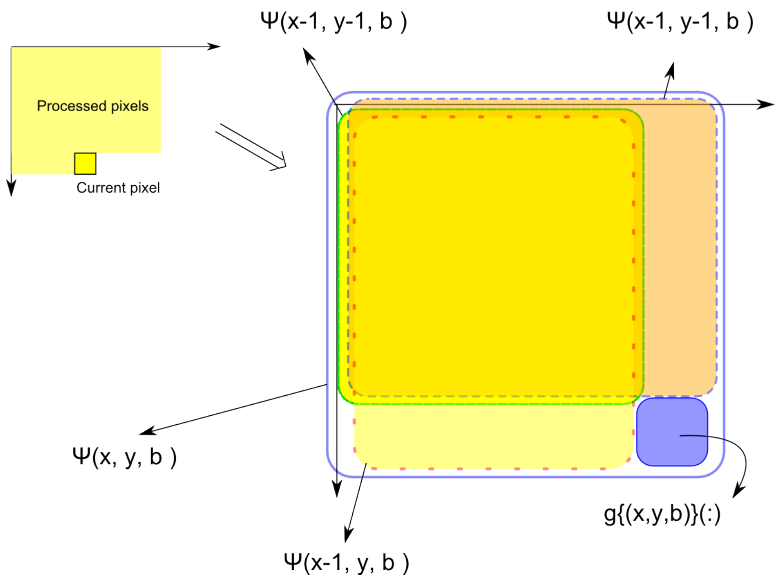
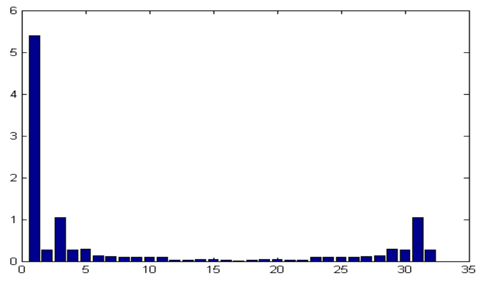


Figure 4.6: Recursive computation of integral DFT.

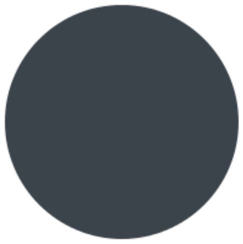
4.5 Circular regions as parts

The magnitude of DFT of circular regions are indistinguishable from that of plane or textured regions, as shown in Figure 4.7.

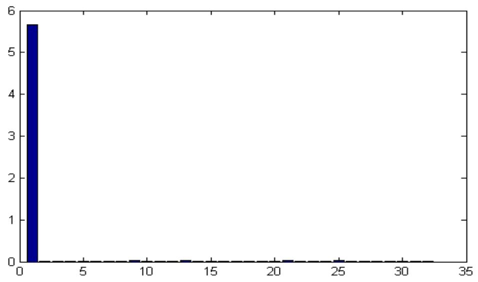
(a)



(b)



(c)



(d)

Figure 4.7: Similarity of features from circular region and textured region, (a) Textured region, (b) Gradient orientation histogram corresponding to textured region, (c) Circular region and (d) Gradient orientation histogram corresponding to circular region.

To solve this problem, instead of taking the magnitude of DFT, HOG feature can be used. Because of the unique geometry of circle, a feature formed by concatenating gradient orientation histograms is rotation invariant. An example of region divisions for a circular region is shown in Figure 4.8.

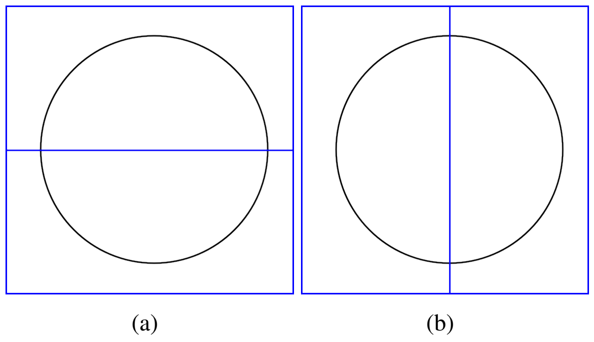


Figure 4.8: Computing features corresponding to circular regions using HOG, (a) 2 cell HOG with horizontal division and (b) 2 cell HOG with vertical division.

4.6 Sliding window shifts

In sliding window based detectors, the number of pixels by which the sliding window is shifted is fixed. Instead of following this approach, the shift δ is taken as a fraction (f) of the size of the window (W), i.e.,

$$\delta = fW \tag{4.31}$$

It can be seen that the number of pixels by which the window is shifted changes with the scale at which the target is searched. This is illustrated in Figure 4.9. The scaling parameter s is defined as the factor by which the size of a search window is increased in the next higher scale. Therefore, for scaling parameter s, the number of pixels by which the window is shifted in the next higher scale is

$$\delta' = sfW \tag{4.32}$$

Sliding window shift proportional to the scale gives significant reduction in computation. Let $M_1 \times M_2$ be the size of the image, W the smallest search window size and N the number of scales. The total number of sliding window evaluations is

$$N_1 = \frac{M_1 M_2}{(fW)^2} + \frac{M_1 M_2}{(sfW)^2} + \frac{M_1 M_2}{(s^2 fW)^2} + \ldots + \frac{M_1 M_2}{(s^N fW)^2} = \frac{M_1 M_2}{(fW)^2} \frac{1 - \left(\frac{1}{s^2}\right)^{N+1}}{1 - \frac{1}{s^2}} \tag{4.33}$$

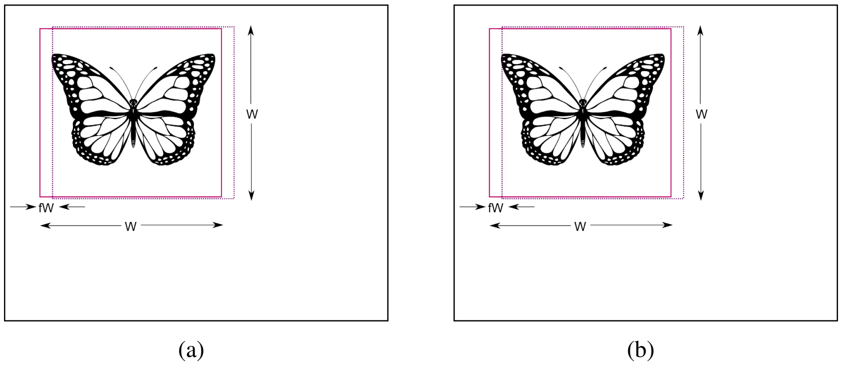


Figure 4.9: The number pixels by which the window is shifted is proportional to the scale, (a) Lower scale and (b) Higher scale.

For large values of N, since $s > 1$, the equation above can be approximated as

$$N_1 \approx \frac{M_1 M_2}{(fW)^2} \frac{s^2}{s^2 - 1} \tag{4.34}$$

In contrast to this, if a fixed slide of 1 pixel is used at all scales, the number of sliding window evaluations is

$$N_2 = (N+1) M_1 M_2 \tag{4.35}$$

The performance improvement is thus

$$\frac{N_2}{N_1} \approx \frac{(N+1)(fW)^2(s^2-1)}{s^2} \tag{4.36}$$

4.7 Classification with support vector machines

In the proposed method, a two class classifier is used to identify parts of interest. This kind of classifier is essentially a function f that maps an N-dimensional feature F to Y such that

$$f : F \to Y,\ x \in \mathbb{R}^N,\ y \in \{+1, -1\} \tag{4.37}$$

+1 represents a positive example and -1, a negative example. If the feature F corresponds to a positive example, f maps to +1 and f maps to -1 otherwise. When such a classifier is trained, it finds a decision boundary separating positive examples and negative examples. This decision boundary can be linear (hyperplane) or non-linear.

One of the most popular classifiers is the Support Vector Machine (SVM). The popularity of SVMs stems from the fact that they perform very well on real world data. SVMs generate optimal hyper planes in the feature space for classifying data. In fact, several neural networks, radial basis function classifiers and polynomial classifiers are special cases of SVMs. Linear SVM decision score for data point x is

$$f(x) = \beta + w^{\mathsf{T}} x \tag{4.38}$$

w has the same dimensionality as x. β is a scalar value called bias. The actual output of the classifier is $sign(f(x)$. Data points are not always linearly separable. This problem is tackled by transforming the data using a function $\phi(.)$ into a space where data is linearly separable. If the data is transformed in this way, the decision function becomes

$$f(x) = \sum_{i=1}^{n} \alpha_i K(x_i, x) + \beta \tag{4.39}$$

$K(.,.)$ is called kernel function. In vector form, $K(x_i, x) = \phi(x_i).\phi(x)$. Vectors x_i are called support vectors and α_i are scalar coefficients. The number of support vectors in (4.39) is n. A commonly used kernel function is the Radial Basis Function (RBF) -

$$K(x_i, x) = e^{-\gamma \|x_i - x\|^2} \tag{4.40}$$

The advantage of using RBF is that there is only a single parameter (γ).

An RBF kernel SVM model is built for each part of the object. A sliding window corresponding to each part is used to extract part features. For the extracted feature, the classifier response is evaluated at every sliding window location.

4.7.1 Probabilistic scaling of SVMs

In several cases, it is advantageous for the classifier to output a confidence value or probability for the classification. Probabilistic scaling of SVMs is based on the fact that points that are closer to the hyperplane have a lower confidence and those that are farther away have higher confidence associated with them. Since confidence values lie between 0 to 1, a function that maps SVM decision scores to a range between 0 to 1 is used to achieve this -

$$\sigma(z) = \frac{1}{1 + e^{Az+B}} \tag{4.41}$$

where $z = f(x)$ is the SVM decision score. The values A and B are found by minimizing the Cross-Entropy (CRE). Cross entropy is defined as

$$CRE = \sum_i y_i log(z_i) + (1 - y_i) log(1 - z_i) \tag{4.42}$$

where y_i is the class value (-1 or +1) and $z_i = \sigma(f(x_i))$

4.7.2 Bootstrapping

In real-life, only a limited number of positive examples is available. On the other hand, the number of negative examples is virtually unlimited. Using a large number of examples to train an SVM, or any classifier, is expensive in terms of training time. Therefore, there is a need to pick the 'right' examples to train the system. These examples are picked using a process called bootstrapping. Bootstrapping is an iterative process for building a training set. A few examples are first used to train an SVM. The trained SVM is then tested on a large collection of examples. Those examples that are misclassified are added to the training set.

4.8 Non-maxima suppression

When a detector is used with a scanning window detector, several positive responses are produced in close proximity to the target as shown in Figure 4.10(a). To deal with this, multiple

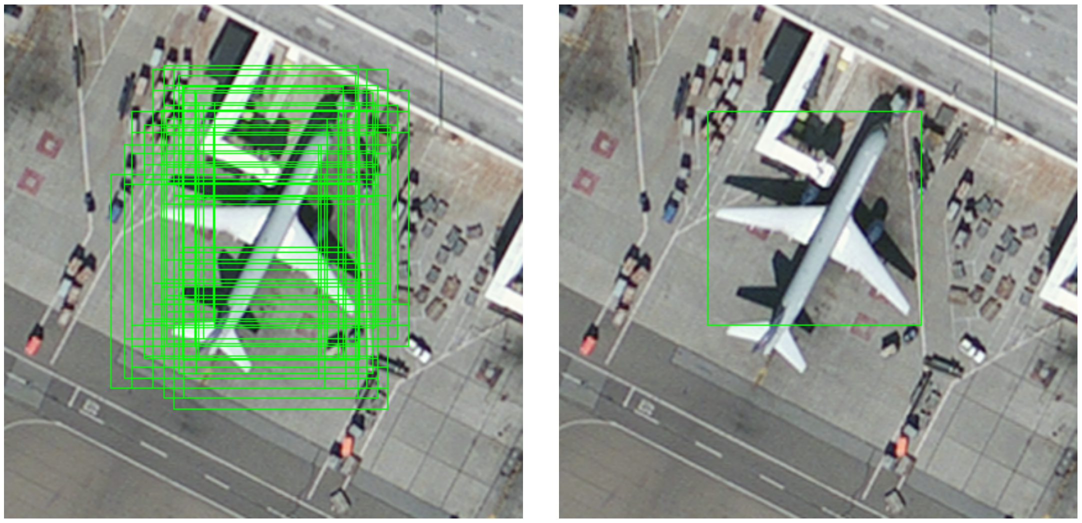

(a)
(b)

Figure 4.10: Detecting the mid-section of an aircraft, (a) Multiple detections given by an SVM and (b) Fused detection after non-maxima suppression.

detections are fused into a single detection as in Figure 4.10(b). For fusing multiple detections, agglomerative clustering algorithm is used. The algorithm is as follows

i. Choose bounding box D corresponding to the detection with highest confidence in the list of detections.

ii. Find all detections with sufficient overlap to D. Overlap between two regions R_1 and R_2 is computed as

   $overlap = \frac{area(R_1 \cap R_2)}{area(R_1 \cup R_2)}$

iii. To merge detections, find the average of all overlapping detections obtained in (ii) and assign the confidence value corresponding to D to it.

iv. Remove all merged detections from the list.

v. Repeat (i) to (iv) until there are no more merges.

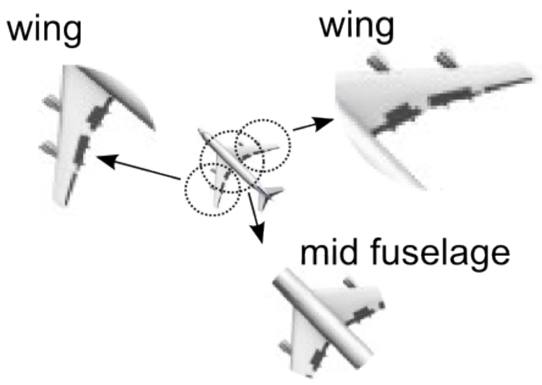


Figure 4.11: An example of spatial arrangement of object parts.

4.9 Object model and part clustering

After detecting parts of an object, if the object is modelled as multi-part, there should be a way to determine whether a collection of detected parts forms an object. The shape of an object is dictated by the relative scale and position of individual parts. Figure 4.11 shows an aircraft with two of its visually salient parts, the mid-section and the wings.

Denoting the size of the detection of mid-section as s_m, the size of detection of wing as s_w, the center of the first wing as c_{w1}, the center of the second wing as c_{w2} and the center of the mid section as c_m, we can construct the following rules.

$$\frac{s_m}{s_w} \approx 1 \tag{4.43}$$

$$\frac{s_m}{d(c_m, c_{w1})} = \frac{s_m}{d(c_m, c_{w2})} = constant \tag{4.44}$$

$$d(c_m, c_{w1}) < d(c_{w1}, c_{w2}) \tag{4.45}$$

where $d(p_i, p_j)$ represents the Eucledian distance between points p_i and p_j.

The method described above requires visual inspection and manual construction of rules. A more principled way to approach this problem is to model the set of detections as a graph. A

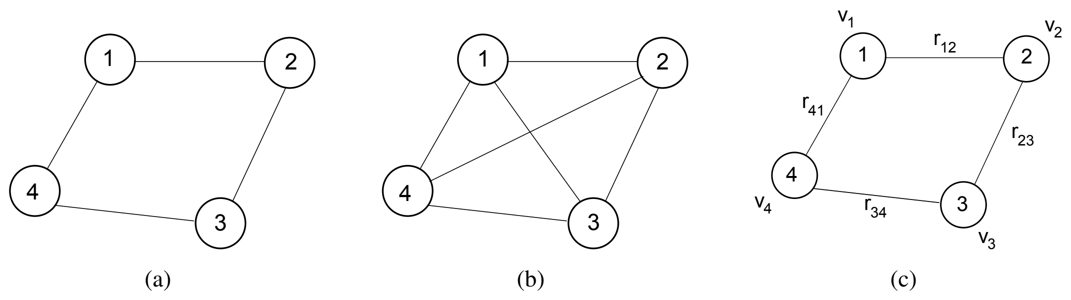


Figure 4.12: Types of graphs, (a) Simple graph, (b) Fully connected graph and (c) Attributed Relational Graph.

graph consists of a set of points called nodes or vertices connected by lines or arcs (called edges). Formally, graph $\mathcal{G}$ is defined as a double, $\mathcal{G} = (V, E)$. V represents the set of vertices and E, the set of edges. Since every node is not necessarily connected to every other node, $E \subseteq V \times V$. In a fully connected graph, every node is connected to every other node. A graph can be augmented by attaching attributes to its nodes and vertices. Such a graph is called an Attributed Relational Graph (ARG). An attributed relational graph with n nodes can be defined as a quad

$$\mathcal{G} = (V, E, A_V, A_E) \tag{4.46}$$

where

$$A_V = \{v_i \mid 1 \leq i \leq n\} \tag{4.47}$$

$$A_E = \{r_{ij} \mid 1 \leq i \leq n,\ 1 \leq j \leq n\} \tag{4.48}$$

In A_V and A_E, v_i and r_{ij} are node and edge attributes respectively. Figure 4.12 shows a simple graph, a fully connected graph and an Attributed Relational Graph.

In the proposed method, a set of detections and their relationships can be modeled as a fully connected ARG. The node attributes are the size of the detected part, defined as the radius of the detection circle or the side of the detection square (s_i) and the detection confidence (p_i). The edge attributes are the distances d_{ij} between two parts. An example is shown in Figure 4.13.

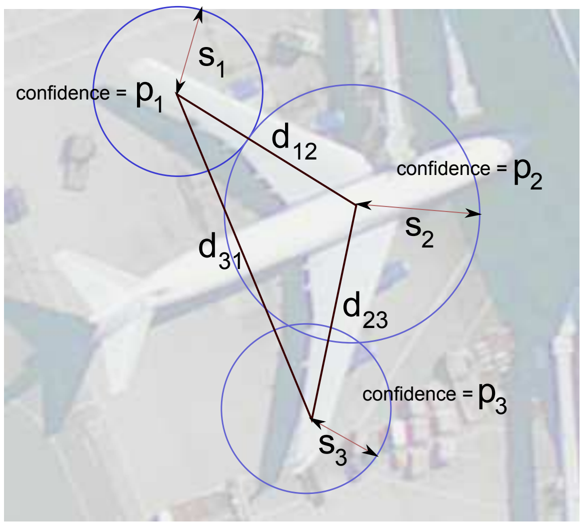


Figure 4.13: Object with graph model overlaid. The corners of the triangle represent graph nodes.

