A DIRECT APPROACH FOR OBJECT DETECTION WITH OMNIDIRECTIONAL CAMERAS

A Thesis Submitted to
the Graduate School of Engineering and Sciences of
İzmir Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of

MASTER OF SCIENCE

in Computer Engineering

by
İbrahim ÇINAROĞLU

July 2014
İZMİR

We approve the thesis of **İbrahim ÇINAROĞLU**

Examining Committee Members:

Asst. Prof. Dr. Şevket GÜMÜŞTEKİN
Department of Electrical and Electronics Engineering, İzmir Institute of Technology

---

Asst. Prof. Dr. Mustafa ÖZUYSAL
Department of Computer Engineering, İzmir Institute of Technology

---

Asst. Prof. Dr. Yalın BAŞTANLAR
Department of Computer Engineering, İzmir Institute of Technology

17 July 2014

Asst. Prof. Dr. Yalın BAŞTANLAR
Supervisor, Department of Computer Engineering
İzmir Institute of Technology

Prof. Dr. Halis PÜSKÜLCÜ
Head of the Department of
Computer Engineering

Prof. Dr. R.Tuğrul SENGER
Dean of the Graduate School of
Engineering and Sciences

ACKNOWLEDGMENTS

First and foremost I would like to express my sincere gratefulness to my thesis director Asst. Prof. Dr. Yalın BAŞTANLAR for the limitless support during my studies. I feel very lucky to work with him with his patience, motivation and leniency during these past three years. This thesis would not have been completed without his generosity in sharing knowledge. I also thank Asst. Prof. Dr. Mustafa ÖZUYSAL and Asst. Prof. Dr. Şevket GÜMÜŞTEKİN for evaluating this work. This thesis work is partially supported by TUBİTAK with project number 113E107.

I wish to express my thanks to all my friends and colleagues from Computer Engineering Department of İYTE, especially to my office mates for their support and friendship during thesis work.

I also would like to thank to my parents Nuran and Muzaffer ÇINAROĞLU, my sister Bahar ÇINAROĞLU, and Sinem ÖREN for all the times they were there when I needed them, and for their unconditional love.

ABSTRACT

A DIRECT APPROACH FOR OBJECT DETECTION WITH OMNIDIRECTIONAL CAMERAS

In this thesis, an object detection system based on omnidirectional camera which has the advantages of detecting a large view-field is introduced. Initially, the traditional camera approach that uses sliding windows and Histogram of Gradients (HOG) features is adopted. Later on, how the feature extraction step of the conventional approach should be modified is described. The aim is an efficient and mathematically correct use of HOG features in omnidirectional images. Main steps are conversion of gradient orientations to compose an omnidirectional sliding window and modification of gradient magnitudes by means of Riemannian metric. Owing to the proposed methods, object detection process can be performed on the omnidirectional images without converting them to panoramic or perspective image. Experiments that are conducted with both synthetic and real images compare the proposed approach with regular (unmodified) HOG computation on both omnidirectional and panoramic images. Results show that the performance of detection has been improved by using the proposed method.

ÖZET

TÜMYÖNLÜ KAMERALAR İLE NESNE TESPİTİ İÇİN DOĞRUDAN BİR YAKLAŞIM

Bu tez çalışmasında, geniş görüş alanı avantajına sahip olan tümyönlü kameralar ile nesne tespiti yapan bir sistem tanıtılmıştır. Öncelikle, geleneksel kameralar için önerilen yaklaşım ile, perspektif imgeler üzerinde kayan pencereler yöntemi kullanılarak HOG özniteliklerinin çıkarılması işlemi gerçekleştirilmiştir. Daha sonra, hali hazırda uygulanan geleneksel öznitelik çıkarım adımlarının bu amaç doğrultusunda nasıl uyarlanacağı tarif edilmiştir. Buradaki amaç, HOG özniteliklerini verimli ve matematiksel olarak doğru bir şekilde tümyönlü kameralar üzerinde uygulayabilmektir. Uygulanan temel değişikliklerden birincisi, gradyan (yön türevi) oryantasyonlarının tümyönlü imgelerde kayan pencereleri elde edecek şekilde uyarlanmasıdır. Uygulanan ikinci önemli değişiklik ise, gradyan şiddetinin Riemannian metriği ile yeniden uyarlanmasıdır. Adapte edilen bu değişiklikler sayesinde, tümyönlü imgelerin perspektif yada panoramik imgelere çevrilmesine ihtiyaç duyulmadan, direk olarak nesne tespiti yapılması mümkün kılınmıştır. Deney kısmında oluşturulan sentetik ve gerçek imgeler vasıtasıyla, önerilen tespit sistemi ile normal (değişikliğe uğramamış) HOG hesaplama yönteminin tümyönlü ve panoramik imgeler üzerinde karşılaştırması yapılmıştır. Deneysel sonuçlarla yapılan analiz, önerilen metot ile uygulanan nesne tespiti işleminin kaydettiği performans artışını gözler önüne sermiştir.

TABLE OF CONTENTS

LIST OF FIGURES

Figure **Page**

LIST OF TABLES

Table **Page**

LIST OF SYMBOLS

v ..... Normalized Descriptor Vector
v ..... Normalization Constant
S_n ..... Number of Scale Steps
O ..... Overlap Ratio
p_i ..... Position of i-th Detection
x_i ..... Upper Left x Coordinates of i-th Detection
y_i ..... Upper Left y Coordinates of i-th Detection
s_i ..... Scale information that Belongs to i-th Detection
w_i ..... Score Value of i-th Detection
S^2 ..... Sphere Surface
G_S ..... Spherical Gaussian Function
er_i ..... end-radius Value of i-th Omnidirectional Sliding Detection
ma_i ..... middle-angle Value of i-th Omnidirectional Sliding Detection
A ..... Whole Area of Omnidirectional Camera

LIST OF ABBREVIATIONS

SIFT . . . . . Scale Invariant Feature Transform
PCA . . . . . Principal Components Analysis
LDA . . . . . Linear Discriminant Analysis
PTZ . . . . . Pan Tilt Zoom
R-HOG . . . . . Rectangular HOG
C-HOG . . . . . Circular HOG
NMS . . . . . Non-Maximum Suppression
Bbox . . . . . Bounding Box
TP . . . . . True Positive
FP . . . . . False Positive
FN . . . . . False Negative

CHAPTER 1

INTRODUCTION

Our world has a life cycle that is human oriented. Human being is in the center of everything, on the other hand the objects composed our surrounding have a direct or indirect impact on our lifes. From existence of mankind for obtaining benefit to itself, human needs to have continuous monitoring. We always observe animate and inanimate objects to gather advantage against the life struggle.

The reason what made people observe was mainly a sense of wonder against unknown. In primitive history we coped with the surveillance problem by man power. With time, numerous inventions had been occurred and variety of equipment were developed for this purpose. One of the most known inventors Galileo Galilei who lived in 16th century and played a major role in the scientific revolution. His most known achievement is invention of the telescope, thus Galileo has been called the ”father of modern observational astronomy”. Over the centuries, our changing lifestyle has shaped the living area; environment in which we interact has constantly changed. Today mostly we are looking for extracting useful points from ongoing and repetitive daily cases rather than discovering the unknown. From that time to present, the aim of gathering more meaningful information efficiently incites human beings search new observation systems. In this way our observation problem converts into a detection problem with the contribution of computing power. For this reason, research topics such as object detection and scene understanding have emerged from computer vision research area.

In this study our starting point is a traffic scene with its most important components like human (pedestrian) and cars. The better organized traffic provides us the more confident living area. Also other cases like traffic security, autonomous systems for residence or car, car assistance system etc. are the some of the fields that make use of object detection. Various types of approaches have been studied, each of them tried to propose more robust systems. Some of them studied on different camera types, some produced a new image description method or classification method and some of them combined known methods to obtain better result. Different type of challenges makes object detection one of the most attractive field of studies.

1.1. What is Object Detection

Object detection is a challenging computer vision problem that copes with identifying a specific class of objects from the real world environment. The objects of interest, such as bicycle, pedestrian, vehicles, buildings are easily detected by human vision system. However, the same process has turned into a complicated process for machines. Different approaches that were previously proposed for human and vehicle detection are examined in Chapter 2 in detail. Although they have provided dissimilar touch for the same objective, they have discussed on the same basic aspects of object detection. We can represent the main components of object detection problem, and then discuss their role in detection systems:

1. *Image descriptors or feature vectors:* In a digital image, every object class has its own special features that identifies the visual character of that object. Despite, most of these features are based on gradients, edges or colors, many different types of feature were introduced for detecting objects. Some of them gain the low level attributes from a distinguishable small area of a region which is named as local features, on the other hand global features are also available. Moreover, some features are easy to compute while others are very difficult.

2. *Detection framework that is constructed on these features:* The whole region in the image must be controlled by an efficient detection infrastructure. This scanning operation is integrated with feature extraction step because they work simultaneously. Besides, shape of the used detector in detection method is generally depends on descriptor formation. This is one of the important factors that effects the rapidness of detection method directly. For instance, sliding window is one of the most preferred detectors in object detection.

3. *Classifier for learning the interested object and giving object/non-object decision:* Although there are verification methods like feature indexing or feature matching, researchers generally tend to verify or reject the candidate object by classification methods. There are several methods subject to classification: Some of them have a probabilistic touch like Bayesian classifier and some others have neural net-based approaches. In spite of this diversity, they all give object / non-object decision according to interested object model. This object model is generated by the learning process that provides important attributes or features of objects.

1.2. Thesis' Aim and Contributions

On the basis of the general object detection context, especially detection of human and car object classes is subject to this thesis. In spite of wide viewing angle (360 degree) advantage of omnidirectional cameras, studies related with the object detection generally have been carried out by normal viewing angle cameras. The algorithms proposed for standard cameras, also called perspective cameras, can not be used directly on omnidirectional images, because descriptor extraction and classification methods are developed and optimized for perspective cameras. It is necessary to conduct specific studies for omnidirectional camera and determine the appropriate methods and algorithms. There are studies reporting an increase in performance which modified some image processing techniques for omnidirectional camera. Also feature extraction methods that are used for classifying the object has the same potential. Despite this potential, none of the encountered studies handled the omnidirectional camera structure with a feature extraction method for a direct classification on it. Additionally some of these works just used omnidirectional cameras for the sake of determining the position of moving objects.

In this thesis, the aim is to propose a direct approach to perform object detection with catadioptric omnidirectional cameras; if a type of mirror is integrated with a conventional camera, it is called as a catadioptric omnidirectional camera. That is, our method does not require the conversion of the omnidirectional images to panoramic or perspective images. Apart from the advantage of eliminating the image conversion step, the detection performance of the proposed approach is superior as will be given in experiments section. Under the light of literature review, the proposed method is the first that mathematically modifies an object detection approach to be effectively used for omnidirectional cameras.

In our study, HOG is employed for the feature extraction step. And then, the binary classifier that carries out the object/non-object determination is created by linear Support Vector Machine (SVM) based learning process. Also, the sliding window detection framework which relies on HOG descriptor formation is used for scanning the whole image in multiple scales. Consequently, we proposed modifications for the HOG + SVM object detection method for an efficient use with omnidirectional cameras.

Although we concentrate on HOG features in this study because of its adaptability to diverse type of classes, other features based on image derivatives can be modified in the similar manner. Also human (pedestrian) and vehicle (car) selected as a detection target because these two object classes are primary actors of traffic scenes and traffic has a direct

impact on the social welfare. A second contribution is that we construct an omnidirectional image dataset with annotated humans and cars and it can be downloaded from our website [1]. We believe this dataset will be useful to the community for omnidirectional vision based object detection research.

1.3. Organization of the Thesis

In Chapter 2, different approaches on object detection are examined individually within the scope of human detection, vehicle detection and object detection on traffic applications including omnidirectional cameras.

Background related to the proposed work is given in Chapter 3. Firstly, camera types with their specifications are introduced from fisheye lenses to catadioptric omnidirectional cameras in the context of catching wide field of view area. Then, we describe HOG + SVM [12] infrastructure for the purpose of the object detection with their main components. Also some implementation detail is explained.

In Chapter 4, initially it is explained why our approach is theoretically correct. After that, we adopt HOG+SVM detection model and explain how we modify the HOG feature extraction step for catadioptric omnidirectional cameras.

Our experiments and implementation details, given in Chapter 5 were held for human and car detection tasks. Their results indicate that the adaptation of HOG features improves the performance when compared to unmodified HOG computation. We also compare our method with object detection on panoramic images converted from omnidirectional ones and conclude that the proposed method is superior for objects with a width / height ratio < 2.5. Additionally as explained in this chapter, we also modify the post processing implementations according to omnidirectional rotating window for a fair comparison.

Future work and our conclusions are given in Chapter 6 and Chapter 7 respectively. Furthermore some algorithms used in this thesis can be found in Appendix as MATLAB code parts.

[1] http://cvrg.iyte.edu.tr/

CHAPTER 2

REVIEW OF LITERATURE

In the literature there are different approaches on object detection. Detection of specific objects with cameras is a chief task for many application areas such as autonomous driving, traffic analysis, safety and visual surveillance. Considerable improvements in object detection have been recorded in the last decade both in terms of effectiveness and processing time. Majority of this research has concerned human and car detection. Most popular detection approaches include feature extraction + classification technique. This technique diversify according to used feature(gradient, color, edge, corner etc) and classification (matching, SVM, boosting etc.) method. For detailed description of these techniques referenced studies are recommended in the following paragraphs.

The organization of the literature review is as follows. First a general review on object detection is given with an emphasis on human / pedestrian detection. Then, the studies on vehicle detection are given in the context of traffic applications. Lastly, the research including omnidirectional cameras is reviewed.

2.1. Object Detection

The method used generally has two basic stages, first one is extraction of image attributes (feature) and identifiers (descriptor) and second step is the implementation of classification method. A major group in these studies uses the sliding window approach in which the detection task is performed via a moving and gradually growing search window. A significant performance improvement was obtained with this approach by employing HOG features. Firstly Dalal and Triggs [11] proposed to use HOG for the feature extraction step and they used SVM for the classification step. After this approach was proposed in 2005, this technique was enhanced with part based models. For instance, Felzenswalb et al. [18] proposed a method using parts of the object which are spring-like connected to each other and can move independently. Alternative improving was using pyramid HOG features and Intersection Kernel SVM [39].

Besides, Scale Invariant Feature Transform (SIFT) is extensively used image descrip-

tor which similarly carries gradient direction and magnitude information like HOG [38]. Instead of using smoothed weighted histograms of SIFT, Ke et al. [32] applied Principal Components Analysis (PCA) to the normalized gradient patch. Their experiments demonstrate that the PCA based local descriptors are more distinctive, more robust to image deformations, and more compact than the standard SIFT representation. Also, detection methods that based on wavelet transformation or based on attributes that are obtained from the detected edges are still used. Ferrari et al. [19] have introduced a local contour features and their application to object detection. These features are able to cover pure portions of an object boundary, without including nearby spurious edges. Moreover, they can form a wide variety of local shape structures, combine informativeness and repeatability in object recognition. Regarding the classification step, developed versions of SVM (e.g. Intersection Kernel SVM [39]) and Boosting methods [49] are comparatively successful; additionally, nearest neighbour and many different classification methods such as artificial neural networks are used in different studies.