The size of a detected part and the distance between two detections are covariant with scale. To deal with scale, for an ARG representing a cluster of part-detections, we can derive the following signature

$$f_G = \left[\frac{1}{s_1}\boldsymbol{S} \;\; \frac{1}{s_1}\boldsymbol{D} \;\; \boldsymbol{P} \right] \tag{4.49}$$

where $\boldsymbol{S} = [s_2 \; s_3 \ldots s_n]$, $\boldsymbol{D} = [d_{12} \; d_{13} \ldots d_{23} \; d_{24} \ldots]$ and $\boldsymbol{P} = [p_1 \; p_2 \ldots p_n]$

In a fully connected graph, several edges may be redundant. Removing such redundancies can reduce the size of $\boldsymbol{D}$, thereby reducing the size of the signature f_G. One way to do this is to construct a Laman graph [68]. It may be noted that using a formulation similar to (4.43)-(4.45) may produce a more compact description of object configuration. However, in most cases, since the number of parts is limited, the size of f_G will not introduce any significant computational disadvantage. In (4.49), several attributes that may be unrelated to each other are concatenated to form a heterogeneous feature. A learning method that classifies such a feature should assign a relative importance (weights) to each attributes. A linear SVM is a natural choice, as it assigns a relative weight to each attribute. Linear SVM decision score in (4.38) can be written as

$$\begin{aligned} f(x) &= \beta + w^{\intercal} f_G \\ &= \beta + [w_S \; w_D \; w_P]^{\intercal} f_G \end{aligned} \tag{4.50}$$

where w_S, w_D and w_P are the weights of the attribute $\frac{1}{s_1}\boldsymbol{S}$,$\frac{1}{s_1}\boldsymbol{D}$ and $\boldsymbol{P}$ respectively.

4.10 Parallelization framework

With the advent of parallel computing systems such as multicore chips, ASICs, FPGAs and GPUs there is a need for algorithms to be parallelizable. This is especially important when dealing with large imagery such as those captured from flying platforms or satellites. There are computer vision algorithms that are not parallelizable, examples include active contour computation and the mean-shift segmentation algorithm. Fortunately, sliding window detection is a parallelizable algorithm. In the proposed method, detecting parts is also parallelizable, as they are done using different sliding window detectors.

At the heart of any parallel computing algorithm is the concept of a thread. A thread is a single task running on a single processing unit (e.g., a single core on x86, an SIMD engine on the GPU). The inherent parallelism of the proposed part based algorithm allows us to parallelize the operations on each part and the operations on each sliding window. A single thread operates on each sliding window as shown in Figure 4.14. With this strategy no detection is missed in contrast to one in which a thread operates on each region of the image as shown in Figure 4.15.

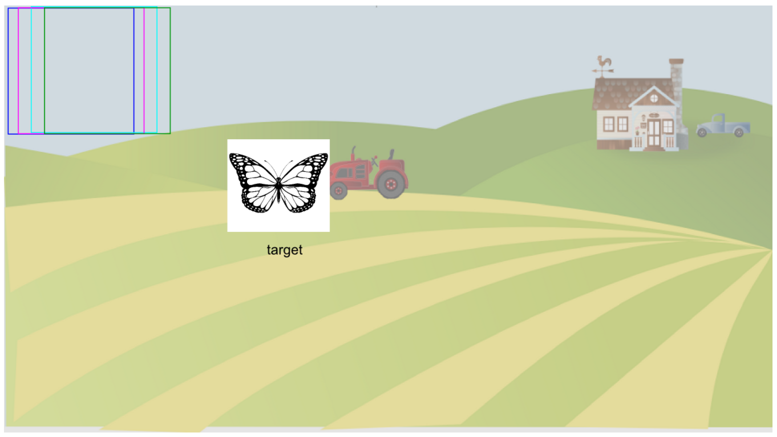


Figure 4.14: Sliding window thread assignment. Each square represents a thread operating on a sliding window.

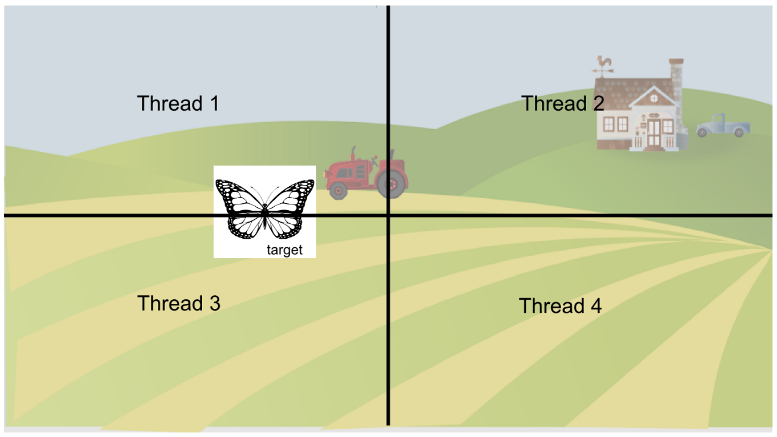


Figure 4.15: Dividing the image among different threads. Each thread operates on a single quadrant.

4.11 Summary

In this chapter, a method to detect objects using gradient-based rotation invariant features was presented. The key ideas in achieving rotation invariance are using a circular division and taking the magnitude of DFT coefficients of gradient orientation histograms. Illumination invariance is achieved by normalizing the gradient orientation histograms. As in the case of intensity-based features, squares are used to approximate circles to allow for fast computation in an integral computation framework. A novel integral computation framework called Integral DFT was also presented. The algorithm is made further robust by designing a part-based method. Parallelization schemes to speed-up computation were also outlined.

CHAPTER V

EXPERIMENTAL RESULTS

In this section, the analysis and experimental results supporting the proposed methods are presented. In section 5.1, the experimental results on tracking vehicles using intensity-based features are presented. The results and analysis of gradient-based features are detailed in section 5.2.

5.1 Object tracking with rotation invariant intensity-based features

Rotation invariant intensity-based features are tested on tracking vehicles in WAMI data. The data is captured using cameras mounted on flying platforms at a height of approximately 7000 ft. The camera captures images of size 2672×4008 at a rate of 2 frames per second.

For speed considerations, instead of taking circular region, concentric square regions are taken to compute the feature. This idea is explained in section 3.4. Specifically for all targets, two square regions are taken. The inner one covers the vehicle. The size of the outer one is about twice that of the inner one. This selection is made manually in the first frame. Since cars are rigid, a single-part model is used.

Methods such as HOG, SIFT and SURF fail to detect objects of interest in WAMI data. Detections given by the HOG descriptor are shown in Figure 5.1. SIFT keypoints are shown in Figure 5.2. However, there are no matches between the target image and the frame. As shown in Figure 5.3, SURF keypoints are incorrectly matched between the target image and the frame.

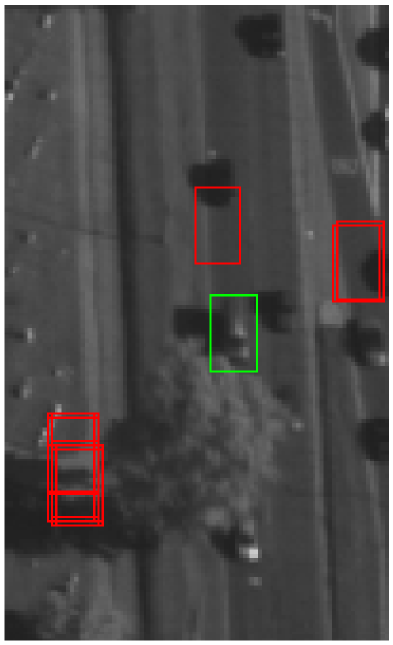


Figure 5.1: Detections with HOG. Red rectangles are the closest eight detections given by HOG. The green rectangle represents the actual target.

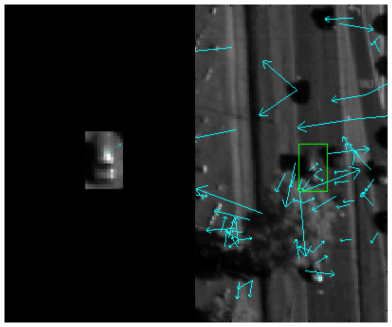


Figure 5.2: SIFT keypoints. The actual target is shown in the green rectangle. There are no matches between the two images.

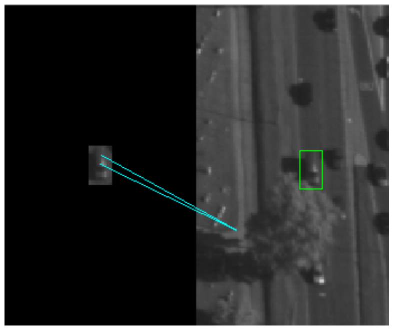


Figure 5.3: Incorrect matches with SURF keypoints. The actual target is shown in the green rectangle.

In [69], a covariance matrix based feature is presented. The feature f_k is associated directly to the pixel coordinates as shown in (5.1).

$$f_k = [x \; y \; I(x,y) \; I_x(x,y) \; I_y(x,y)] \tag{5.1}$$

where (x, y) is pixel coordinate, $I(x, y)$, the intensity at (x, y), $I_x(x, y)$, the gradient along x direction and $I_y(x, y)$, the gradient along y direction. Region covariance matrix is defined as

$$C_R = \frac{1}{MN}\sum_{k=1}^{MN}(f_k - \mu_R)(f_k - \mu_R)^T \tag{5.2}$$

In (5.2), C_R is the $d \times d$ covariance matrix corresponding to the $M \times N$ region R. μ_R represents the vector of the means of features in region R. The distance metric for this tracker uses the sum of squared logarithms of generalized Eigenvalues to compute the dissimilarity between the covariance matrices. The distance metric ρ between two covariance matrices C_i and C_j is defined in (5.3).

$$\rho = \sqrt{\sum_{k=1}^{d} ln^2 \lambda_k(C_i, C_j)} \tag{5.3}$$

In (5.3), $\lambda_k(C_i, C_j)$ are the generalized Eigenvalues of C_i and C_j. The covariance matrix 'balances-out' illumination changes, and consequently the changes is target intensities also disappear. Target intensities play a significant role in recognition when resolution is extremely low. Furthermore, distance computation in covariance tracking is relatively expensive. Mean-shift tracking is another histogram based tracking method [70]. It computes the most probable location of the target based on mean-shift iterations. For target modeling, it uses a metric based on Bhattacharya coefficient. For two normalized m-bin histograms $p = \{p_i\}_{i=1,2,\ldots,m}$ and $q = \{q_i\}_{i=1,2,\ldots,m}$ ($\sum_{i=1}^{m} p_i = 1$ and $\sum_{i=1}^{m} q_i = 1$), the sample estimate of the Bhattacharya coefficient is

$$\rho(p,q) = \sum_{i=1}^{m} \sqrt{p_i q_i} \tag{5.4}$$

Using (5.4), the distance between the two distributions is defined as

$$d(p,q) = \sqrt{1 - \rho(p,q)} \tag{5.5}$$

Reilly et al. have presented a method in [71] to track targets in CLIF data. The method finds a background model as the median of several frames. False detections are eliminated by suppressing the gradient. The method works well when target velocities are high enough to give distinguishable blobs through background subtraction, and fails when velocities are low. Figure 5.4 shows an image and the corresponding background subtracted version along with ambiguous or missed detections.

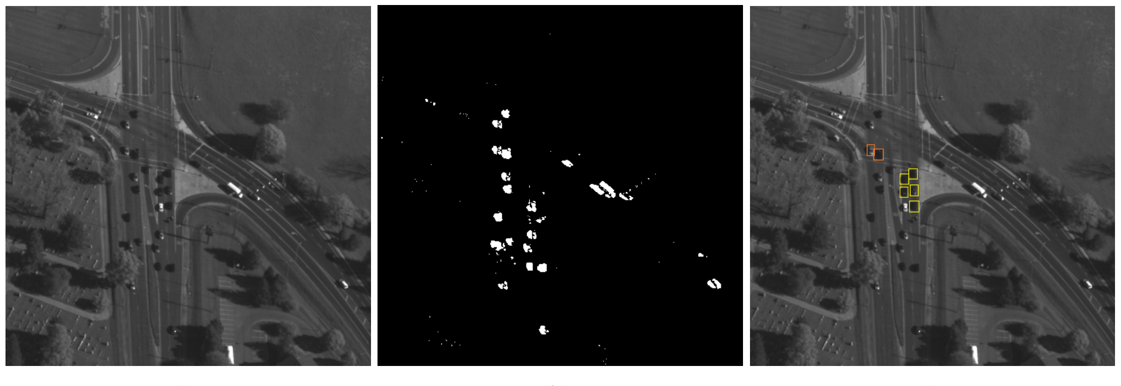

(a) (b) (c)

Figure 5.4: Examples of target misses in methods based on background subtraction, (a) Frame, (b) Frame with background subtracted and (c) Missed or ambiguous detections. Those in yellow are missed and those in red are ambiguous.

Table 5.1: Performance comparison of the proposed method with mean-shift and covariance tracker.

	Number of targets tracked (out of 23 targets)
Mean-shift tracker	2
Covariance tracker	12
Proposed method	22

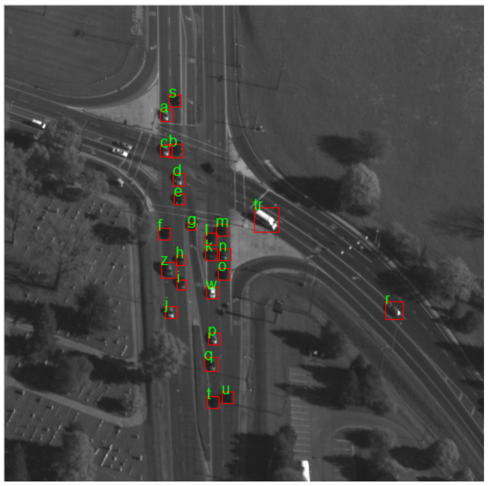
Method in [71] misses 7 targets after the first step. This is because the targets are either moving slowly or moving in close proximity to each other. Our tracking results are compared with covariance tracker and meanshift tracker. Figure 5.5 shows the result of tracking with the proposed descriptor in a video sequence in WAMI. Twenty two out of 23 moving targets are tracked accurately using the proposed method. Figure 5.6 shows tracking the same targets using mean-shift. As can be seen, only two target are tracked correctly. This is because mean-shift tracking cannot cope with large target movements. For the computation of the mean-shift vector, successive target locations need to be overlapping.

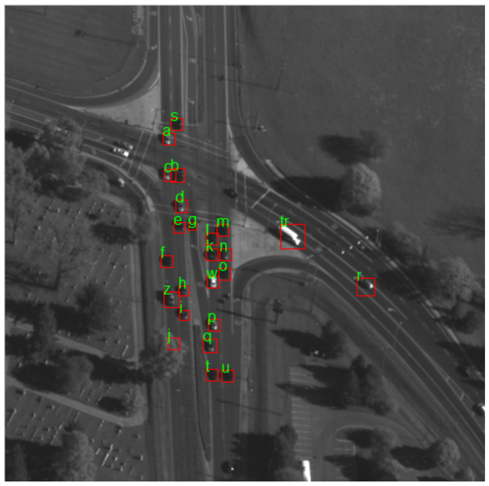
Figure 5.7–5.9 shows the pixel errors for three targets.

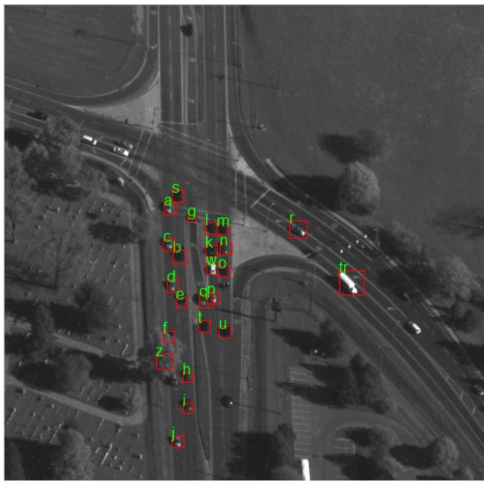
Mean-shift can track only 2 targets in the sequence.

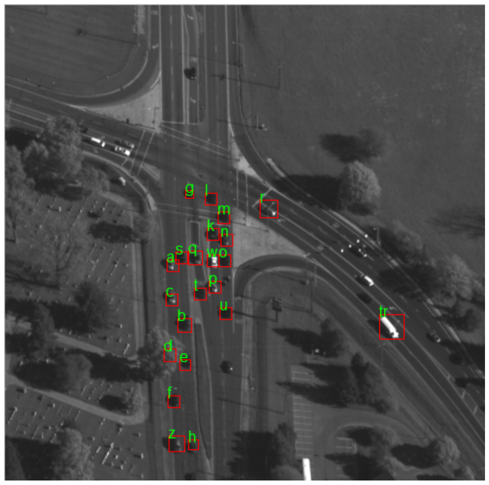
Tracking results with covariance matrices are shown in Figure 5.10.

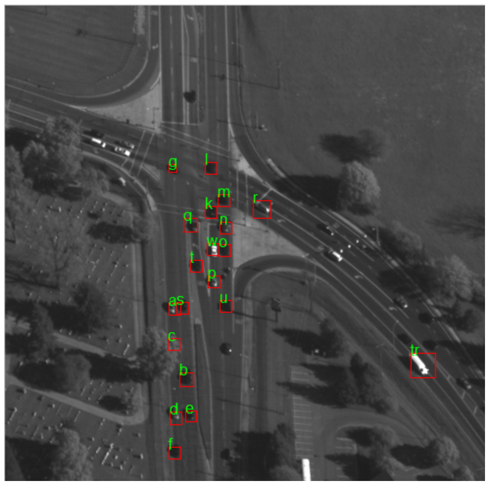
As can be seen, covariance tracker tracks only 12 out of 23 targets. Table 5.1 summarizes the comparison of tracking performance. The comparison is based on 23 targets selected in the first frame. A target is taken as tracked if there is atleast 90% overlap between the ground truth bounding box and the detection bounding box in all frames.