A different model in which a histogram is obtained by grouping the extracted features from the image and operated classification is called 'Bag-of-Words' [17, 46]. Although a lot of changes were suggested on this method, because of containing no attribute location information this approach is mostly suitable for general image categorization and it does not seem suitable for human or vehicle detection. Another approach [36] which can be defined as Shape-based object detection and image segmentation requires the pixel level grouping so it is usually not preferred in fastness based traffic scenario. Still another segmentation based object detection studies are available (Gould et al. [23], Shotton et al. [45]). Edge based features [53] and shapelets [42] are examples of other features that can be used with sliding window approach. More recently, it was shown that using combinations of features outperforms the approaches that use a single type of feature [51]. For a detailed summary and comparison of methods, specific to pedestrian detection, we refer readers to [15].

2.2. Vehicle Detection

Most of the studies explained above focus on the human detection in an exterior or traffic scene, on the other hand detection of vehicle is subject to many remarkable studies. Main approaches related with vehicle in the literature concern detection of a vehicle, vehicle tracking and classification of vehicle. Gupte et al. [24] modelled the vehicle types as rectan-

gles and took advantage of attributes such as width, length and speed, after on they applied a hierarchical classification. Kumar et al. [34] differentiate vehicle by means of background extraction method using stationary camera and they extracted attributes like vehicle size, speed, shape, and position. They classified tractors, trucks, cars, motorcycles and pedestrians classes with Bayesian network based classification method under the light of obtained properties. Morris et al. [40] increased efficiency of detection by adding tracking process to classification system and separated vehicles into three classes (sedan, van, truck). In this study, for conversion of attributes PCA and Linear Discriminant Analysis (LDA) had been tried, classification step is made by weighted nearest K neighbour method. Unlike the others, Kanhere and Birchfield [29]'s work is an useful example of how to use the 3D coordinates of the point (of the height from the road) for classification. Also this research provides some tips in 3D scene for increasing the success of developed methods.

2.3. Detection with Omnidirectional Cameras

With its enlarged view advantage, fewer omnidirectional cameras which provide 360 degree horizontal field of view in a single image may take the place of many perspective cameras. However, so far omnidirectional cameras have not been widely used in object detection research area and also in traffic applications like pedestrian and vehicle detection. The works explained in the following parts address omnidirectional cameras as a subject.

In a study on object detection with omnidirectional cameras [52], a mobile robot is given the images of several objects in the environment and it is asked to recognize these objects. Actually, the omnidirectional image is warped into a cylindrical panoramic image before matching with the images of the objects using SIFT. In another study [2], objects in an indoor office environment (tables, chairs, etc) are classified with a generative model. By these two last researches, the system is first trained with annotated images. Then, given some other images from the same environment, system tries to detect object of certain classes. SIFT features are employed without any modification for omnidirectional cameras.

On the other hand, extraction of scale space on SIFT and conversion of finding attributes into convenient form for the geometry of the omnidirectional image are applied by [41]. Furthermore he examines their method with the version in which SIFT performed on omnidirectional images without any modification and they show their method's superiority. Also the success of the modified SIFT on omnidirectional images against to other descriptors

is shown in Arican et al. [3]' s study. Like it is mentioned in previous research, modification of feature extraction method have been carried out for omnidirectional camera however there have not been any study which use this conversion for object detection and classification for HOG+SVM approach.

Gamallo et al. [21] have been presented A SLAM algorithm, based on FastSLAM, using omnivision. Their system uses an omnidirectional camera, which is specially interesting in indoor environments with a low density of landmarks. And they have performed vision-based Simultaneous Localization and Mapping algorithm by the several landmarks usually detected in each image. This algorithm can be useful for extraction of object features in object detection. In another study, Gandhi and Trivedi [20] discussed on the development of a mobile platform-based vehicle classification and logging system. They proposed a motion-based vehicle detection using an omnidirectional camera on top of a vehicle, also histogram-of-gradients (HOG) based classification were discussed in this study. It was seen that the HOG approach was more appropriate for confirming the presence of vehicle and discrimination at coarse level such as between cars and other vehicles.

In their study, Cheng et al. [8] assume an omnidirectional camera is mounted on a moving platform, which travels with a planar motion.They utilizes a hyperboloid-type omnidirectional camera to expand fields of view by mimicking compound eye of insects. Herceg et al. [26] have analysed an optical flow vector information from the pyramidal Lucas-Kanade algorithm applied to an omnidirectional image. Their experiments showed that the algorithm is able to detect a moving object in an adverse scenario under natural and artificial lighting. In another research, Demiroz et al. [14] have described the first indoor multi-omnidirectional camera dataset for activity recognition and provided benchmark algorithms for tracking human. It is named as BOMNI dataset collected with two omnidirectional cameras simultaneously.

When we analyze the use of omnidirectional cameras in traffic; Cheng and Trivedi [9] worked on a system which detects the line on the roadway by an omnidirectional camera placed on the roof of the car and simultaneously observes the driver. Layerl et al. [35] did a similar work and managed to gather higher resolution image for surveillance of driver's face by the proposed different mirror system. Also, there are works for autonomous driving to find vehicle ways by itself or surpass a trouble in roadway [47]. In pursuit of autonomous driving, shadow in scenes is deleted and background is distinguished in Scharfenberger et al. [43]'s approach.

Various studies were encountered on non-traffic cases in which omnidirectional and

Pan Tilt Zoom (PTZ) cameras are used as a combined system. In one of these studies, Scotti et al. [44] made moving object detection on omnidirectional cameras. By means of making calibration between omnidirectional and PTZ cameras they achieved to direct PTZ camera towards moving objects and made possible to take higher-resolution video stream. In a similar manner, Chen et al. [7] developed a PTZ camera system directed by coming data from omnidirectional camera. Differently, In case of more than one object enter the area of omnidirectional camera at a moment, they proposed the 'hopping from a target to target' algorithm for following the each object in a given time.

Iraqui et al. [27] integrated a mirrored omnidirectional sensor with PTZ camera and by this generated camera system they applied a human face detection in two stages. Firstly, authors use Haar features to perform face detection with catadioptric omnidirectional cameras. Instead of modifying the feature extraction step, they convert the omnidirectional images into panoramic images and directly use the conventional (perspective) camera technique. Then PTZ camera is directed to identified zone and performed the same traditional detection method elaborately. In a similar manner, panoramic images are used in [30] for human detection and in [31] for car detection.

A human tracking method for omnidirectional cameras is proposed in the study of Tang et al. [48]. As a part of the proposed algorithm, HOG features are computed. However, a rectangular rotating and sliding window is used with no mathematical modification for the omnidirectional camera. A study on vehicle detection and classification uses omnidirectional and PTZ cameras in conjunction [33]. They use HOG + SVM approach for vehicle detection in the images of PTZ camera, looking direction of which is determined by the omnidirectional camera. However omnidirectional camera is just used to detect the movement and to determine the size of the moving objects.

In other studies related with this thesis, omnidirectional camera was used as an assistant element near other conventional camera types. In this work an effective and direct solution is targeted by modification of HOG descriptor for omnidirectional cameras, furthermore, a supervised classification method SVM and sliding window localization method are applied for detection step. Also in our previous work [10], HOG + SVM detection model is applied on omnidirectional image, which included experiments with a limited image dataset and considered only human detection.

CHAPTER 3

BACKGROUND

3.1. Camera Types

Many different image acquisition technologies have been employed in computer vision and other areas, all of them aimed to provide a more informative view. Gathering more comprehensive image forces invitation of various camera types.

3.1.1. Fixed Cameras with Standard Viewing Angle

We encounter with this fixed camera type in our daily life frequently. It is generally located to observe a certain viewing angle. It is used to capture perspective (normal viewing angle) images (Fig. 3.1) and it cannot handle the wide field of view scenario that we focus on. In other words perspective cameras are able to surveillance just the objects entering into its view angle from a single viewpoint. So, narrow viewing angle of perspective cameras has limited applications. The way of observing more than one object that belongs to distant views is locating many cameras at strategic points, which makes the solution an expensive one.