(a)


(b)


(c)


(d)


(e)

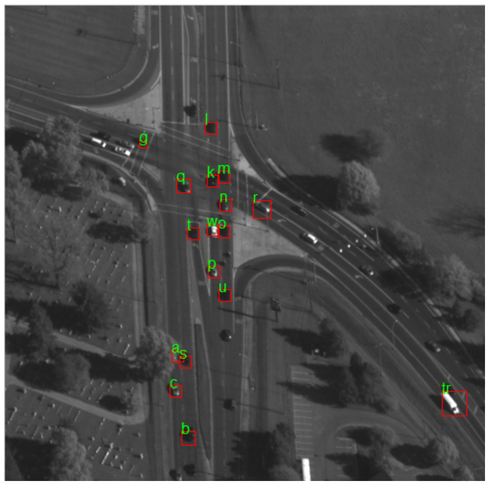
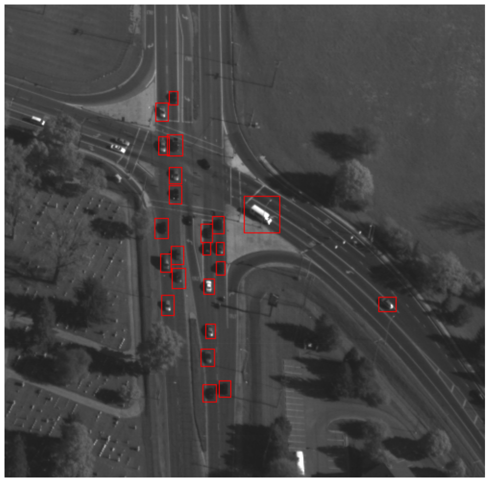
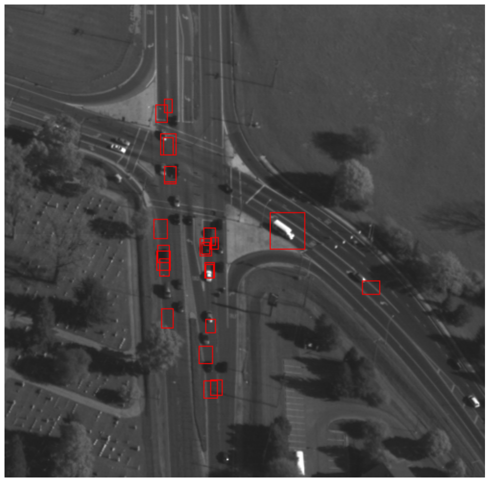
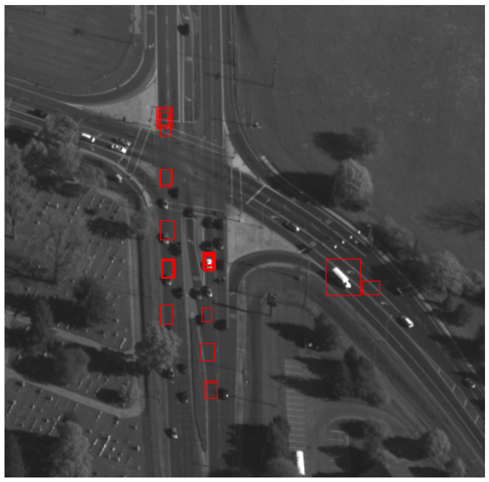
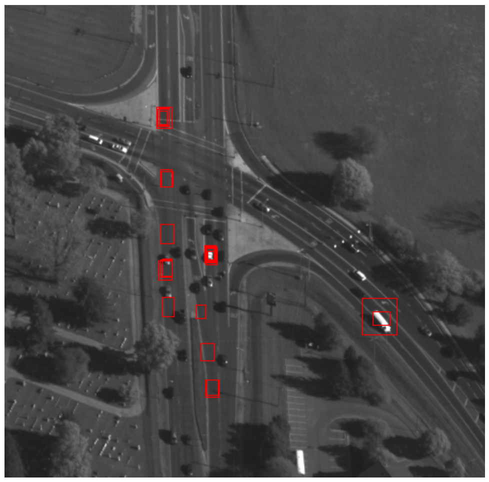

(f)

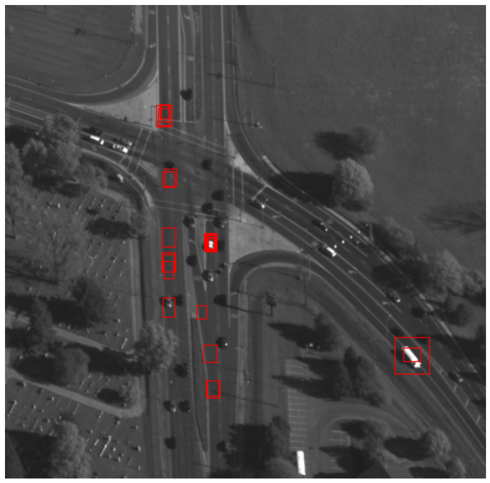
Figure 5.5: Tracking results for (a) frame 1, (b) frame 3, (c) frame 8, (d) frame 12, (e) frame 15 and (f) frame 18 using the proposed method.


(a)


(b)


(c)


(d)


(e)

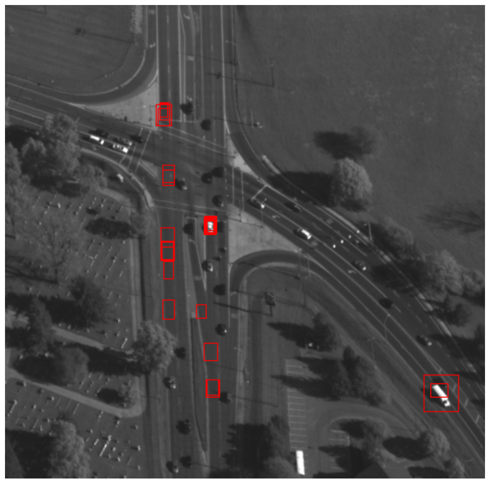

(f)

Figure 5.6: Tracking results for (a) frame 1, (b) frame 3, (c) frame 8, (d) frame 12, (e) frame 15 and (f) frame 18 using mean-shift.

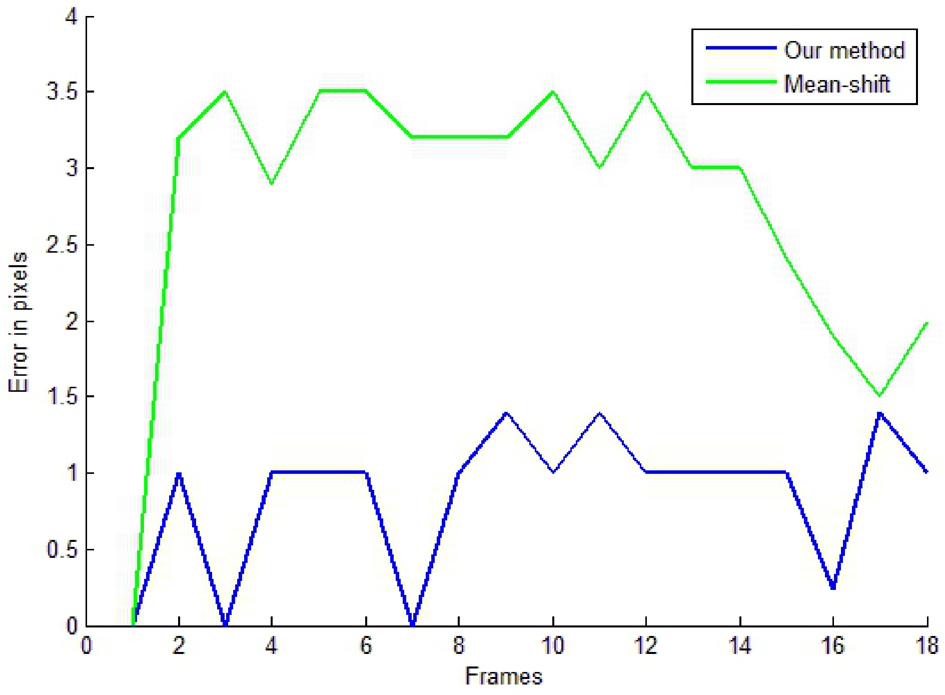


Figure 5.7: Error plot for target 1.

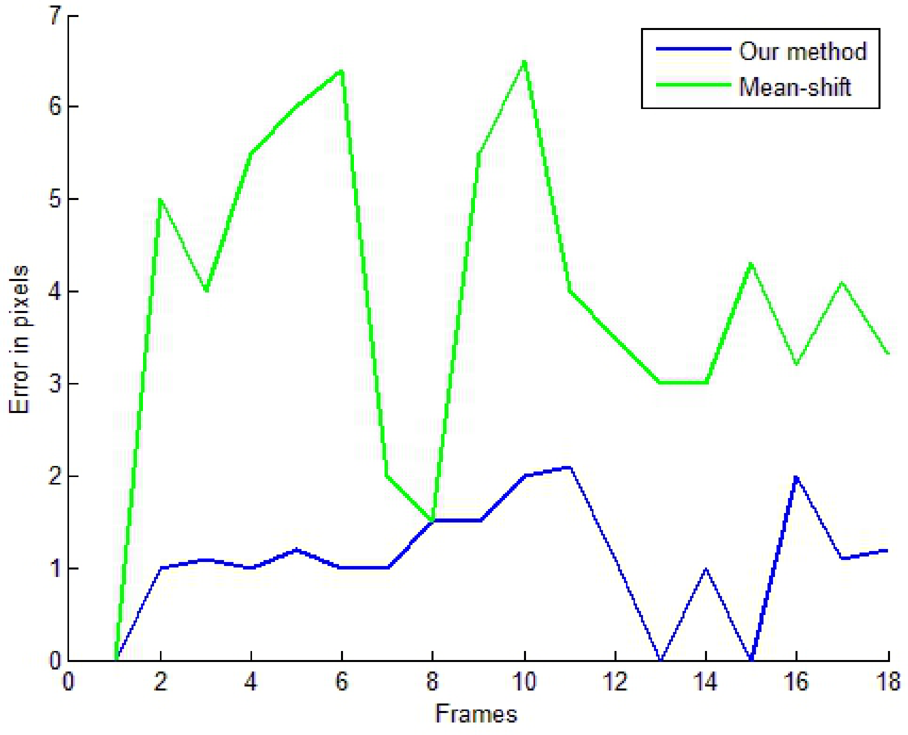


Figure 5.8: Error plot for target 2.

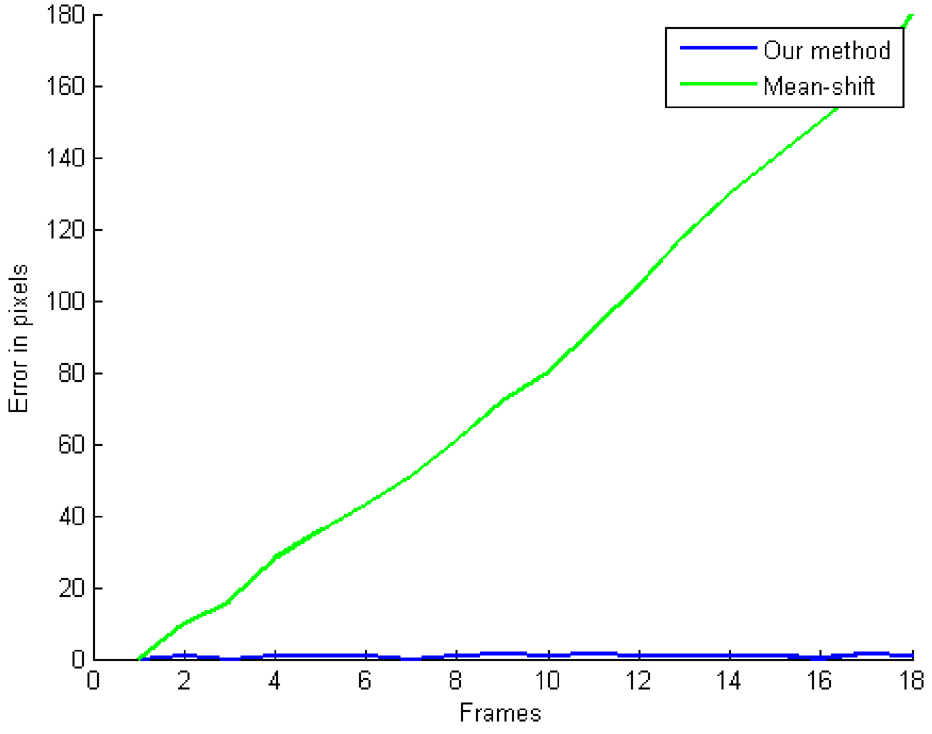
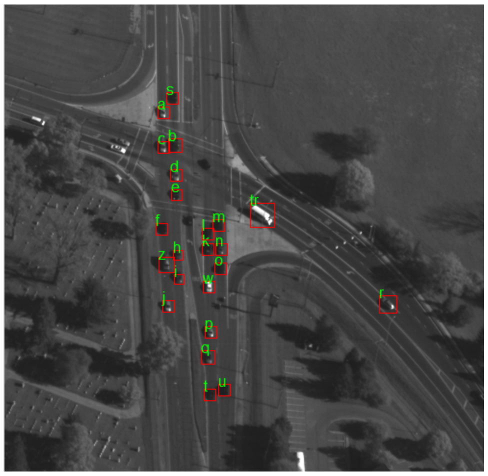
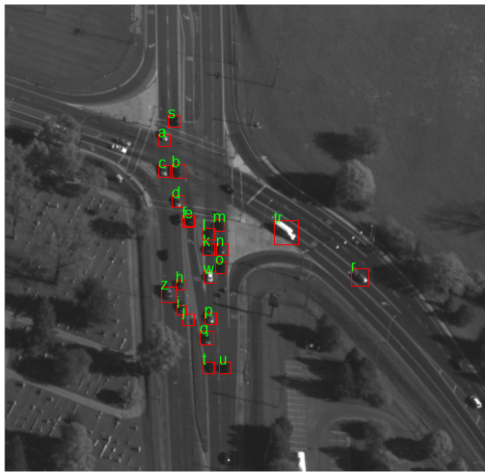
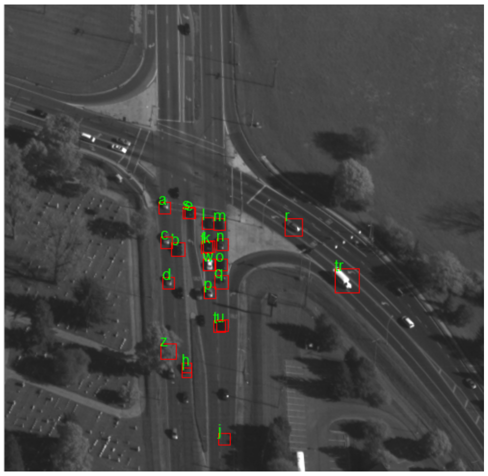
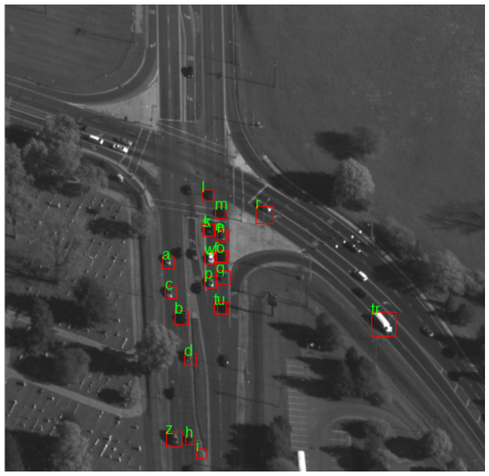


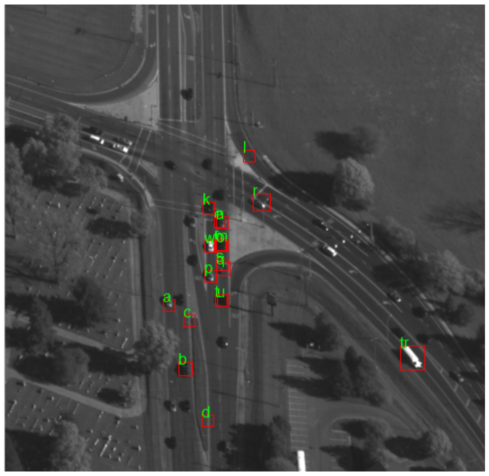
Figure 5.9: Error plot for target 3.


(a)


(b)


(c)


(d)


(e)

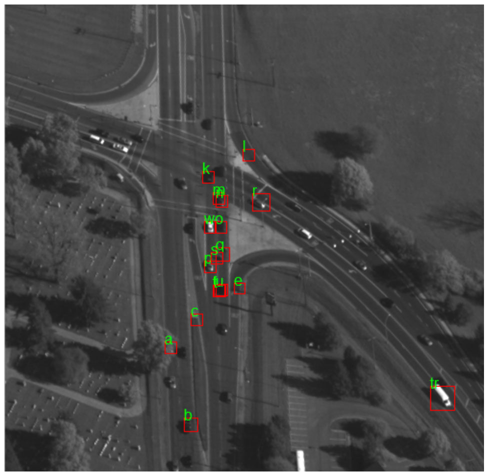

(f)

Figure 5.10: Tracking results for (a) frame 1, (b) frame 3, (c) frame 8, (d) frame 12, (e) frame 15 and (f) frame 18 using covariance tracker.

5.2 Object detection with rotation invariant gradient-based features

In this section, the performance analysis of the gradient-based object detection algorithm, the difference between circular and annular geometry of part division, analysis of rotation robustness, effects of approximating circular regions as square regions and the results on three datasets - Airplanes in Digital Aerial imagery and Backhoes and All-Terrain Vehicles (ATV) in low altitude aerial imagery are presented.

5.2.1 Performance analysis

Figure 5.11 shows how computation time is distributed among each step in the algorithm. A single integral formulation can be used for all parts, as the differentiation kernel and the number of angle bins are the same. As can be seen from the graph, non-maxima suppression and part assembly takes only an insignificant amount of time ($< 1\%$). The graph shown is for a Matlab implementation of aircraft detection using a two-part model. Each part is searched over 3 scales in an airport scene of size 1366×768. The amount of time spent in sliding window detection depends on the number of parts used and number of scales. The corresponding implementation without integral computation framework takes about 4.3 hours.

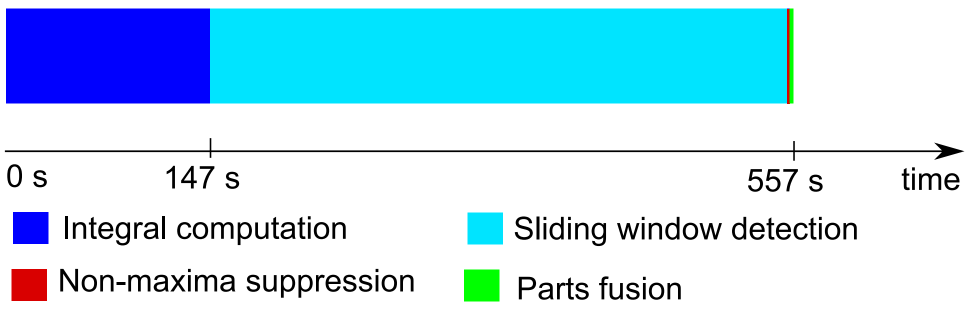


Figure 5.11: Distribution of computation time among various steps in object detection with rotation invariant gradient-based features.

The corresponding C++ implementation takes 8.85 s. Parallelization with OpenMP and Intel SSE gives approximately 3 times improvement in speed.

5.2.2 Circular region vs. annular region

An alternative geometry for region division is to use annular regions, as shown in Figure 5.12, instead of circular regions. Using annular regions gives cross-validation accuracies that are comparable to that given when circular regions are used. However, the number of computations involved in computing the feature in an annular region is higher.

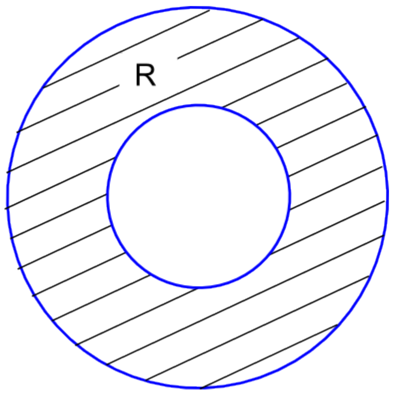


Figure 5.12: Annular region.

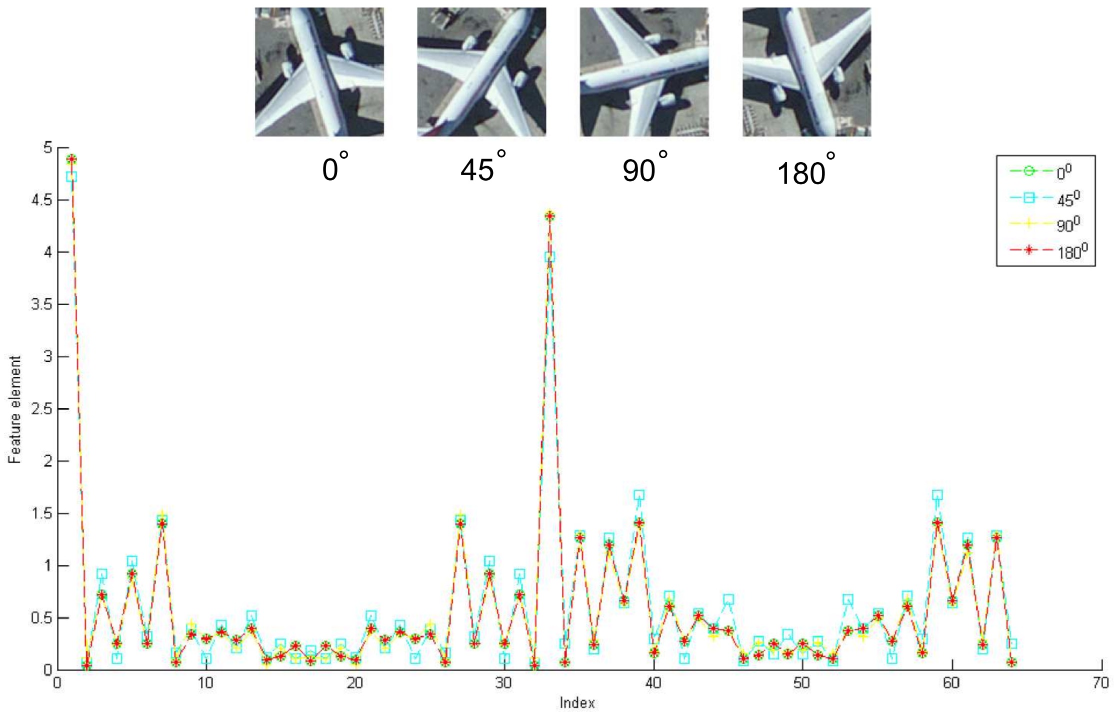
The histogram in an annular region R shown in the figure is computed as

$$h_R = h_{outer} - h_{inner} \tag{5.6}$$

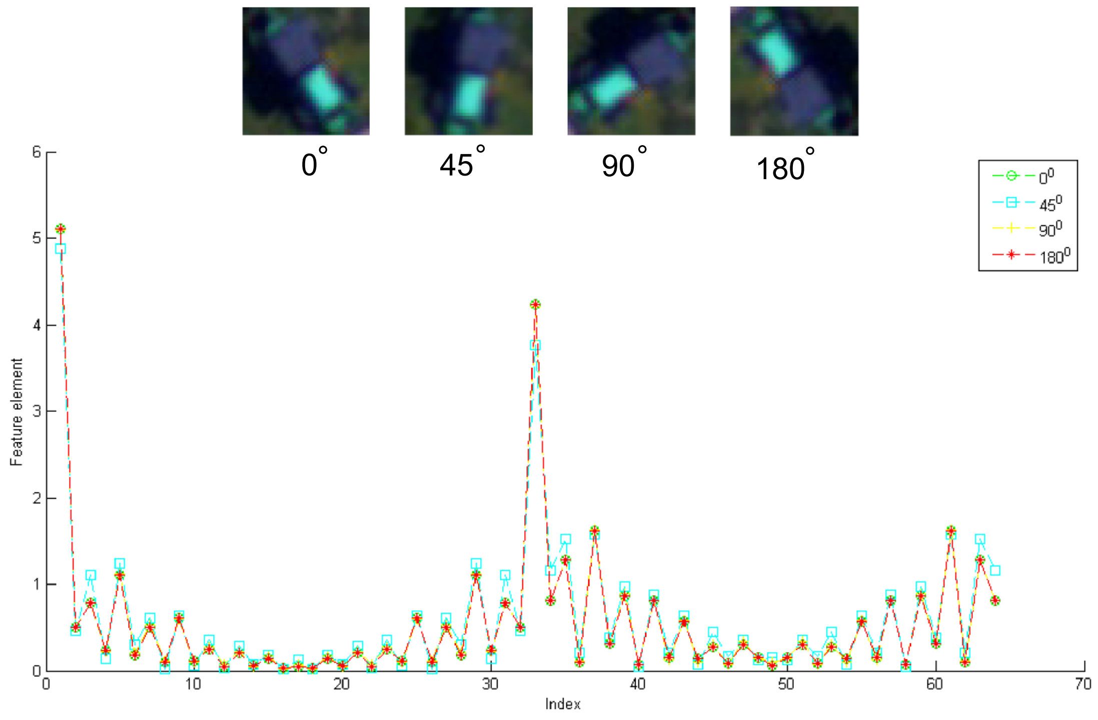
where h_{outer} is the histogram computed in the outer circle and h_{inner}, the histogram computed in the inner circle. If the number of summations required to compute the histogram in a circular region is $N_{circular}$, computing the histogram in an annular region requires $2N_{circular} + 1$ summations. Because of this added computational complexity, circular regions are used instead of annular regions.

5.2.3 Rotation robustness

Although the rotation invariance property of the descriptor was proven mathematically, there can be slight differences in feature vectors of rotated versions due to discretization effects of DFT. The effects of rotation of some object parts are shown in Figure 5.13. It can be seen that the feature does not vary much with rotations. The slight variations are accounted for by training the system with different rotations of the same sample.



(a)

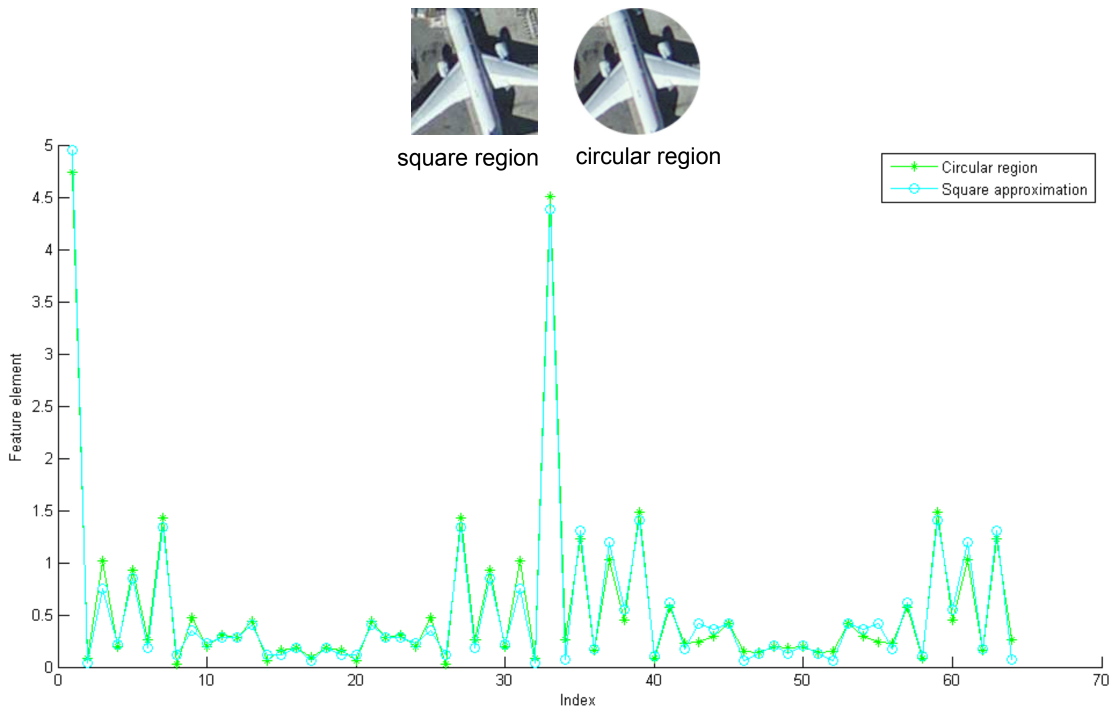


(b)

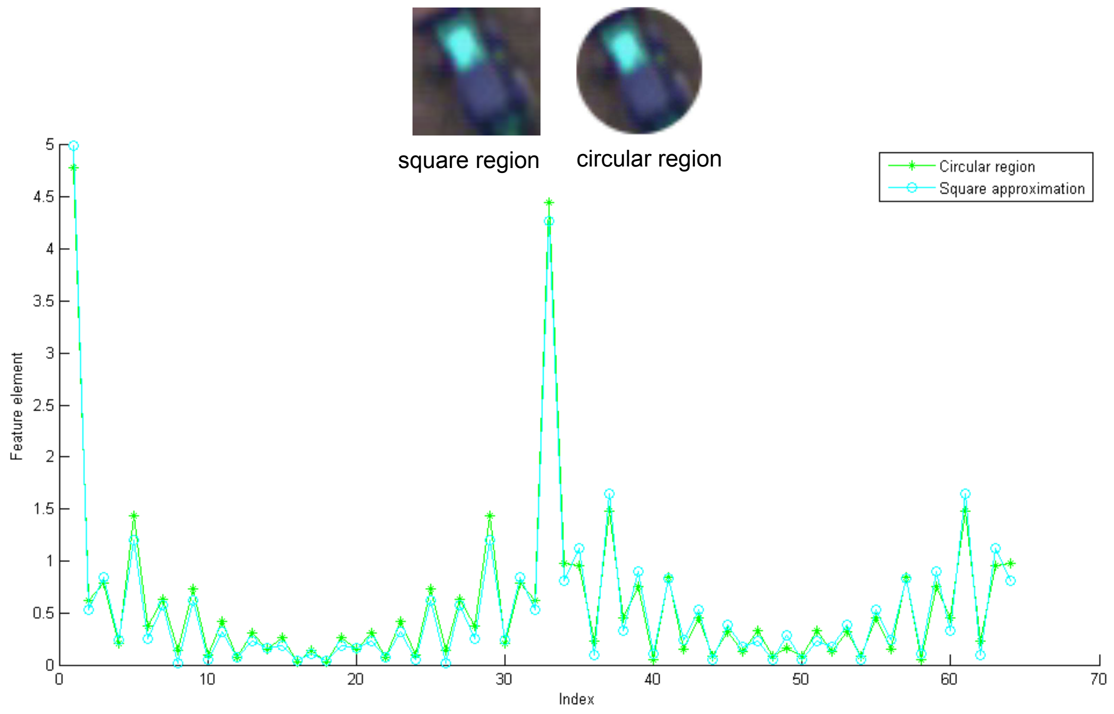
Figure 5.13: Rotation invariance of object parts, (a) Mid-section of aircraft and (b) One of the parts of backhoe.

5.2.4 Using square regions as approximation of circular regions

It was mentioned in Chapter III that a square region can be thought of as a highly discretized circle. As shown earlier, if the target is fully contained within the window, there is no significant difference. This is also shown experimentally for some representative object parts in Figure 5.14.



(a)



(b)

Figure 5.14: Comparison of features extracted from square region and circular region, (a) Mid-section of aircraft and (b) One of the parts of backhoe.

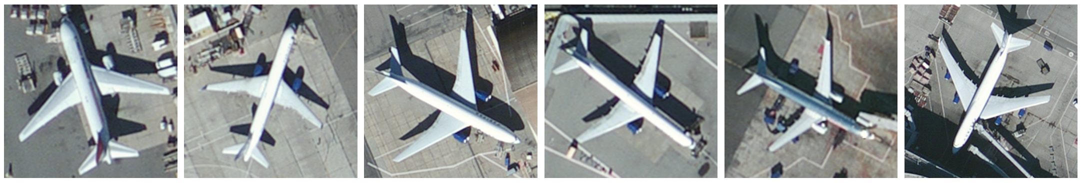


Figure 5.15: Representative images from Digital Aerial data.

5.2.5 Results on Digital Aerial dataset

Digital Aerial data is captured from cameras mounted on airplanes. The resolution of the data is 30 cm. Some examples of images from the dataset are shown in Figure 5.15. The method is used to detect airplanes in complex airport scenes. For training, as positive examples, we use 192 images obtained from 4 rotations of 48 airplane images collected from top-view of San Fransisco International airport. The number of negative examples used is 1513 and 1865 for the mid-section and wing respectively. The method is tested on 88 positive samples generated by rotating 22 airplanes images in 4 directions, and 2854 negative examples.

We select three visually salient parts - mid section of the fuselage and two wings, as shown in Figure 5.16, based on the key observation that they are least likely to be occluded. The appearance of nose of the airplane can change drastically when a dock is present. The empennage shows very high intra-class variability. Furthermore, the two wings of the airplane are reflections of each other, allowing us to use a single classifier for both wings. To build the histogram, 32 angle bins are used. The part with which the graph signature is normalized (4.49) is the mid-section. The negative samples to learn feature f_G in (4.49) are built by randomly selecting locations for parts in negative training samples. The positive samples are built from detections on the training set. For efficient testing, f_G is built with wing detections within a city block distance equal to thrice the size of mid-section detection. The size of our feature is only 64 (2 regions $\times$ 32-point DFT) for each part. For

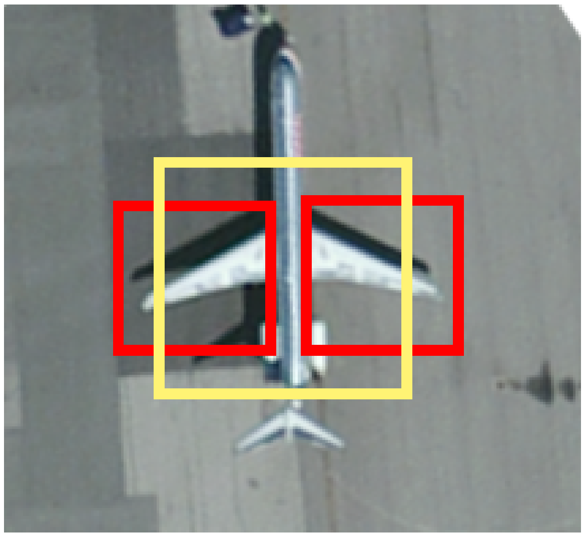


Figure 5.16: Selected parts of airplanes in Digital Aerial imagery.

gradient computation the kernel K_x = [-1 -2.6 0 2.6 1] is used. Number of regions, r, used to build feature $F(x, y)$ in (1.1) is 2 . The computational complexity increases as the number of regions is increased. To compute the histogram using integral histogram, the number of summations required when there are r regions is

$$N_s = cr \tag{5.7}$$

where c is the number of summations required to compute the histogram in one region. Square regions are used instead of circular regions. Since $c = 3$ for square regions, $N_s = 6$.

To define the relative size of the two regions, the parameter S is defined as the ratio of size of outer region to that of inner region. Since square regions are used,

$$S = \frac{side\ length\ of\ outer\ square}{side\ length\ of\ inner\ square} \tag{5.8}$$

In general, high accuracy on the training set does not guarantee high detection accuracy on unseen examples. High cross-validation accuracy over the training set translates to high accuracy on a test set. The values of S are therefore found by a grid search (parameter sweep) with 2 fold cross-validation over the training set.

Parameter selection and SVM parameter stability

Three parameters, algorithm parameter S in (5.8), C and γ (SVM parameters) need to be estimated. This can be done with a fine grid search over S, C and γ. However, a fine grid search is prohibitively expensive in terms of computation time. To solve this, the following strategy is used. S is varied in fine intervals and the highest cross-validation accuracies obtained by coarse variations of C and γ are recorded. The value of S that gives the highest cross-validation accuracy is selected. Once the value of S is obtained, C and γ are varied in fine intervals to determine their values based on the highest cross-validation accuracy.

Table 5.2 and 5.3 show the variation of cross-validation accuracies of mid-section and wing with S. It can be seen that for both parts, the cross-validation accuracies vary smoothly with S.

Table 5.2: Cross-validation accuracies and SVM parameters for mid-section.

S	Accuracy (%)	C	γ
1.2500	98.2714	4.0000	0.5000
1.3750	98.3995	4.0000	0.5000
1.5000	98.4635	8.0000	0.1250
1.6250	98.3355	4.0000	0.2500
1.7500	98.2074	4.0000	0.5000
1.8750	98.2074	4.0000	0.5000
2.0000	98.3355	8.0000	0.2500
2.1250	98.3355	8.0000	0.2500
2.2500	98.3355	4.0000	0.5000
2.3750	98.4635	8.0000	0.5000
2.5000	98.4635	4.0000	0.5000

Table 5.3: Cross-validation accuracies and SVM parameters for wing.

S	Accuracy (%)	C	γ
1.2500	93.2945	8.0000	0.5000
1.3750	93.4402	4.0000	1.0000
1.5000	93.2945	8.0000	0.5000
1.6250	93.1973	8.0000	1.0000
1.7500	93.1001	8.0000	1.0000
1.8750	93.3431	8.0000	0.5000
2.0000	93.2945	8.0000	1.0000
2.1250	93.4888	8.0000	1.0000
2.2500	93.4888	32.0000	0.5000
2.3750	93.2945	16.0000	1.0000
2.5000	93.3916	8.0000	0.5000

Based on the cross-validation accuracies, the choice of S for mid-section and wing are 2.3750 and 2.125 respectively. The variation of SVM parameters C and γ with these choices of S are shown in Figure 5.17 and Figure 5.18 for the mid-section and wing respectively. The cross-validation accuracy of mid-section is higher than that of wing because of its well-defined shape, symmetry and low likelihood of occlusion and background clutter. For the range of C and γ shown in Figure 5.17, the cross-validation accuracies of mid-section vary between 96.86% and 98.72%. Although the range of C and γ for wing is 81.55% to 96.47%, the values are distributed over a large range of C and γ, indicating high parameter stability.

The robustness of the method was also evaluated by adding 21 images of mid-section and 29 images of wings from Google Earth imagery. The highest cross-validation accuracies of mid-section and wings in a training set consisting of images from both Digital Aerial and Google Earth Imagery are 97.3% and 94.50% respectively. It may be noted that there are no drastic changes in cross-validation accuracies.