Pan Tilt Zoom (PTZ) Cameras : Instead of using more than one camera, a PTZ camera will be used with its capability of remote control on directional moving and zooming. Pan means horizontal moving angle, tilt denotes the vertical moving angle and zoom expresses the capability of zooming in to target. PTZ controls are used with professional video cameras in television studios and referred to as camera robotics. These systems can be remotely controlled by automation systems. PTZ cameras may resolve some shortcomings, a higher resolution may be accessed by zooming skill and can follow objects by rotating tool, taking images from different angles can gather more information for detection. However PTZ cameras can not view outside of the focusing area. Therefore it is not adequate to observe the whole scene at a given time.

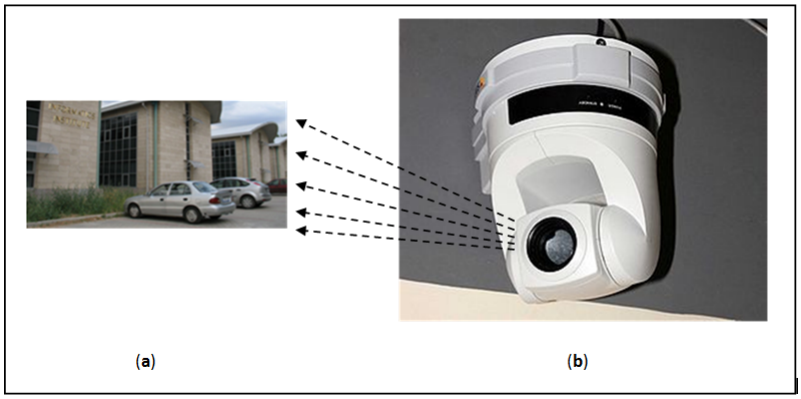


Figure 3.1. (a) A sample perspective(normal view angle) image obtained by PTZ camera. (b) PTZ camera with its movable part.

3.1.2. Wide Field View Cameras

Disadvantage of narrow field view camera and how wide angle cameras cope with this problem are depicted in Fig.3.2. In this sample scenario there is a crossroad which has a traffic flow with its four intersecting roads. Also there is a building overlooking to this intersection area. The fixed camera and wide field of view camera are placed at the corner top of the building, numbered as 1 and 2 in the figure respectively. We can easily see that the fixed camera has a capability of observing unchanging side with its tight angle, whereas the omnidirectional camera located with same place is able to detect all objects coming from each four ways at a moment. In this scenario, performing the task of the wide angle camera with normal viewing angle cameras can be possible by using four fixed cameras facing different directions. This approach will be costly and more complicated to synchronize with the each of four cameras at the same time.

Drawback of fixed camera drive us developing cameras whose observation field cover the whole scene for an extensive object detection. Within the scope of wide field context, omnidirectional imaging systems came out. The main advantage of these systems over a system with a regular lens is the large field of view, which further reduces the number of images necessary to represent a location. The large field of view also makes the system robust to small changes in the environment. Most commonly used image acquisition technologies in omnidirectional imaging are fisheye lenses, panoramic systems and catadioptric systems. A brief description of these camera systems is given in the following paragraphs.

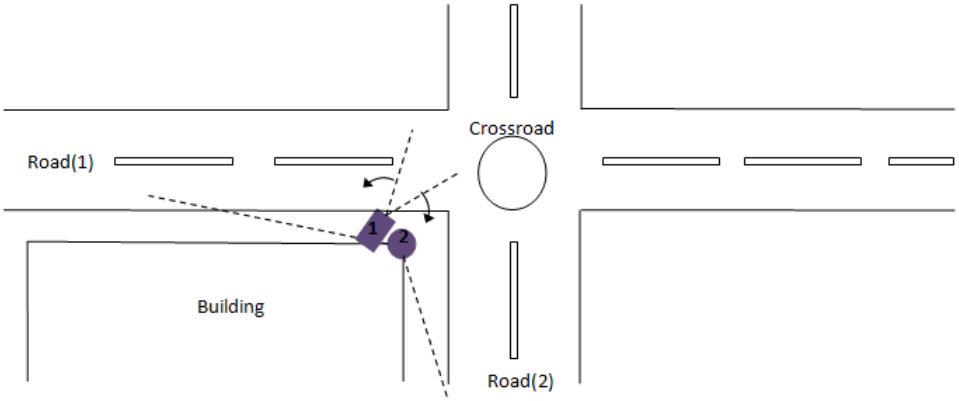


Figure 3.2. A sample scenario comparing the view field of fixed and omnidirectional camera. Both camera are placed corner top of the building which overlooking to crossroad. Rectangle one numbered as 1 is the fixed camera, omnidirectional camera is indicated by circled shape that is numbered as 2. Dashed lines represent the point of view of the each camera.

1. *Fisheye Lenses:* The concept of fisheye view and lens dates back to more than a century. The term fisheye was coined in 1906 by American physicist and inventor Robert W. Wood based on how a fish would see an ultra-wide hemispherical view from beneath the water Fisheye lenses achieve extremely wide angles of view which gives images a characteristic convex non-rectilinear appearance. With a special kind of lens mounted on a standard camera, called 'fisheye lens' (Fig. 3.3), it is possible to obtain a field of view up to about 180-degrees both in horizontal and vertical directions. The widest fisheye lens ever produced featured a 220-degrees field of view. Although images acquired by fisheye lenses may prove to be good enough for some visualization applications, the distortion compensation issue has not been solved yet, and the high unit-cost is a major drawback for its wide-spread applications.

2. *Panoramic Systems:* There are many techniques to acquire panoramic images. Panoramic photography is a technique of photography, using specialized equipment or software, that captures images with elongated fields of view. It is sometimes known as wide format photography. This generally means it has an aspect ratio of 2:1 or larger, the image being at least twice as wide as it is high. The resulting images take the form of a wide strip. Some panoramic images have aspect ratios of 4:1 and sometimes 10:1,

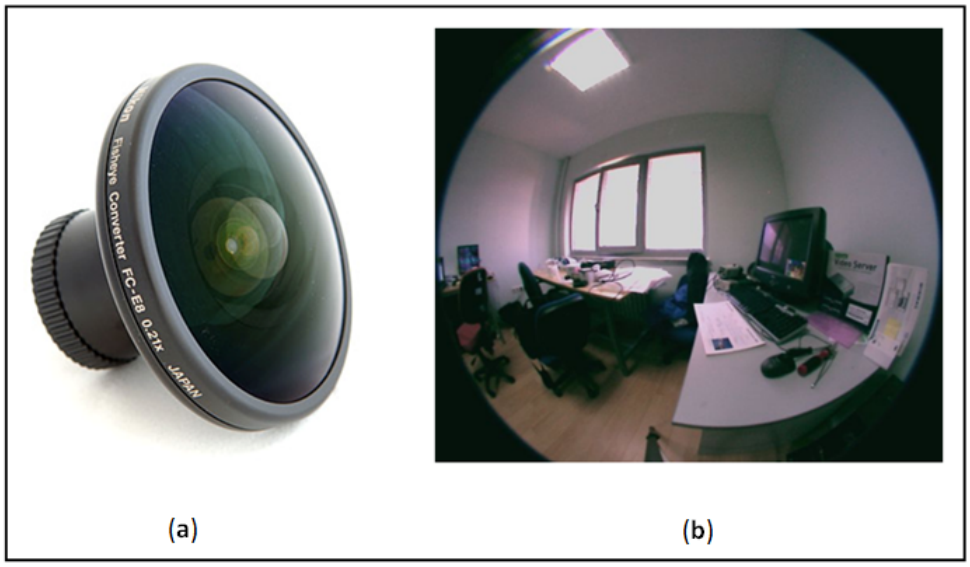


Figure 3.3. (a) A fisheye lens that can capture an 185 degree angle image and (b) a sample omnidirectional image taken by fisheye lens [4].

covering fields of view of up to 360 degrees. Both the aspect ratio and coverage of field are important factors in defining a true panoramic image. Panoramic image sample can be found in Fig. 3.4b that is converted from fixed one by a geometric conversion.

Primitive version of panoramic cameras is named as slit imaging systems. Slit imaging has been one of the first techniques to obtain panoramic images by a physical camera. Various prototypes existed already in the nineteenth century, usually based on a moving slit-shaped aperture. Most of these systems either use a 2D camera or a 1D camera (also called linear camera or pushbroom camera) which scans a scene while moving, generating a panoramic image. For instance, we can acquire panoramas through rotating a fixed camera around a horizontal axis, then by glueing together pixel columns from each image, and used them for map building or 3D measurement. Furthermore, there are new generation cameras that directly serve panoramic images without any moving. Also some more recent digital panoramic imaging cameras [1] (Fig. 3.4a) are mainly developed for robotic and computer vision; similar systems were also developed for photogrammetric applications.

[1] http://ww2.ptgrey.com/spherical-vision

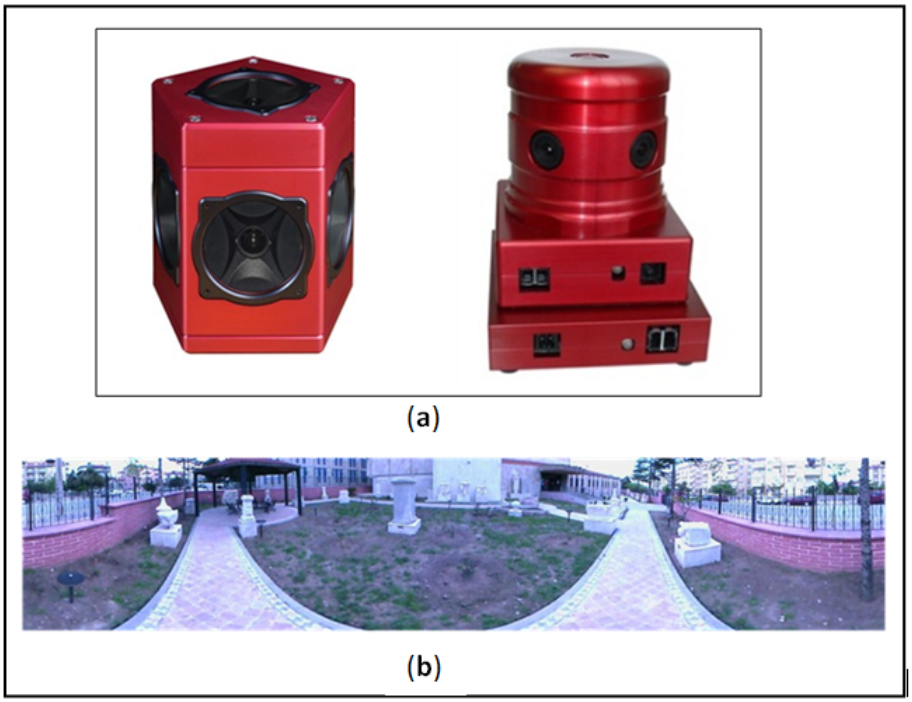
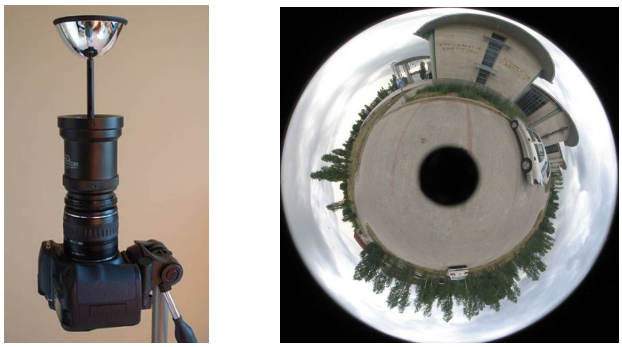


Figure 3.4. Obtaining a panoramic image sample (b) which is captured by digital panoramic camera (a)

3. *Catadioptric Systems:* The third way providing us the omnidirectional vision is catadioptric system. In the literature, different types of catadioptric sensors can be found. In particular, what characterises each catadioptric sensor is the shape of the adopted mirror (parabolic, hyperbolic, conical or ellipsoidal). The most popular catadioptric cameras are para-catadioptric and hypercatadioptric ones, based on paraboloidal and hyperboloidal mirrors. Like the one used in this thesis, if a convex parabolic mirror is placed in front of a conventional camera for this purpose, then the imaging system is called a para-catadioptric omnidirectional camera (Fig. 3.5). Besides, the viewing field can be enlarged by adding different types of mirrors. This kind of special compact system is used for tracking more than one target at a same time. For instance, in their study Layerle et al. [35] proposed a compact catadioptric sensor that tracks the driver face and road line simultaneously.

   Catadioptric cameras provide 360 degree horizontal field of view in a single image (vertical field of view varies). With its enlarged view advantage, fewer omnidirectional cameras may substitute many fixed cameras; also it achieves a field of view possibly

even larger than with a fisheye lens. However, so far omnidirectional cameras have not been widely used in object detection research area and also in traffic applications like pedestrian and vehicle detection.



(a) **(b)**

Figure 3.5. (a) A convex parabolic mirror apparatus is placed in front of a conventional camera to obtain a catadioptric omnidirectional camera. (b) An example image obtained by such a camera [4].

Omnidirectional & PTZ Hybrid Camera Systems: From fisheye lenses to catadioptric omnidirectional vision system, their ability to catching wide field of area increases depend of their technology. However, due to their low and non-uniform resolution, omnidirectional cameras are only able to provide moderate accuracy in both motion/target detection and tracking.On the other hand PTZ cameras stand out with their focusing skill on their target so they can capture an image in high resolution. Different advantages of the two camera types create a dilemma between viewing angle and resolution. To overcome this problem many researchers use both cameras in a combined system.

The common usage of PTZ + Omnidirectional system is implemented by a physically separated PTZ camera that put in together with an omnidirectional camera. In a widely used, moving objects are detected simultaneously by calibrating this combined system [7, 40, 44], the geometry relationship between the omnidirectional and PTZ cameras can be formulated. In this communicated system, PTZ camera is generally directed by an input value that is transmitted from an omnidirectional camera. A sample scenario can be demonstrated in Fig. 3.2. If there was a PTZ camera instead of the fixed (numbered as 1) one in this figure, it might easily scan the range from road 1 to road 2. Therefore, the dilemma between viewing angle and resolution can be solved with this integration.

Consequently, steering the PTZ camera to the all moving targets is not a wise action. The more effective solution for a rapid multi-target detection is performing pre-classification on omnidirectional camera. We must emphasize that, this is the motivation to our work for detecting certain objects on omnidirectional images.

3.2. Object Detection with HOG + SVM

This chapter describes the how HOG and SVM can be integrated for the purpose of the object detection. HOG feature set and this combined approach are firstly proposed by Dalal and Triggs [11]. The entire object detection flowchart, based on HOG + SVM model can be composed like in Fig. 3.6. The object detection process is divided into three core phases, including:

1. The feature extraction phase corresponds to encoding the regions of image effectively. In other words representation of the whole image is generated by collecting HOGs over detection window from all locations of image.

2. The classification phase gives decision whether incoming window is interested object or not. This phase also includes the learning sub-phase to shape the object class model.

3. Detecting with window classifier result in multiple overlapping detections. Fusion of multiple detections is provided with suppressing the less meaningful ones in this phase.