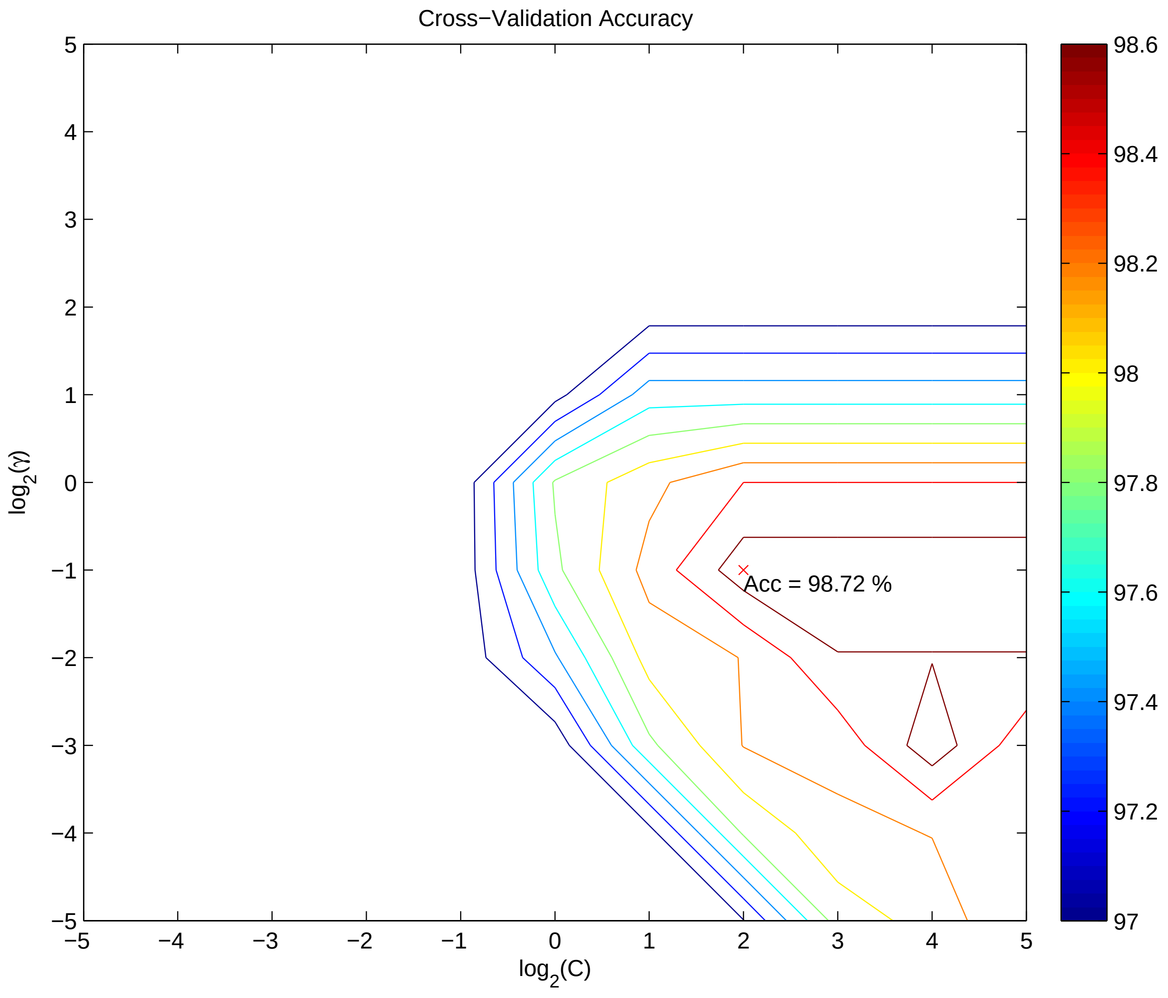


Figure 5.17: Variation of cross-validation accuracies with C and γ for the mid-section of aircraft.

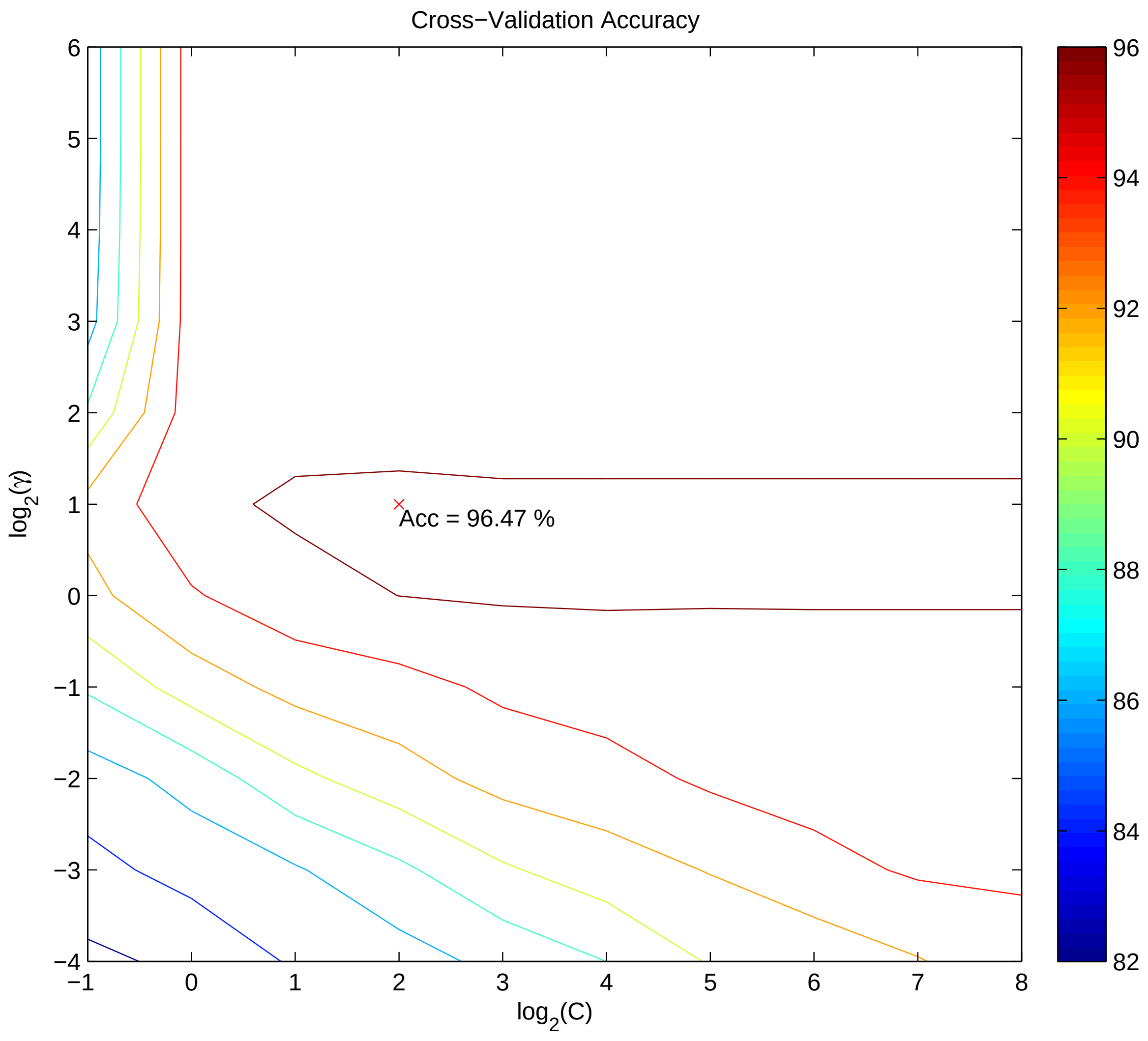


Figure 5.18: Variation of cross-validation accuracies with C and γ for wing.

For the SVM corresponding to the mid-section the values $C = 4$, $\gamma = 0.5$ are chosen and for wings, the values $C = 4$, $\gamma = 2$ are chosen.

Distribution of SVM scores

The SVM classifier outputs a confidence value (SVM score) along with the decision. The SVM gives a score above 0.5 for all positive examples and less than 0.5 for negative examples. In some cases, it is desirable to fix a value other than 0.5 as the threshold separating positive and negative examples. For example, in detecting cancerous tumors, a value much lower than 0.5 may be fixed as the threshold since missed detections can be catastrophic. The distribution of data points for mid-section and wing are shown in Figure 5.19 and 5.20 respectively. In both cases, it can be seen that negative and positive examples are well separated, allowing a fixed threshold of 0.5.

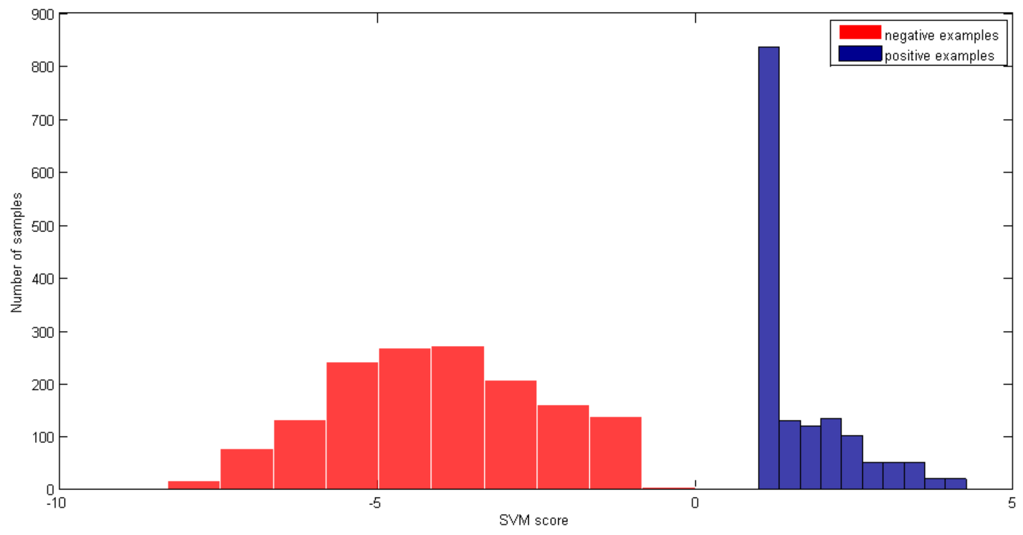


Figure 5.19: Distribution of SVM decision scores for mid-section of aircraft.

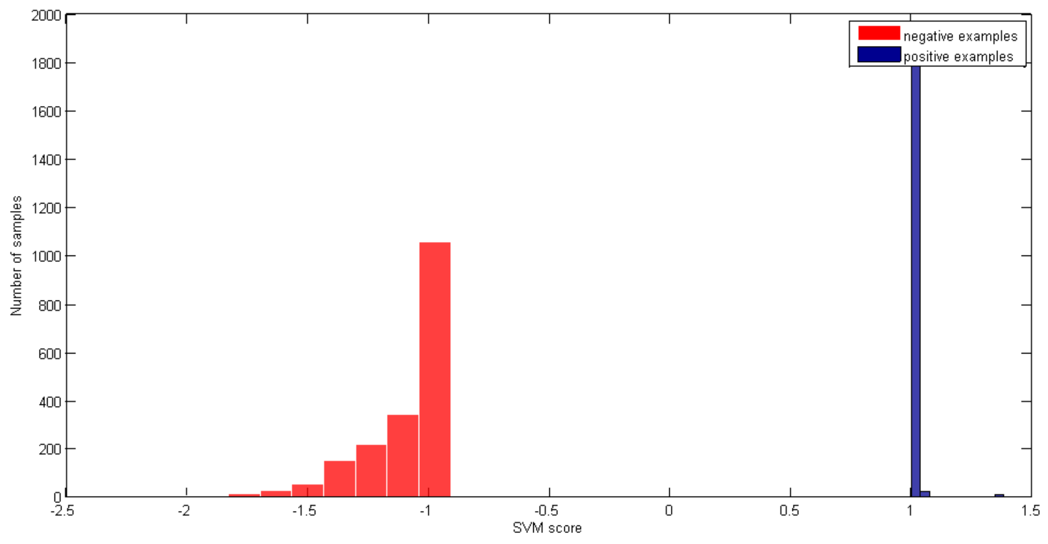


Figure 5.20: Distribution of SVM decision scores for aircraft wing.

Detection results

Figure 5.21 shows detecting parts of an aircraft. All detections of wings are shown in Figure 5.21(a). Figure 5.21(b) shows the detections after non-maxima suppression. Figure 5.21(c) shows detections of the mid-section of the aircraft. Detections after non-maxima suppression are shown in Figure 5.21(d).

Figure 5.22 shows the detections of aircrafts in an airport scene. Minor shifts in detections are due to strong self-shadows.

Comparison

Performance of detection algorithms can be quantified using two measures - precision and recall. Precision is computed as

$$precision = \frac{tp}{tp + fp} \tag{5.9}$$

where tp is the number of true positives and fp, the number of false positives.

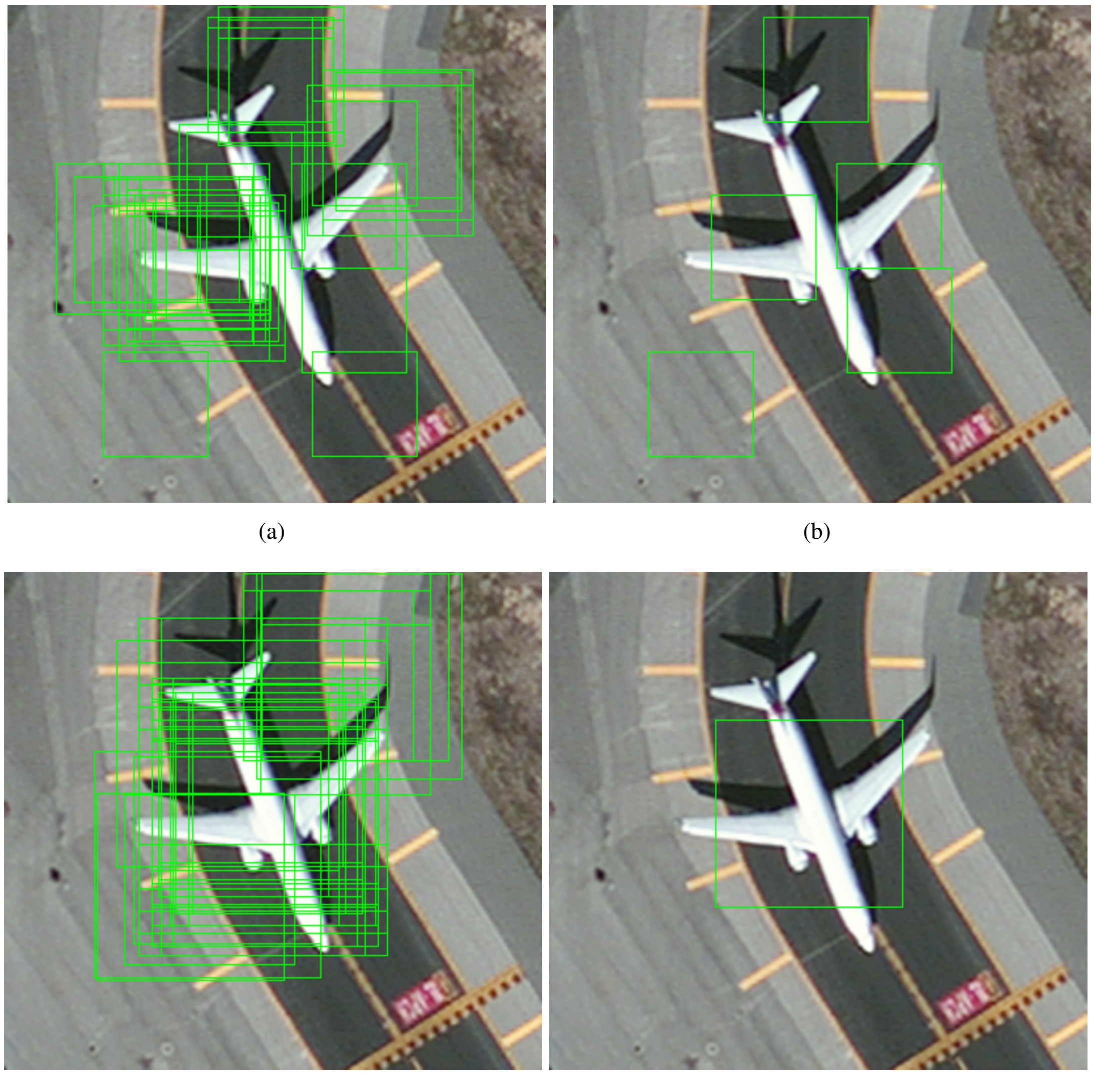

(a)
(b)
(c)
(d)

Figure 5.21: Detecting parts of an aircraft, (a) Wing detections, (b) Wing detections after non-maxima suppression, (c) Mid-section detections and (d) Mid-section detections after non-maxima suppression.

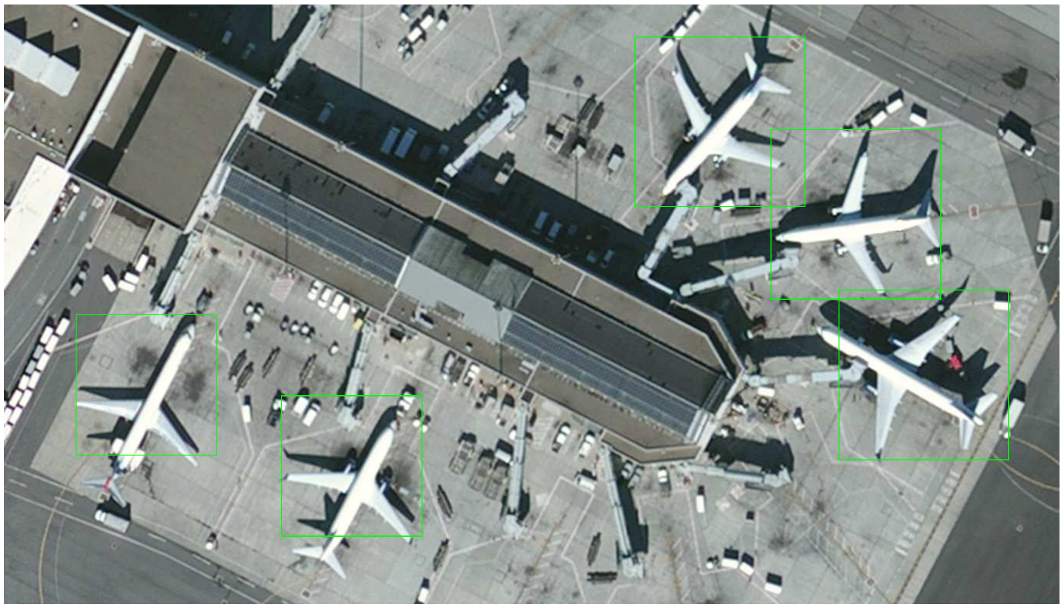


Figure 5.22: Examples of detections in a complex airport scene.

Recall is computed as

$$recall = \frac{tp}{tp + fn} \tag{5.10}$$

where fn is the number of false negatives.

The method is compared with a part-based detection scheme presented in [60]. Figure 5.23 shows the precision-recall characteristics of the two methods. The graph is plotted by varying the SVM decision threshold.

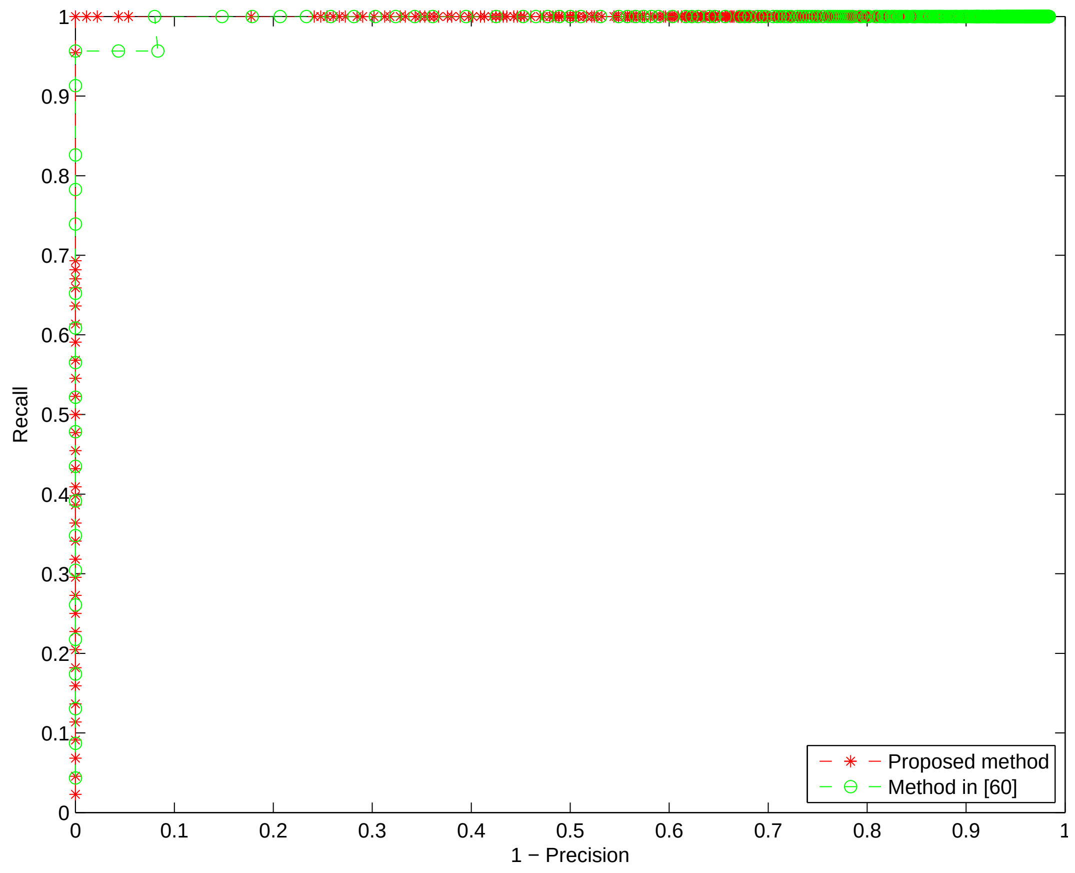


Figure 5.23: Precision-recall curve showing a comparison of the proposed method with that in [60].

HOG performs at a level comparable to ours in terms of accuracy, but is several times slower than our method. We emphasize that our method easily beats state-of-the-art methods in speed by several orders of magnitude, while maintaining a comparable level of accuracy. The performance comparison of our method, in terms of speed, true positive rate (TPR) and false positive rate (FPR) is summarized in Table 5.4. Rotation invariance with HOG is realized by using 24 SVMs corresponding to 24 orientations of the target. The execution time in Table 5.4 is the time a Matlab implementation takes to run on a 1366 $\times$ 768 image on a 2.53GHz PC with 4GB RAM for a single scale.

Table 5.4: Performance comparison of the proposed method with HOG and [60].

Method	Execution time	TPR	FPR
Proposed method	**195.3 s**	95.45%	0.013
Method in [60]	3.38 hr	90.9%	0.016
HOG (24 orientations)	2.54 hr	95.45%	0.011

There are two ways to speed up computation of the feauture. The first is to use L_1 norm when normalizing the histogram for illumination invariance. This is equivalent to using the DC component $H_i(x, y, 0)$ instead of $\|H_i(x, y, :)\|_2$ in (4.18). The second is to square all elements of the vector $\widetilde{f}_i(x, y)$ in (4.18). This avoids square root computations. Although there is some improvement in speed, there is a decrease in accuracy. The results with these approaches are shown in Table 5.5

Table 5.5: Performance of the algorithm when L_1 norm or square of feature elements is used.

Method	Execution time	TPR	FPR
Squared feature elements	162.04 s	90.9%	0.043
L_1 norm	186.3 s	90.9%	0.032

5.2.6 Results on backhoe detection in low-altitude imagery

This imagery is also captured from aircrafts. The images are captured by different vendors at different times of the day or on different days. This results in photometric variations and changes in viewpoint. Imagery from different vendors appear in different colors and levels of blur because of difference in camera sensor. Unlike airplanes, this dataset is extremely challenging mainly because of poorly defined contours of the target images. Since this imagery is data-limited, a slightly different testing methodology is followed. For training, 10 positive examples are used. 30 other positive examples are generated by rotating in angles 90°, 180° and 270°, yielding a total of 40 images. Examples of positive and negative samples are shown in Figure 5.24.

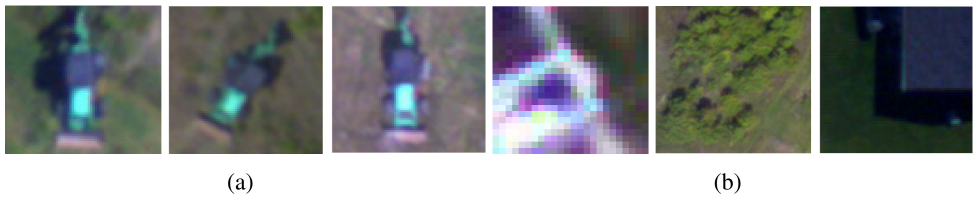


Figure 5.24: Examples of images used in training the backhoe detection system, (a) Examples of positive samples and (b) Examples of negative samples.

Backhoes are represented using a three part model as shown in Figure 5.25. The parts are selected based on the fact that these are the most visually salient rigid regions. Part 1, marked in yellow, is roughly square in geometry and part 2, marked in green, is roughly rectangular. Part 3 is formed by the combination of part 1 and part 2. The number of negative examples used for part 1, part 2 and part 3 is 997, 957 and 900 respectively. The part with which the graph signature is normalized (4.49) is part 3. The same strategy followed in airplane detection is used to build the training set for training the SVM that learns part geometry. For efficient testing, f_G is built with detections of part 1 and part 2 within a city block distance equal to the size of detection of part 3.

The algorithm is tested on two scenes containing target objects. One of the scenes was captured using a sensor that is different from that used in capturing the training images.

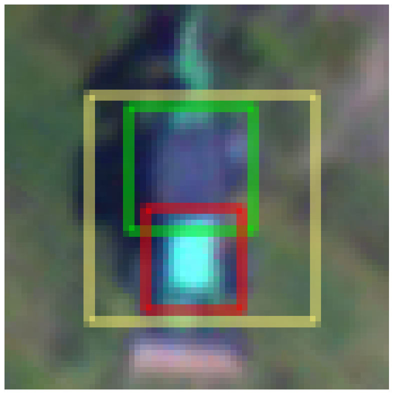


Figure 5.25: Backhoe represented as three parts.

The following setup is used -

The kernel K_x = [-1 -2.7 0 2.7 1] is used to compute the gradients. The number of angle bins used for all parts is 32.

Parameter selection and SVM parameter stability

The same strategy as that used for airplane detection is followed for selecting the algorithm parameter S and the SVM parameters C and γ. Table 5.6, 5.7 and 5.8 show the variation of accuracy with S.

Table 5.7: Cross-validation accuracies and SVM parameters for part 2.

S	accuracy (%)	C	γ
1.2500	98.3003	2.0000	1.0000
1.3750	98.5836	2.0000	1.0000
1.5000	98.3003	2.0000	1.0000
1.6250	98.8669	8.0000	0.5000
1.7500	98.3003	4.0000	0.5000
1.8750	97.7337	2.0000	1.0000
2.0000	97.7337	2.0000	2.0000
2.1250	97.7337	8.0000	0.5000
2.2500	98.0170	8.0000	0.5000
2.3750	97.4504	4.0000	0.5000
2.5000	97.4504	2.0000	0.5000

Table 5.6: Cross-validation accuracies and SVM parameters for part 1.

S	Accuracy (%)	C	γ
1.2500	95.6522	2.0000	1.0000
1.3750	95.6522	2.0000	0.5000
1.5000	96.7391	4.0000	0.5000
1.6250	96.7391	4.0000	0.5000
1.7500	98.9130	4.0000	0.1250
1.8750	98.9130	4.0000	0.1250
2.0000	97.8261	2.0000	0.2500
2.1250	98.9130	4.0000	0.0625
2.2500	97.8261	2.0000	0.1250
2.3750	97.8261	4.0000	0.0625
2.5000	96.7391	4.0000	0.0625

Table 5.8: Cross-validation accuracies and SVM parameters for part 3.

S	Accuracy (%)	C	γ
1.2500	98.8372	0.5000	1.0000
1.3750	98.8372	1.0000	0.5000
1.5000	98.8372	0.5000	1.0000
1.6250	100.0000	2.0000	0.2500
1.7500	100.0000	8.0000	0.1250
1.8750	98.8372	0.5000	0.5000
2.0000	98.8372	0.5000	0.5000
2.1250	98.8372	0.5000	1.0000
2.2500	98.8372	1.0000	0.2500
2.3750	98.8372	1.0000	0.2500
2.5000	98.8372	1.0000	0.2500

Based on the cross-validation accuracies, the values of S used for part 1, part 2 and part 3 are 2.125, 1.625 and 1.6250 respectively. For the selected values of S, the variation of cross-validation accuracies for part 1, part 2 and part 3 are shown in Figure 5.26, Figure 5.27 and Figure 5.28 respectively. As can be seen from the figures, there is a large range of C and γ over which cross-validation accuracies are high. It may also be noted that there are no abrupt jumps in cross-validation accuracies.

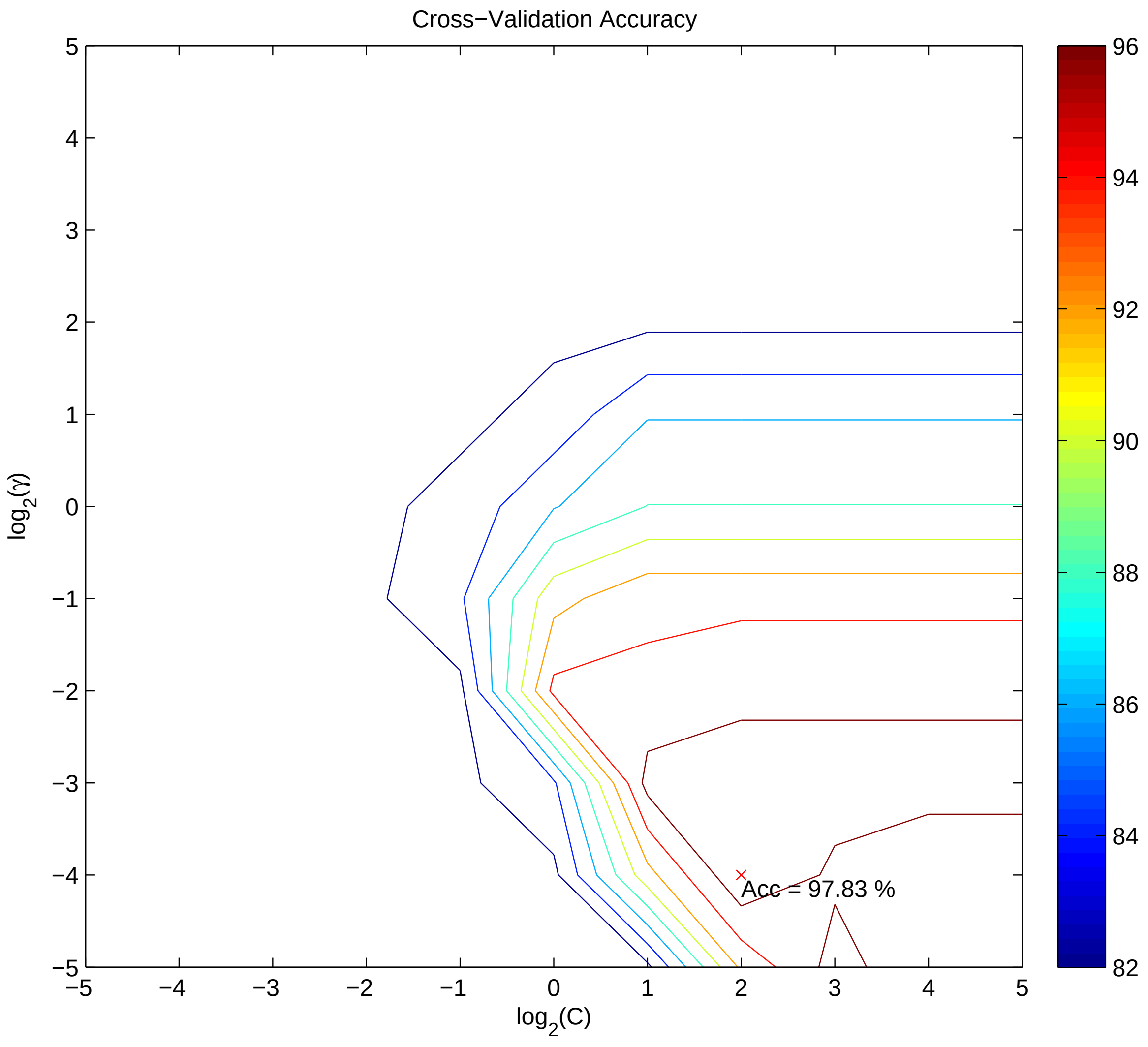


Figure 5.26: Variation of cross-validation accuracies with C and γ for part 1 of backhoe.

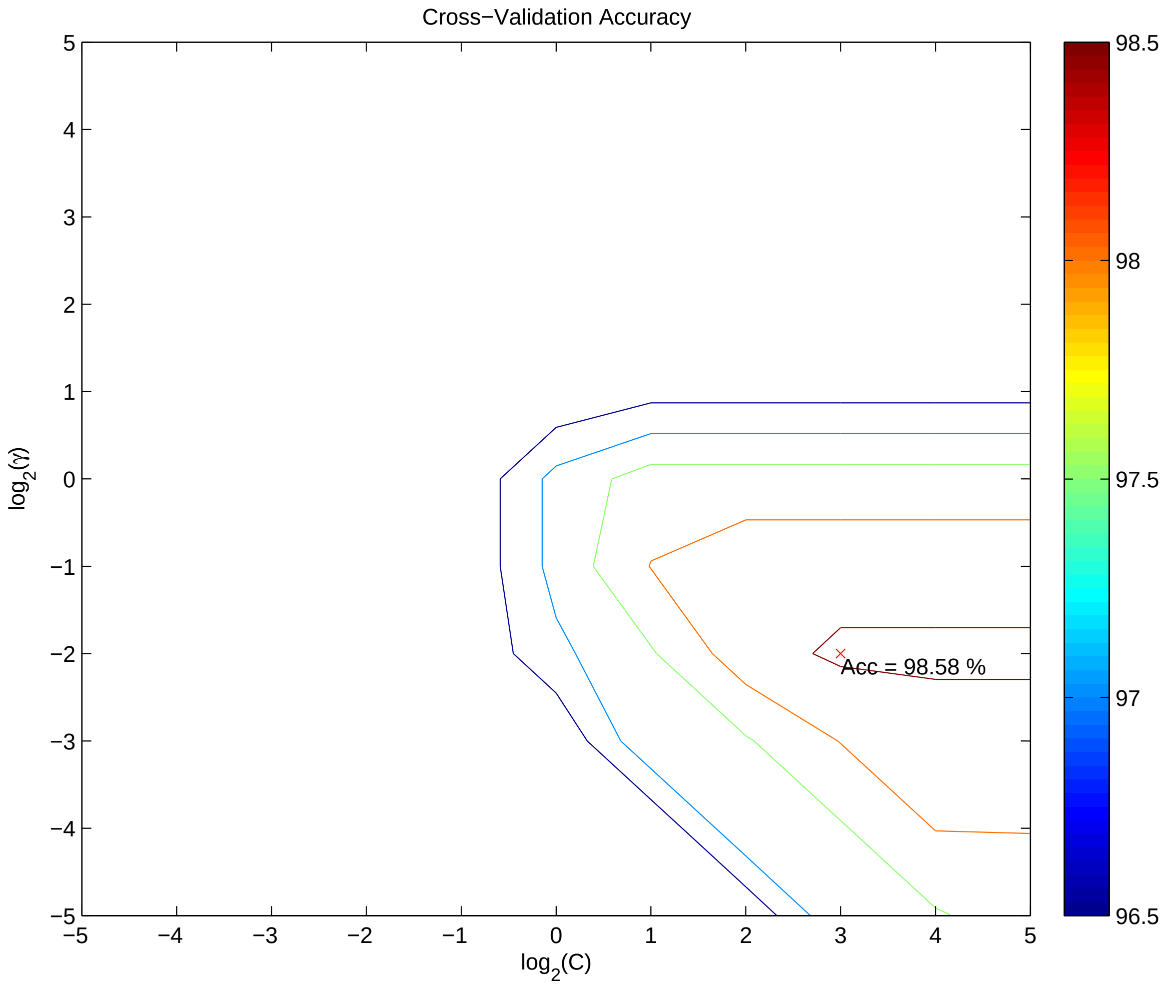


Figure 5.27: Variation of cross-validation accuracies with C and γ for part 2 of backhoe.

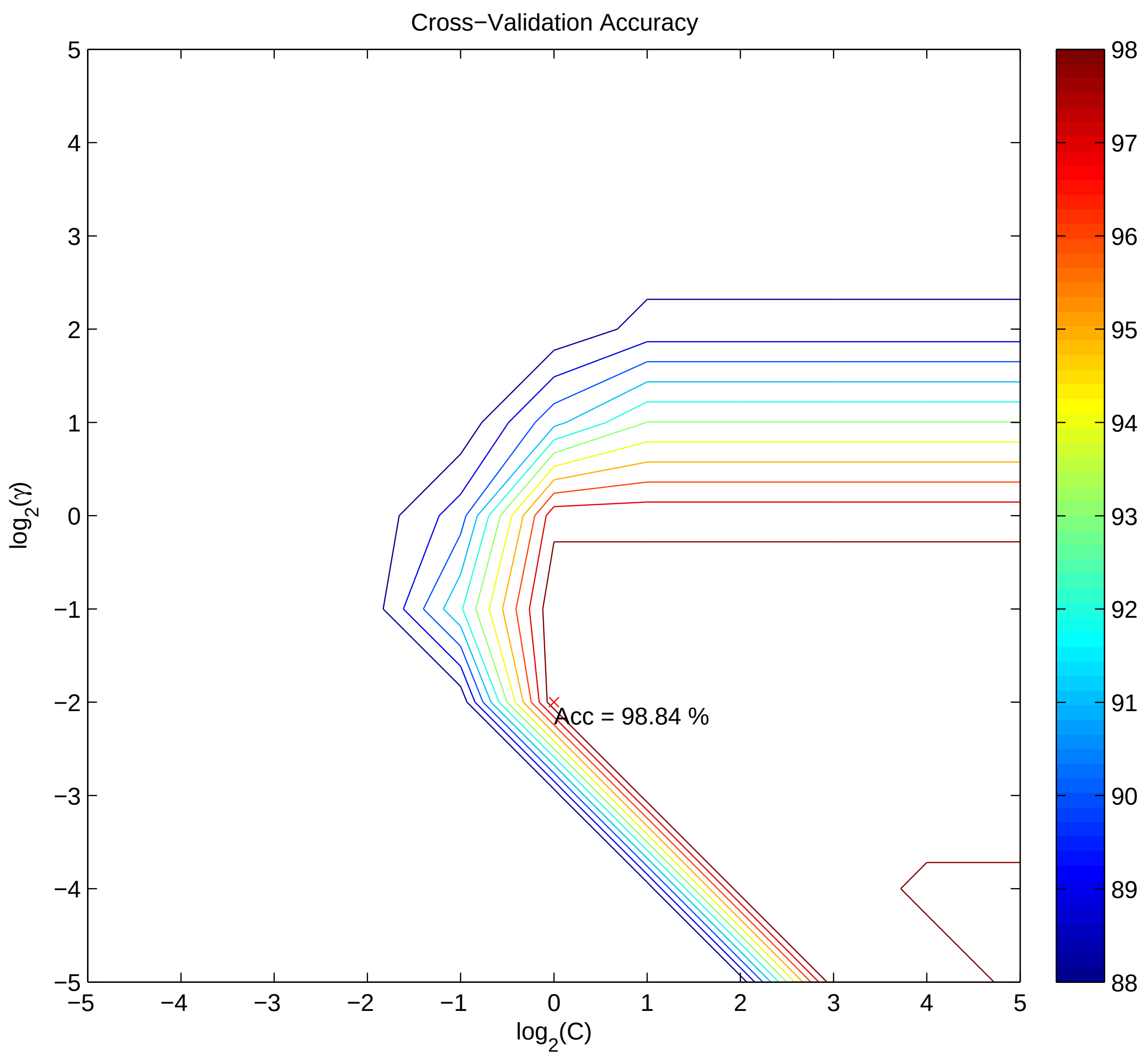


Figure 5.28: Variation of cross-validation accuracies with C and γ for part 3 of backhoe.