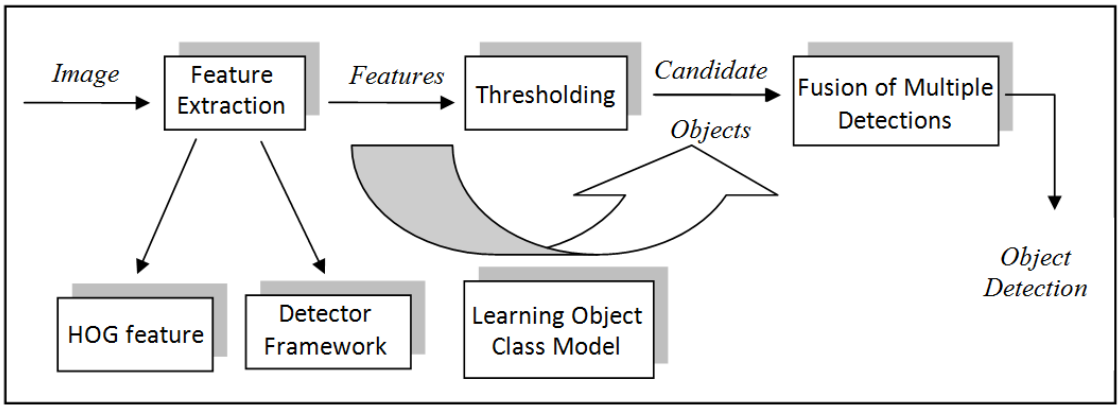


Figure 3.6. The entire object detection steps based on HOG+SVM model.

To summarize the flow briefly, the sliding window detector scans the image in all possible scales, so by this fixed window we can reach all local parts of the image. These local parts are encoded in context of HOG feature and all this fixed windows turn into feature vectors. These feature vectors are also used in supervised learning phase, training set serves the necessary information of object classes. After creation of interested object class pattern named as model, linear SVM evaluates the feature vector which is in the same format with training feature but extracted from the test images. Although the evaluated results that belongs the local parts are thresholded, this verification still results in multiple overlapping detections. Later on, these overlapping detections called as bounding boxes are fused into most meaningful one. Final bounding boxes denote the object instances (car or human) in explored image.

Following sections provide profound explanation of these chief phases mentioned above with their parametric details.

3.2.1. HOG Based Descriptor

With the scope of object detection, edge orientation histogram has been widely used as an image identifier. Also the concept of dense and local histograms of oriented gradients became popular as a descriptor. This section gives implementation details of HOG feature extraction chain. Main objective of such method is generating characteristics in a local part of an image by overlapping normalized histograms of gradient orientation. These histograms reflect the appearance and shape of image regions.

Basically, types of HOG descriptor are proposed according to their pattern of cell grids. If a rectangular cell is used, it is called as Rectangular HOG (R-HOG), alternatively it can be applied as Circular HOG (C-HOG). The only difference point between these two descriptor is; one captures the local shape information in rectangular pattern, the other represents the same local region in circular pattern. Both descriptor types have the same fundamental computation steps.

R-HOG descriptor is used as the default descriptor of all work in this thesis. Since we prefer a rectangular shape sliding window detector for detection phase, using this shaped construction makes our descriptor generation process less time consuming. Roughly, encoding image gradients orientations in histograms can be achieved by the following sequential operations: First of all, we compute gradient of the image, and then local gradients are

binned according to their orientation, weighted by their magnitude. These accumulated bins construct histogram of each cell. After desired number of cells are grouped and compose the blocks, overlapping blocks are normalized. Finally, compiles the blocks and reach the feature vector of our image. The following paragraphs explain each of these steps in more comprehensive way.

1) *Gradient Computation :* The image is filtered with two one dimensional filters so on the gradient of the image has been gained easily. In horizontal direction, (-1 0 1) filter is practised on image initially, and the transpose of the same filter (-1 0 1) is applied in vertical direction. This gradient computation filter is provided as default in digital image environment. Also there are lots of parameters that we need to set in gradient computation. One of them is signed or unsigned gradient settings. The unsigned case is used where the direction of the contrast is unneeded, on the other hand if direction of contrast is important for our implementation we will use signed gradient. The gradient values range from 0 to π and 0 to 2π in case of unsigned and signed respectively. In our case, we use unsigned gradient because we are just interested in the differential zone regardless of its direction. After gradient computation steps computing the histogram for each cells according to the number of bins.

2) *Cell and Block computation:* The power of HOG depends on separation of the image into cells. Cells are formed with a size of pixel, this size give the shape of the cells. Simply it can be defined as the number of pixels contained in a cell. In this work we use [4x4] pixel sized cells for car descriptor and [8x8] cells for human descriptor. The difference between cell sizes is derived from the differences in used sliding window size. Then cells are converted into histogram of gradient by means of compiled bins (Fig. 3.7). Each bin has a vote and votes are weighted with the magnitude of a given point which belongs to cell. If we prefer to construct our histogram of cell with larger number of bins, our histogram becomes more detailed. In this study our histograms have same number of bins. Both the car and human descriptor uses 9 bins. These histograms that denote the each cell get together and grouped histograms form our block. We need these blocks because we can not accumulate this histogram directly in a single feature vector. Before the creation of our feature vector we need normalization, thus normalization is done among a group of cells, which is called block. The block sizes in our study are fixed for both object classes. Each of the car and human descriptor uses [2x2] block size; hence it means our blocks include 4 histograms. Later on, normalization function is computed over the block and all histograms inside the blocks are normalized according to this normalization function. Shift value, which is named

as descriptor stride, means the number of cells overlapped by block, in other words it is the quantity of block striding cell by cell. In this work the descriptor stride is set to half the block width, in order to have 50% overlap. This amounts to 4 pixels for car descriptor and 8 pixels for human descriptor. Also the types of normalization function could be applied along these four schemes [12]:

1. L2-norm, $v \leftarrow v/\sqrt{\| v \|_2^2 + \xi^2}$;
2. L2-Hys, L2-norm followed by limiting the maximum values of v to 0.2 and renormalising, as in Lowe [38];
3. L1-norm, $v \leftarrow v/(\| v \|_1 + \xi)$;
4. L1-sqrt, L1-norm followed by square root, $v \leftarrow \sqrt{v/(\| v \|_1 + \xi)}$.

We use L2 normalization function in our both car and human descriptor. In these schemes, 'v' is the normalized descriptor vector and 'ξ' is a small normalization constant to avoid division by zero.

When this normalization stage is operated, all the histograms can be compiled in a single feature vector and the complete HOG feature extraction process is depicted in Fig. 3.7

3.2.2. Detection Framework

Majority of object detection systems use the sliding window approach. This simple window architecture has various advantages. It permits a traditional classifier to be used for detection and a significant performance enhancement is gained with this approach by employing HOG. For giving a brief explanation, sliding window can be imagined as fixed sized mask, which moves over the image with a suitable sliding window stride.