The following choices are made for SVM parameters - for part 1, $C = 4$, $\gamma = 0.0625$, for part 2, $C = 8$, $\gamma = 0.25$ and for part 3, $C = 1$, $\gamma = 0.25$.

Distribution of SVM scores

The distribution of SVM scores for part 1, part 2 and part 3 are shown in Figure 5.29, 5.30 and 5.31 respectively.

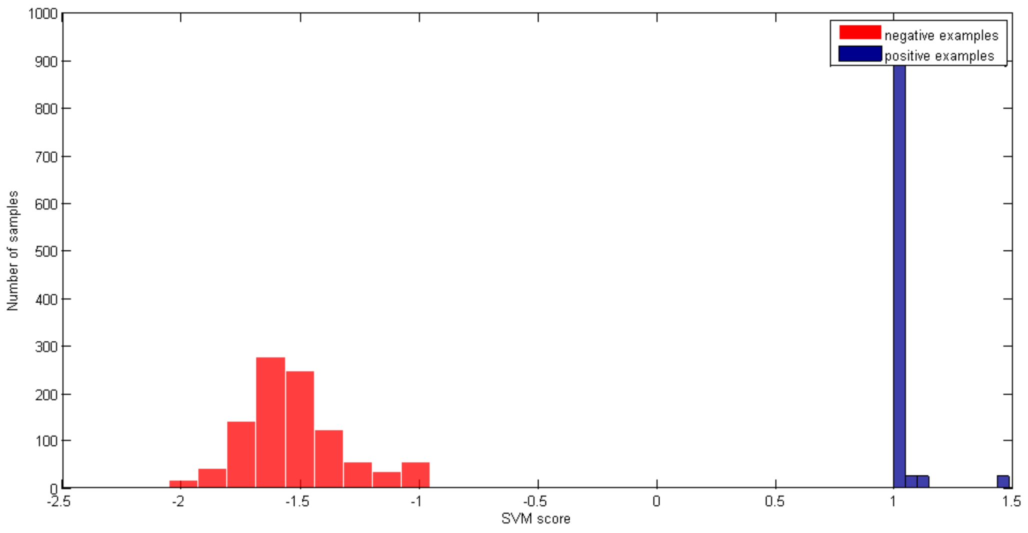


Figure 5.29: Distribution of SVM decision scores for part 1.

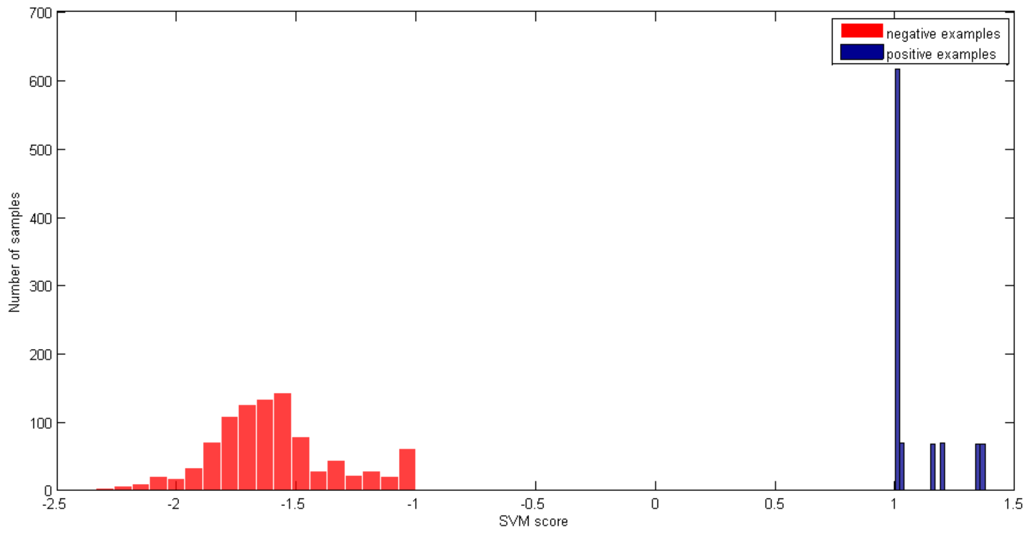


Figure 5.30: Distribution of SVM decision scores for part 2.

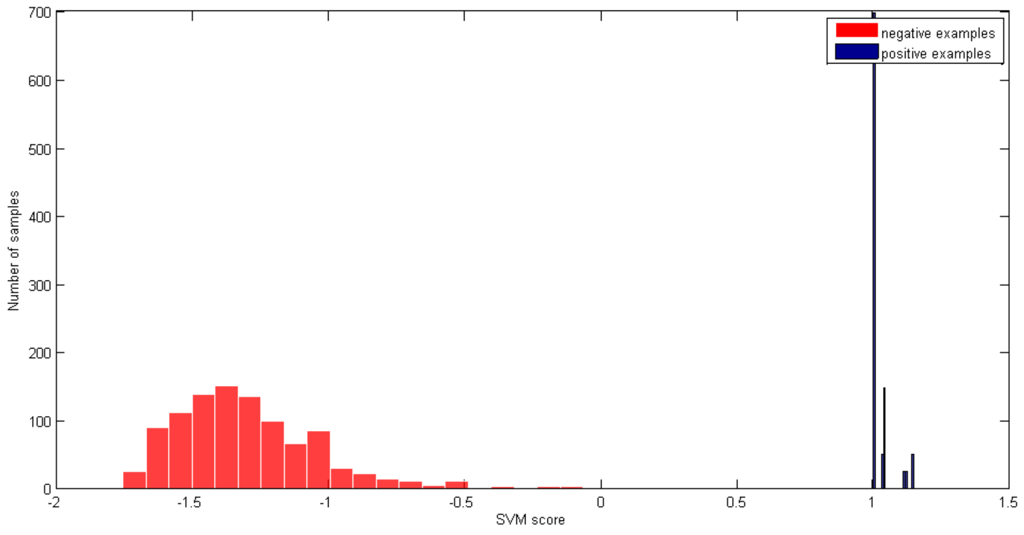


Figure 5.31: Distribution of SVM decision scores for part 3.

It can be seen that the scores are well separated for all the selected parts.

Detection results and comparison

Figure 5.32 shows the detection of a backhoe using the proposed algorithm.

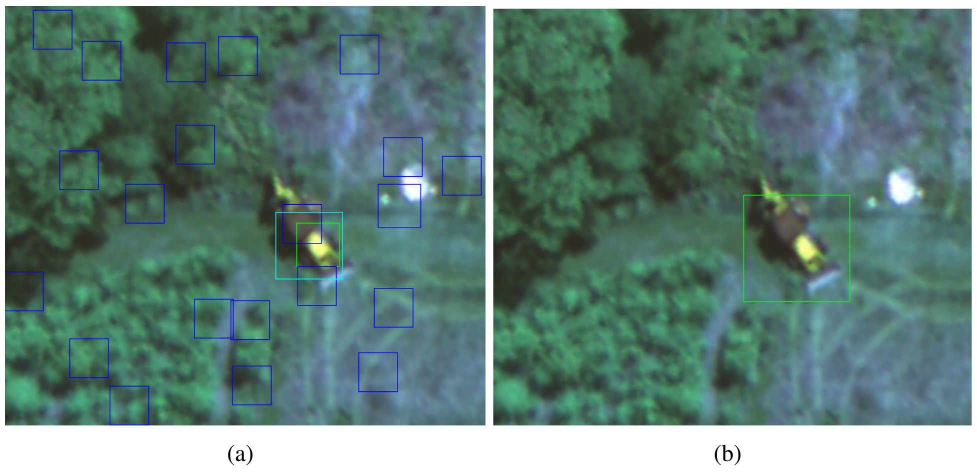


Figure 5.32: Backhoe detection, (a) Part detections and (b) Detection after part assembly.

Since the dataset is limited in terms of number of data, precision-recall curve is not a reliable measure of performance. On the tested scenes, all methods perform equally well in terms of accuracy. Rotation invariance with HOG is realized by using 24 SVMs corresponding to 24 orientations of the target. However, the proposed method is several times faster than HOG and [60]. Table 5.9 shows a comparison of execution times for an image of size 933×813.

Table 5.9: Performance comparison of the proposed method with HOG and [60].

Method	Execution time
Proposed method	**67.7 s**
Method in [60]	1.62 hr
HOG (24 orientations)	35.93 min

5.2.7 Results on All-Terrain Vehicle detection in low-altitude imagery

This imagery is also captured using cameras mounted on aircrafts. The training and the test set differ in the appearance of target. Target appearance is affected by background and level of blur. For training, as positive examples, we use 28 images obtained from 4 rotations of 7 All-Terrain Vehicle (ATV) images. As negative examples, 1487 patches of background (non-ATV images) are used. The kernel K_x = [-1 -2.7 0 2.7 1] is used to compute the gradients. The number of angle bins used is 32. As ATVs do not have distinguishable parts at the available resolution, a single-part model is used. Some examples of positive and negative examples are shown in Figure 5.33.

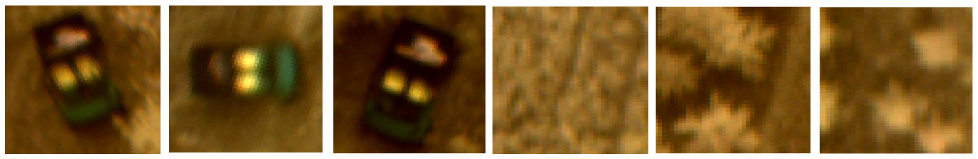

(a) (b)

Figure 5.33: Examples of images used in training the backhoe detection system, (a) Examples of positive samples and (b) Examples of negative samples.

Parameter selection and SVM parameter stability

The variation of cross-validation accuracies and SVM parameters for backhoe is shown in Table 5.10.

Table 5.10: Cross-validation accuracies and SVM parameters for ATV.

S	Accuracy (%)	C	γ
1.2500	98.4410	4.0000	0.2500
1.3750	98.6637	8.0000	0.2500
1.5000	98.8864	4.0000	0.5000
1.6250	99.3318	16.0000	0.1250
1.7500	98.6637	4.0000	0.2500
1.8750	98.2183	4.0000	0.5000
2.0000	98.4410	4.0000	0.5000
2.1250	98.6637	4.0000	0.5000
2.2500	98.6637	4.0000	0.5000
2.3750	98.4410	4.0000	0.5000
2.5000	98.6637	8.0000	0.2500

S = 1.5 is chosen based on the highest cross-validation accuracy in Table 5.10. Figure 5.34 shows the variation of C and γ for the selected value of S. As can be seen, there is a large range of C and γ for which cross-validation accuracy is greater than 90%.

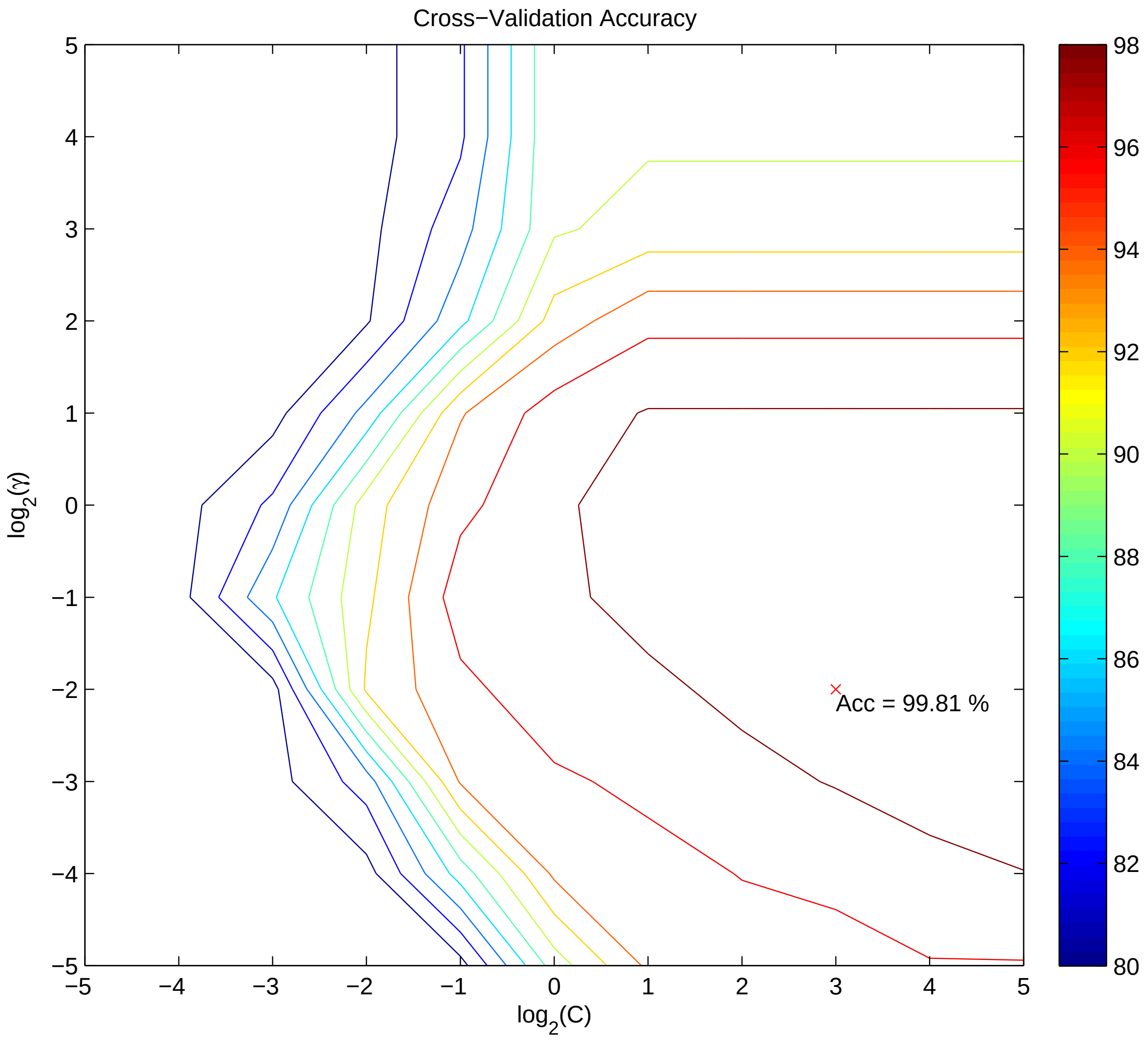


Figure 5.34: Variation of cross-validation accuracies with C and γ for ATV.

Based on the highest cross-validation accuracy, C = 8, γ = 0.25 are chosen as SVM parameters.

Distribution of SVM scores

The distribution of SVM scores for ATV, showing the separability of positive and negative examples, is shown in Figure 5.35.

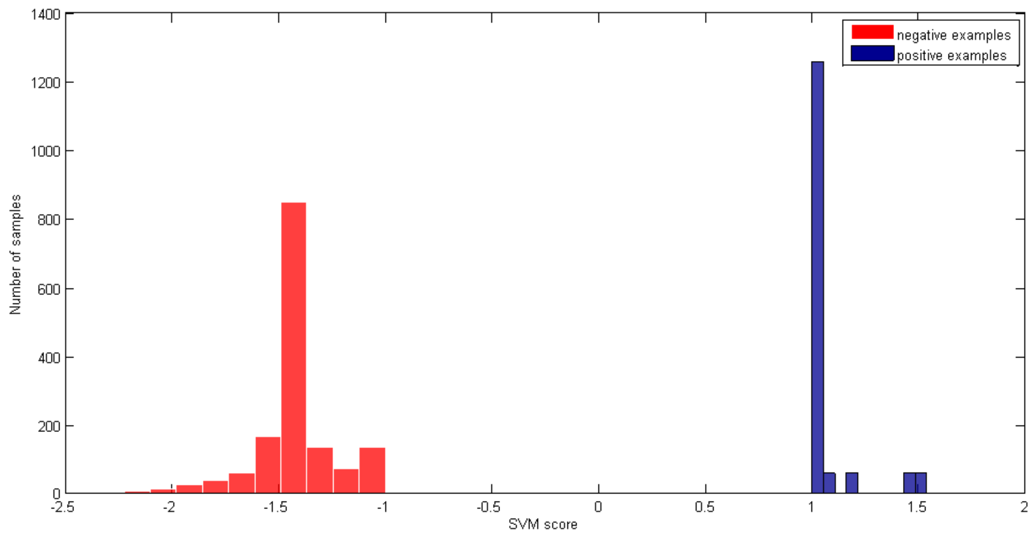


Figure 5.35: Distribution of SVM decision scores for ATV.

Detection results and comparison

Figure 5.36 shows some examples of ATV detection with the proposed method.

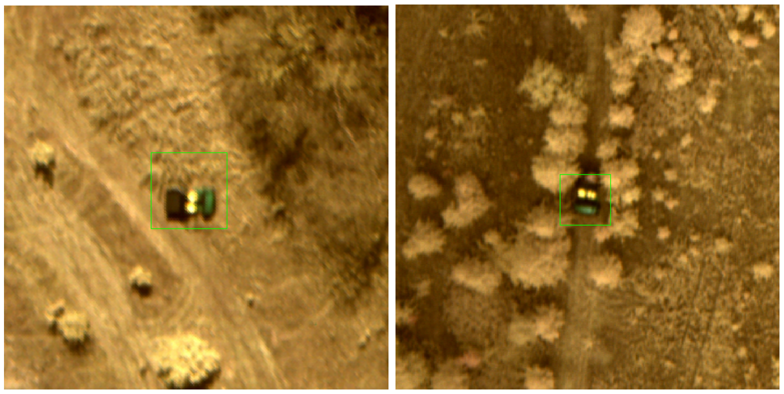


Figure 5.36: ATV detections in two test scenes.

All methods detect the target on the tested scenes without false positives. However, the proposed method is several times faster than HOG and [60]. Rotation invariance with HOG is realized by using 24 SVMs corresponding to 24 orientations of the target. Table 5.11 shows a comparison of execution times for an image of size 500 $\times$ 500.

Table 5.11: Performance comparison of the proposed method with HOG and [60].

Method	Execution time
Proposed method	**16.2 s**
Method in [60]	17.2 min
HOG (24 orientations)	11.09 min

5.3 Summary

The experimental results supporting the proposed algorithm were presented in this chapter. Tracking with rotation invariant intensity-based features was tested on tracking vehicles in WAMI data. The proposed method was shown to be superior to two other state-of-the-art methods. The results on object detection with gradient-based rotation invariant features were presented. The proposed method was compared with two other methods. The method was shown to be several times faster, while maintaining comparable accuracy. Several optimization techniques were used to speed up the algorithm. The analysis substantiating these method were also presented.

CHAPTER VI

CONCLUSION AND FUTURE WORK

In this dissertation, a novel method to detect and track objects in aerial imagery was presented. The method exploits the surface characteristics of concentric regions centered at visually salient locations of the object for feature representation. The surface characteristics are represented either using concatenated intensity histograms or spectral magnitude of gradient histograms. The division into concentric regions serves two purposes, 1) the isometric property of circle allows for rotation invariance and 2) encoding spatial layout of the object increases expressiveness of the descriptor. It was shown that circle can be approximated as squares without loss of accuracy. Intensity-based rotation invariant features were used in tracking applications where gradient information is not available. Gradient-based features were used for detecting objects in aerial imagery. Intensity-based features are inherently rotation invariant, since rotation of an image patch does not cause its intensity histogram to change. However, this principle does not hold for gradient orientation histograms. To achieve rotation invariance of gradient orientation histograms, Fourier representation was used. Illumination robustness was achieved though normalizing spectral coefficients. The sheer size of aerial imagery demands an efficient way to compute features. To this end, a very fast method to compute the gradient-based feature in a sliding window detector was also presented. This was achieved through employing a representation called Integral DFT. It was shown that extracting a single rotation invariant feature from the object as a whole might not be enough to model an object.

To address this issue, a rule-based object detection strategy using features of various parts of an object was developed.

The results on object tracking with intensity-based features was shown on WAMI data. It was demonstrated that point detectors failed to capture object information effectively. The superiority of the proposed method was established by comparing it with two other popular methods. The results of object detection with gradient-based methods was shown on three aerial datasets - airplanes in Digital Aerial imagery, backhoes in low-altitude aerial imagery and All-Terrain Vehicles (ATV) in low-altitude aerial imagery. The parameters for the method were found using cross-validation. The method was shown to be several times faster than two state-of-the-art methods.

In the proposed part based approach, visually salient locations are selected manually. As future work, it will be interesting to adopt an automatic part selection strategy. This will obviate the need to annotate and record part positions, thus reducing data preparation time significantly. Since the method is closely related to point features like SIFT and SURF, it is possible to use the gradient-based feature in applications like scene-matching and image stitching by matching similar points. This can be done by combining the method with an efficient interest point detector such as FAST, and computing the gradient-based feature in a small neighborhood around it.

BIBLIOGRAPHY

[1] T. Tuytelaars and K. Mikolajczyk, "Local invariant feature detectors: a survey," *Foundations and Trends® in Computer Graphics and Vision*, vol. 3, no. 3, pp. 177–280, 2008.

[2] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[3] ——, "Object recognition from local scale-invariant features," in *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, vol. 2. IEEE, 1999, pp. 1150–1157.

[4] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 1. IEEE, 2005, pp. 886–893.

[5] J. Birk, R. Kelley, N. Chen, and L. Wilson, "Image feature extraction using diameter-limited gradient direction histograms," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. PAMI-1, no. 2, pp. 228–235, April 1979.

[6] I. Biederman and G. Ju, "Surface versus edge-based determinants of visual recognition," *Cognitive psychology*, vol. 20, no. 1, pp. 38–64, 1988.

[7] T. Sanocki, K. W. Bowyer, M. D. Heath, and S. Sarkar, "Are edges sufficient for object recognition?" *Journal of Experimental Psychology: Human Perception and Performance*, vol. 24, no. 1, p. 340, 1998.

[8] M. J. Swain and D. H. Ballard, "Color indexing," *International journal of computer vision*, vol. 7, no. 1, pp. 11–32, 1991.

[9] F. Porikli, "Integral histogram: A fast way to extract histograms in cartesian spaces," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 1. IEEE, 2005, pp. 829–836.

[10] L. Bretzner and T. Lindeberg, "Feature tracking with automatic selection of spatial scales," *Computer Vision and Image Understanding*, vol. 71, no. 3, pp. 385–392, 1998.

[11] C. Harris and M. Stephens, “A combined corner and edge detector,” in *In Proc. of Fourth Alvey Vision Conference*, 1988, pp. 147–151.

[12] J. Shi and C. Tomasi, “Good features to track,” in *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR ’94., 1994 IEEE Computer Society Conference on*, Jun 1994, pp. 593–600.

[13] S. M. Smith and J. M. Brady, “Susana new approach to low level image processing,” *International journal of computer vision*, vol. 23, no. 1, pp. 45–78, 1997.

[14] E. Rosten, R. Porter, and T. Drummond, “Faster and better: A machine learning approach to corner detection,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 32, no. 1, pp. 105–119, Jan 2010.

[15] J. R. Quinlan, “Induction of decision trees,” *Machine learning*, vol. 1, no. 1, pp. 81–106, 1986.

[16] E. Mair, G. D. Hager, D. Burschka, M. Suppa, and G. Hirzinger, “Adaptive and generic corner detection based on the accelerated segment test,” in *Computer Vision–ECCV 2010*. Springer, 2010, pp. 183–196.

[17] M. Brown and D. Lowe, “Invariant features from interest point groups,” in *Proc. BMVC*, 2002, pp. 23.1–23.10, doi:10.5244/C.16.23.

[18] Y. Ke and R. Sukthankar, “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*, vol. 2. IEEE, pp. II–506.

[19] M. Grabner, H. Grabner, and H. Bischof, “Fast approximated sift,” in *Computer Vision–ACCV 2006*. Springer, 2006, pp. 918–927.

[20] J. van de Weijer, T. Gevers, and A. Bagdanov, “Boosting color saliency in image feature detection,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 28, no. 1, pp. 150–156, Jan 2006.

[21] A. Bosch, A. Zisserman, and X. Muoz, “Scene classification using a hybrid generative/discriminative approach,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 30, no. 4, pp. 712–727, April 2008.

[22] K. Mikolajczyk and C. Schmid, “A performance evaluation of local descriptors,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 27, no. 10, pp. 1615–1630, Oct 2005.

[23] F. Bellavia, D. Tegolo, and E. Trucco, “Improving sift-based descriptors stability to rotations,” in *Pattern Recognition (ICPR), 2010 20th International Conference on*. IEEE, 2010, pp. 3460–3463.

[24] M. Ambai and Y. Yoshida, “Card: Compact and real-time descriptors,” in *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE, 2011, pp. 97–104.

[25] S. Lazebnik, C. Schmid, and J. Ponce, "A sparse texture representation using local affine regions," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 27, no. 8, pp. 1265–1278, 2005.

[26] A. E. Johnson and M. Hebert, "Surface registration by matching oriented points," in *3-D Digital Imaging and Modeling, 1997. Proceedings., International Conference on Recent Advances in*. IEEE, 1997, pp. 121–128.

[27] G. Takacs, V. Chandrasekhar, S. Tsai, D. Chen, R. Grzeszczuk, and B. Girod, "Unified real-time tracking and recognition with rotation-invariant fast features," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. IEEE, 2010, pp. 934–941.

[28] G. Takacs, V. Chandrasekhar, H. Chen, D. Chen, S. Tsai, R. Grzeszczuk, and B. Girod, "Permutable descriptors for orientation-invariant image matching," in *SPIE Optical Engineering+ Applications*. International Society for Optics and Photonics, 2010, pp. 77 980M–77 980M.

[29] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *Computer Vision–ECCV 2006*. Springer, 2006, pp. 404–417.

[30] M. Agrawal, K. Konolige, and M. R. Blas, "Censure: Center surround extremas for realtime feature detection and matching," in *Computer Vision–ECCV 2008*. Springer, 2008, pp. 102–115.

[31] S.-C. Pei and J.-H. Horng, "Design of fir bilevel laplacian-of-gaussian filter," *Signal Processing*, vol. 82, no. 4, pp. 677–691, 2002.

[32] E. Tola, V. Lepetit, and P. Fua, "Daisy: An efficient dense descriptor applied to wide-baseline stereo," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 32, no. 5, pp. 815–830, 2010.

[33] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "Brief: Binary robust independent elementary features," in *Computer Vision–ECCV 2010*. Springer, 2010, pp. 778–792.

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

[35] J. Leitloff, S. Hinz, and U. Stilla, "Vehicle detection in very high resolution satellite images of city areas," *Geoscience and Remote Sensing, IEEE Transactions on*, vol. 48, no. 7, pp. 2795–2806, 2010.

[36] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 24, no. 4, pp. 509–522, 2002.

[37] T. Zhang, Y. Y. Tang, B. Fang, Z. Shang, and X. Liu, "Face recognition under varying illumination using gradientfaces," *Image Processing, IEEE Transactions on*, vol. 18, no. 11, pp. 2599–2606, 2009.

[38] T. Ahonen, A. Hadid, and M. Pietikainen, "Face description with local binary patterns: Application to face recognition," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 28, no. 12, pp. 2037–2041, 2006.

[39] X. Wang, T. X. Han, and S. Yan, "An hog-lbp human detector with partial occlusion handling," in *Computer Vision, 2009 IEEE 12th International Conference on*. IEEE, 2009, pp. 32–39.

[40] B. Jun, I. Choi, and D. Kim, "Local transform features and hybridization for accurate face and human detection," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 35, no. 6, pp. 1423–1436, 2013.

[41] I. M. Creusen, R. G. Wijnhoven, E. Herbschleb, and P. De With, "Color exploitation in hog-based traffic sign detection," in *Image Processing (ICIP), 2010 17th IEEE International Conference on*. IEEE, 2010, pp. 2669–2672.

[42] M. Villamizar, F. Moreno-Noguer, J. Andrade-Cetto, and A. Sanfeliu, "Efficient rotation invariant object detection using boosted random ferns," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, June 2010, pp. 1038–1045.

[43] C. Zhu, C.-E. Bichot, and L. Chen, "Multi-scale color local binary patterns for visual object classes recognition," in *Pattern Recognition (ICPR), 2010 20th International Conference on*. IEEE, 2010, pp. 3065–3068.

[44] T. Ahonen, J. Matas, C. He, and M. Pietikäinen, "Rotation invariant image description with local binary pattern histogram fourier features," in *Image Analysis*. Springer, 2009, pp. 61–70.

[45] U. Schmidt and S. Roth, "Learning rotation-aware features: From invariant priors to equivariant descriptors," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 2050–2057.

[46] J. Peng, B. Yu, and D. Wang, "Images similarity detection based on directional gradient angular histogram," in *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, vol. 1. IEEE, 2002, pp. 147–150.

[47] K. Liu, H. Skibbe, T. Schmidt, T. Blein, K. Palme, T. Brox, and O. Ronneberger, "Rotation-invariant hog descriptors using fourier analysis in polar and spherical coordinates," *International Journal of Computer Vision*, vol. 106, no. 3, pp. 342–364, 2014.

[48] W. Li, S. Xiang, H. Wang, and C. Pan, "Robust airplane detection in satellite images," in *Image Processing (ICIP), 2011 18th IEEE International Conference on*. IEEE, 2011, pp. 2821–2824.

[49] J. Matas, O. Chum, M. Urban, and T. Pajdla, "Robust wide-baseline stereo from maximally stable extremal regions," *Image and vision computing*, vol. 22, no. 10, pp. 761–767, 2004.

[50] M.-Y. Liu, O. Tuzel, A. Veeraraghavan, and R. Chellappa, "Fast directional chamfer matching," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. IEEE, 2010, pp. 1696–1703.

[51] M. A. Fischler and R. A. Elschlager, “The representation and matching of pictorial structures,” *IEEE Transactions on Computers*, vol. 22, no. 1, pp. 67–92, 1973.

[52] J. Sivic and A. Zisserman, “Video google: A text retrieval approach to object matching in videos,” in *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*. IEEE, 2003, pp. 1470–1477.

[53] S. Xu, T. Fang, D. Li, and S. Wang, “Object classification of aerial images with bag-of-visual words,” *Geoscience and Remote Sensing Letters, IEEE*, vol. 7, no. 2, pp. 366–370, April 2010.

[54] M. C. Burl, M. Weber, and P. Perona, “A probabilistic approach to object recognition using local photometry and global geometry,” in *Computer VisionECCV98*. Springer, 1998, pp. 628–641.

[55] M. Weber, “Unsupervised learning of models for object recognition,” Ph.D. dissertation, California Institute of Technology, 2000.

[56] W. Förstner and E. Gülch, “A fast operator for detection and precise location of distinct points, corners and centres of circular features,” in *ISPRS intercommission conference on fast processing of photogrammetric data, Proceedings of*, June 1987, pp. 281–305.

[57] L. Fei-Fei, R. Fergus, and P. Perona, “Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories,” *Computer Vision and Image Understanding*, vol. 106, no. 1, pp. 59–70, 2007.

[58] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, “Object detection with discriminatively trained part-based models,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 32, no. 9, pp. 1627–1645, 2010.

[59] K. Mikolajczyk, B. Leibe, and B. Schiele, “Multiple object class detection with a generative model,” in *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, vol. 1. IEEE, 2006, pp. 26–36.

[60] W. Zhang, X. Sun, K. Fu, C. Wang, and H. Wang, “Object detection in high-resolution remote sensing images using rotation invariant parts based model,” *Geoscience and Remote Sensing Letters, IEEE*, vol. 11, no. 1, pp. 74–78, Jan 2014.

[61] “AFRL CLIF 2007,” https://www.sdms.afrl.af.mil/index.php?collection=clif2007.

[62] C. Yang, M. Bakich, and E. Blasch, “Pose angular-aiding for maneuvering target tracking,” in *Information Fusion, 2005 8th International Conference on*, vol. 1. IEEE, 2005, pp. 8–pp.

[63] H. Ling, L. Bai, E. Blasch, and X. Mei, “Robust infrared vehicle tracking across target pose change using l1 regularization,” in *Information Fusion (FUSION), 2010 13th Conference on*, July 2010, pp. 1–8.

[64] A. Yilmaz, O. Javed, and M. Shah, “Object tracking: A survey,” *Acm computing surveys (CSUR)*, vol. 38, no. 4, p. 13, 2006.

[65] C. J. Veenman, M. J. Reinders, and E. Backer, “Resolving motion correspondence for densely moving points,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 23, no. 1, pp. 54–72, 2001.

[66] D. Serby, E. Meier, and L. Van Gool, “Probabilistic object tracking using multiple features,” in *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, vol. 2. IEEE, 2004, pp. 184–187.

[67] Y. Rubner, C. Tomasi, and L. J. Guibas, “The earth mover’s distance as a metric for image retrieval,” *International Journal of Computer Vision*, vol. 40, no. 2, pp. 99–121, 2000.

[68] G. Laman, “On graphs and rigidity of plane skeletal structures,” *Journal of Engineering mathematics*, vol. 4, no. 4, pp. 331–340, 1970.

[69] O. Tuzel, F. Porikli, and P. Meer, “Region covariance: A fast descriptor for detection and classification,” in *Computer Vision–ECCV 2006*. Springer, 2006, pp. 589–600.

[70] D. Comaniciu, V. Ramesh, and P. Meer, “Real-time tracking of non-rigid objects using mean shift,” in *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, vol. 2. IEEE, 2000, pp. 142–149.

[71] V. Reilly, H. Idrees, and M. Shah, “Detection and tracking of large number of targets in wide area surveillance,” in *Computer Vision–ECCV 2010*. Springer, 2010, pp. 186–199.

PUBLICATIONS

1. Alex Mathew and Vijayan K. Asari, "Rotation-invariant histogram features for threat object detection on pipeline right-of-way," *Proceedings of IS&T/SPIE International Conference on Electronic Imaging: Video Surveillance and Transportation Imaging Applications*, 2014.

2. Ann Theja Alex, Vijayan K. Asari, and Alex Mathew, "Local difference of Gaussian binary pattern: Robust features for face sketch recognition," *IEEE International Conference on Systems, Man and Cybernetics - (SMC)*, 2013.

3. Alex Mathew and Vijayan K. Asari, "Tracking small targets in wide area motion imagery data," *Proceedings of IS&T/SPIE International Conference on Electronic Imaging: Video Surveillance and Transportation Imaging Applications*, 2013.

4. Ann Theja Alex, Vijayan K. Asari, and Alex Mathew, "Gradient feature matching for in-plane rotation invariant face sketch recognition," *Proceedings of IS&T/SPIE International Conference on Electronic Imaging: Image Processing: Machine Vision Applications VI*, 2013.

5. Alex Mathew and Vijayan K. Asari, "Local histogram based descriptor for object tracking in wide area motion imagery," *International Journal of Information Processing*, 2012.

6. Alex Mathew and Vijayan K. Asari, "A local histogram based descriptor for tracking in wide area imagery," *Lecture Notes in Computer Science, Published by Springer-Verlag Berlin Heidelberg, Wireless Networks and Computational Intelligence (Communications in Computer and Information Science series), Edited by K.R. Venugopal and L.M. Patnaik, Proceedings of the Sixth International Conference on Information Processing (ICIP)*, 2012.

7. Ann Theja Alex, Vijayan K. Asari, and Alex Mathew, “Local alignment of gradient features for face sketch recognition,” *Lecture Notes in Computer Science, Published by Springer-Verlag Berlin Heidelberg, Advances in Visual Computering, Edited by G. Bebis et. al., Proceedings of the 8th International Symposium on Visual Computing (ISVC)*, 2012.

8. Ann Theja Alex, Vijayan K. Asari, and Alex Mathew, “Gradient feature matching for expression invariant face recognition using single reference image,” *Proceedings of IEEE International Conference on Systems, Man and Cybernetics (SMC)*, 2012.

9. Alex Mathew and Vijayan K. Asari, “Local region statistical distance measure for tracking in wide area motion imagery,” *Proceedings of IEEE International Conference on Systems, Man and Cybernetics (SMC)*, 2012.

10. Alex Mathew and Vijayan K. Asari, “Local histogram based descriptor for object tracking in wide area motion imagery,” *International Journal of Information Processing*, 2012.

11. Ann Theja Alex, Vijayan K. Asari, and Alex Mathew, “Neighborhood dependent approximation by nonlinear embedding for face recognition,” *Lecture Notes in Computer Science, Published by Springer, Image Analysis and Processing Part I, Edited by Giuseppe Maino and Gian Luca Foresti, Proceedings of the 16th International Conference on Image Analysis and Processing (ICIAP)*, 2011.

12. Alex Mathew, Ann Theja Alex, and Vijayan K. Asari, “A linear manifold representation for color correction in digital images,” *Communications in Computer and Information Science, Published by Springer, Computer Networks and Intelligent Computing, Edited by K.R. Venugopal and L.M. Patnaik: Proceedings of the Fifth International Conference on Information Processing (ICIP)*, 2011.

13. Alex Mathew, Ann Theja Alex, and Vijayan K. Asari, “A manifold based methodology for color constancy,” *Proceedings of IEEE Computer Society Workshop on Applied Imagery Pattern Recognition (AIPR)*, 2010.