Sliding window must have an efficient scanning strategy. Bypassing any local part without scanning is unacceptable case that may causes to miss the target object instance. Therefore it is essential to scan the image in all possible scales. To handle this important scanning problem, one of the applicable solution is changing the sliding window size. However, this strategy forces us to create separate descriptor formation for each of the resized window. So generating of the HOG descriptor again and again, costs too much in terms of running time and memory usage. Another approach for solving this problem is resizing the image in different scales with bilinear interpolation. In this study the detection framework

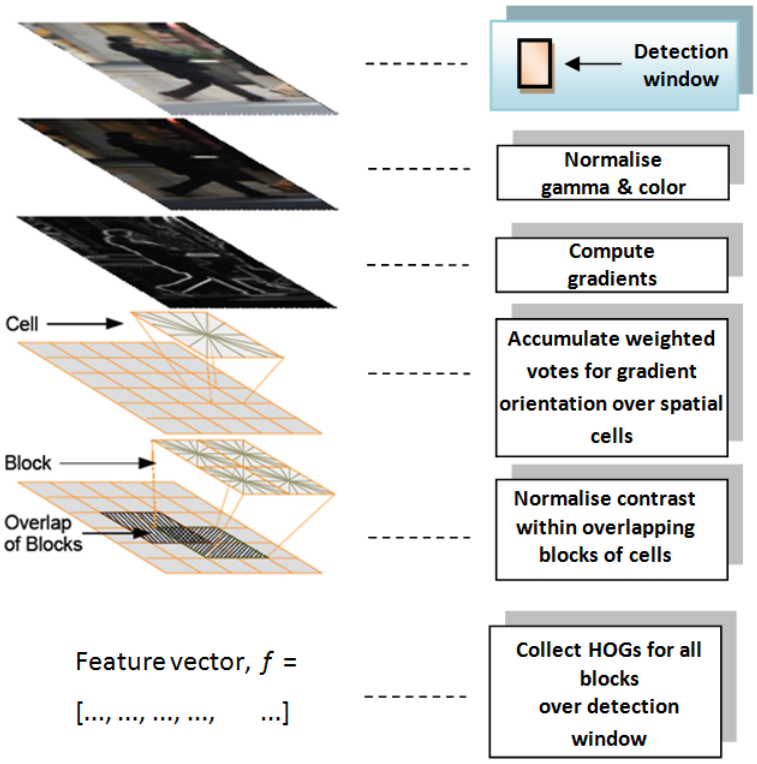


Figure 3.7. Feature extraction process [12].

is built on this simple scanning strategy. Hence, thanks to the fixed size window, we can scan the target object instances which will have different size and position in image. By the resized images our detector is able to capture the whole region by sliding. Resizing the image instead of resizing the sliding window brings about the same effect on detection process without the need of any other HOG formation.

In order to determine how many resizing operations our image needs, we take into account the sliding window size and window shifting in pixel size which is named as sliding window stride. After computing the number of scale steps S_n, our sliding window knows how many time it is going to move over the each scaled image in horizontal and vertical direction. This stride parameter expresses how far away the sliding window moves in the each iteration of each scaled images. The necessary computational steps as codes are given in Appendix A under the title of Software for Detection Framework.

After explaining the scanning technique, we will consider on what window size is the most suitable one for our car and human classes. Different window sizes for different object classes are recommended in related works. We use [128x64] pixel sized sliding window

for human detection as proposed by Dalal and Triggs [11]. [40x100] pixel sized window is preferred for car detection and height / width ratio is obtained from the images in the training dataset. Later on, with determination of sliding window size for each class, we can compute the dimension of final feature vector. While computing this vector we take in account the parameter values which are mentioned in cell and block computation subsection.

For human image, descriptor stride is 8 pixels which means our [128x64] sliding window moves by 8 pixels in every iteration. Horizontal step number can be computed from $((128 \ / \ 8) - 1)$ as 15 and vertical step number is 7 from $((64 \ / \ 8) - 1)$ computation. Also each histogram have 9 bins and each block have 4 histograms, thus 36 oriented bins come from every computed blocks. As a result this multiplication ($15 \times 7 \times 36$) gives the final feature vector dimension. To sum up, for human detection our sliding window return the [3780 x 1] fixed sized matrix. It means that each of the candidate detected area is encoded into this feature vector. For car image, all parameters are similar with human descriptor without the descriptor stride and sliding window size. In car image descriptor stride are 4 pixels so [40x100] sliding window moves by 4 pixels in every iteration. With the same computation method car descriptor has 9 horizontal step numbers and the vertical step number is 24. Therefore from this multiplication operation ($9 \times 24 \times 36$) all car detections formed in [7776 x 1] feature vector. In addition this feature computation operation allows us pre-allocation of the register. Thanks to this beforehand preparation we achieve to improve our algorithm running time performance.

3.2.3. Classification with SVM

As we mentioned in previous parts, the detection system in this study is constructed on a supervised learning technique. It means we have predefined samples which are sampled as object or non-object. We find these labelled inputs from a set of training image examples with and without object instances. Therefore, the HOG descriptor form 'feature vector' is used as an input value which represent the labelled fix-sized samples to learn a decision function. In this thesis, we have used a Support Vector Machines (SVM) classifier.

SVM is a binary classifier algorithm that creates set of hyperplanes in a high dimensional space and makes its mind according to this metric. There are several kinds of SVM but we use linear kernel implementation named as SVMLight in these all applications. SVM-Light is an implementation of Vapnik's Support Vector Machine [50] for the problem of pat-

tern recognition. The optimization algorithms used in SVMLight are proposed in Joachims' study [28]. Our SVMLight classifier just need 'C ' parameter in learning process, this value represents the weight for misclassified points. Later on, in our whole learning process, we adjust this parameter as 0.5 for generating both car and human classifier.

Before conducting the SVM classifier on candidate feature vector, we have to complete the learning phase and generate the model also it can be named as binary classifier. Each of the car and human binary classifier is trained in two stages like shown in Fig. 3.8. In the first stage, we train detector with positive and negative training image sets. Both the positive and negative images are converted into the form of the object sliding window form, it means car training input images must be in size of [40x100] and human input images have to be in size of [128x64]. After training on these centred and normalized images we generate the initial object model. It is called as initial classifier because it is the temporary one which is used to create the final classifier. This classifier enhancement is obtained by retraining our model in the second stage. The initial classifier is used to scan the negative training images for catching the false negative samples. Also it can be named as 'hard negative examples' which affects our classifier performance in a bad way, because our classifier evaluates this object-free window as a target object. Then, the classifier retrained with this enriched training dataset. This new training set comprise from the positive training set which is not changed and the negative training set which contains initial negatives and hard examples. Our test shows us, larger number of hard examples does not necessarily improve the performance, this situation changes from detector to detector. Also the number of hard examples is determined with the initial detector performance. The necessary computational steps as codes are given in Appendix B under the title of Software for Learning Phase.

Our training examples are labelled with (-1, 1) values. It means, our positive training feature vectors are assigned as '1', and the negative training feature vectors are labelled as '-1'. By means of these labelled data our classifier learns the average target object appearance and generates the model.

As pointed out in above, we can built our classifier on training dataset. Thanks to the modification of HOG on omnidirectional image, we introduce in Chapter 4, we are able to use the same classifiers which is trained with perspective car and human image. If we explain further, we can classify the either omnidirectional or panoramic images with the same classifier that generated here on the perspective camera dataset.

Under the light of these information, we create the object model with SVMLight and previously defined HOG detector for human with INRIA human dataset [11]. This dataset

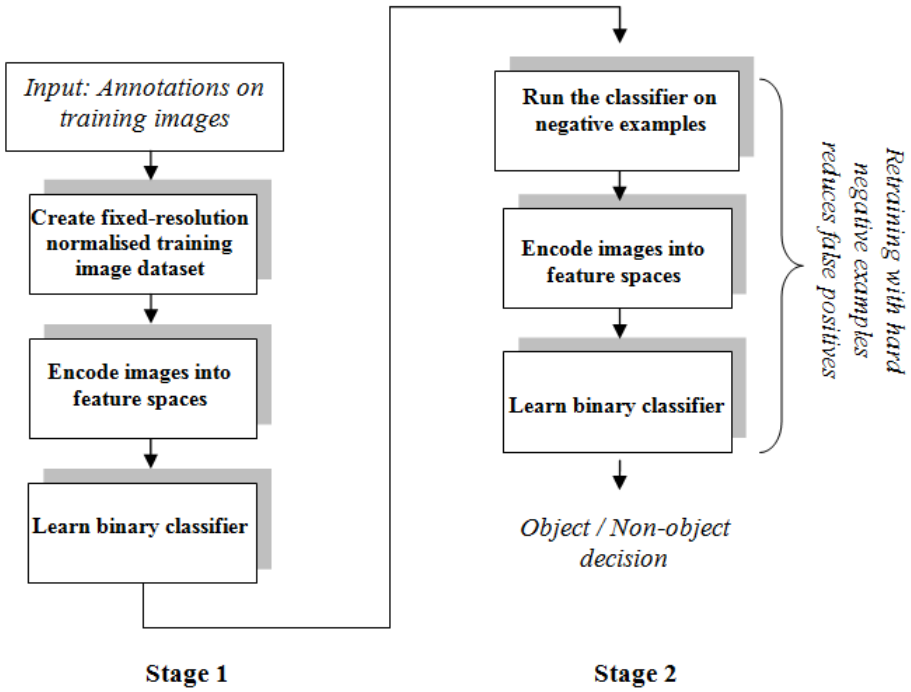


Figure 3.8. Learning Phase [12].

includes 2416 positive windows and 1218 negative images as a training dataset. It is easy to capture the [128x64] window samples from positive training window, because of the centred human instances. Also the negative images is randomly cropped for ten times from the each negative images, thus we use 12180 negative sample windows for training. After that we check our human classifier on INRIA's test set which contains 1126 positive window with 453 negative images. Results show us human detection is carried out successfully.

In a similar manner, we trained our car detection classifier with combination of UIUC car dataset [1] and TU Darmstadt car database [36]. Mainly we use 550 UIUC positive training images with 500 negative training images of UIUC dataset. In addition we enrich this positive training set with Darmstadt's 52 side view images. By this combined training set, which totally contains 602 images, we enhance the success of our car detector. After extraction of 2733 false negative samples we apply the retraining process, and then we reach the more successful classifier. Also the test result on UIUC test set which includes 170 images approves the perfection of our classifier at detecting the cars. All this learning steps are conducted as mentioned in Appendix B.

After we obtained our final classifier, it is ready to examine on test image in detec-

tion life-cycle. The candidate feature vectors, which is extracted from test image by HOG detector, are evaluated with classification function of SVMLight. Each of the detected candidate window is scored that is placed on upper-left corner of each detected bounding box like demonstrated in Fig. 3.9. This SVM score tags represent the possibility of belonging to interested object class, moreover we use these scores in all evaluation cases like fusing multiple detection, determining the overlapping detections, thresholding etc.

3.2.4. Localization of Multiple Detections

After obtaining corresponding scores for each feature vector by classifier, multiple overlapping detections for each object instance are produced. We need to eliminate low scored detected windows by thresholding. The threshold may vary according to the employed SVM implementation. But thresholding the lower scored detections is not enough for a successful result. We still need to fuse multiple detections which intersect with each other on the same object instance, and also we need to evaluate our final detections

Before this localization operation, we decide on by which computation metric we determine whether detections are overlapping or not.

3.2.4.1. Overlapping Computation Metric

For eliminating or comparing the bounding boxes, we have to find out how much these boxes are far away to each other. In other way our main problem can be defined as determining the intersection ratio between the intersecting boxes. Equation 3.1 gives us the overlapping determination strategy: firstly, the intersection area of these boxes is worked out, and then the found value is proportioned to the value that corresponds to the union area of these boxes [16, 12]. As explained before, our detection network is based on the window scanning approach, using this overlapping computation metric is the suitable way because our detection results are rectangle shaped. In this equation O denotes the overlap ratio and detected windows represent the compared any two bounding boxes. To be said that two bounding boxes belong to the same object, O must exceed 0.5 (50%). This threshold is accepted as a default value in the examined related works.

$$O = \frac{area(detectedwindow1 \cap detectedwindow2)}{area(detectedwindow1 \cup detectedwindow2)} \tag{3.1}$$

3.2.4.2. Non Maximum Suppression

After we defined our overlapping computation method, now we are going to explain how we handle with multiple overlapping detected windows. For this purpose several methods centered on heuristic are proposed, but we need the one that especially works on our detection scale space like *Non-Maximum Suppression* (NMS). Several types of NMS versions are available, some of them are easy to implement and others can be highly efficient. We employ the basic NMS form [12] for our implementation in this thesis.

The essence of this process is while eliminating the redundant intersecting detections, trying to choose the highest scored one between its neighbourhoods as a final detection. It is named as *Non-Maximum Suppression* because this fusing method makes a local maximum search. Where local maximum is greater than the others in their region, the NMS algorithm chooses this one as a superior bounding box of this focused area. The raw incoming detections from our SVM classifier are at very different scales or positions, in other words they are not in same scale space to finding the local maximums together. Therefore, for finding the dominant detection we initially have to know the accurate position of the boxes on the original image. The solution is representing detections in a position scale pyramid that means we map our detections in 3-D space and weight them by their SVM scores.

Let $p_i = [x_i, y_i]$ and s_i denote the raw detection position (upper left coordinates) and scale, respectively, for the i-th detection. The detection score is denoted by w_i. The *integrated form* is defined as $[x_i, y_i, s_i, w_i]$, that corresponds to the input parameters of our NMS function especially for rectangular shaped Bounding Box (Bbox). Firstly we convert this 3-D space form, $[x_i, y_i, s_i]$, into its real size in 2-D. Roughly it is managed by resizing the raw coordinate $[x_i, y_i]$ according to s_i information. This conversion is conducted for all BBoxes to find out the real position and relationship between them. This reduced form is called as rescaled-real detection in this thesis. Secondly the detections are sorted in descending order by this sorting operation we guarantee to pick the greatest scored Bboxes finally. Then, we check whether compared Bboxes intersect or not, if there is no intersection it means these Bboxes belong to different object instances so they are not suppressed. However if there is an intersection, we compute the intersecting area. The maximum coordinate values of left-

top corner gives the intersecting area's left-top coordinate, in a similar manner the minimum coordinate values of right-bottom corner gives the intersecting area's right-bottom coordinate. After gathering the intersection boundary we compute the union area and compute the O value by the same overlapping computation metric that is defined before. Finally if O value greater than threshold ratio, NMS function decide on these Bboxes belong to same object instance and suppresses the lowest scored one. In other words NMS function returns the picked detections indexes. NMS algorithm for regular sliding windows (rectangular Bboxes) is presented in Appendix C.

The threshold of 50% may not be adequate in certain applications and it can be adjuster by the user. Intersecting bounding boxes in a different overlapping ratio and imaginary case are demonstrated in Fig. 3.9. In this imaginary case we set our threshold as %50. Firstly our NMS function sorts the BBoxes in descending order BBox3, BBox2, BBox1 respectively. This order depends on the score of these detections, so the comparison process between these BBoxes starts from BBox3. In the first case, (BBox 3 with BBox2 and BBox3 with BBox1) because of insufficient O value both these two intersecting boxes are accepted as different two detections. However in the second case, (BBox 2 with BBox1) bounding boxes intersect with a O value that grater then threshold so one of them is accepted as detection, the other one, namely Bbox1, is eliminated. By this elimination BBox1 is situated out of the next comparison. As a result, after NMS applied on these three detections BBox2 and BBox3 are picked as final detections. Like being in this sample case the accuracy of our NMS algorithm is tested by various synthetic cases.

Moreover, for visualization of all detections we also need a drawing function that is called for drawing bounding boxes. The same representation of detection in scale space pyramid with confidence scores is also used as a parameter for this function. Before plotting a detected zone, rescaled-real form of Bbox is obtained like in NMS function, and then plotting the found coordinates create the view of detection. In figure 3.10, a sample NMS operation shows the local maximums on an UIUC test image and how multiple detections are suppressed for both cars.

3.2.4.3. Evaluation Methodology

The same overlapping computation metric with the same BBoxes comparison methodology in the NMS is also used for evaluating our object detector performance. Differently

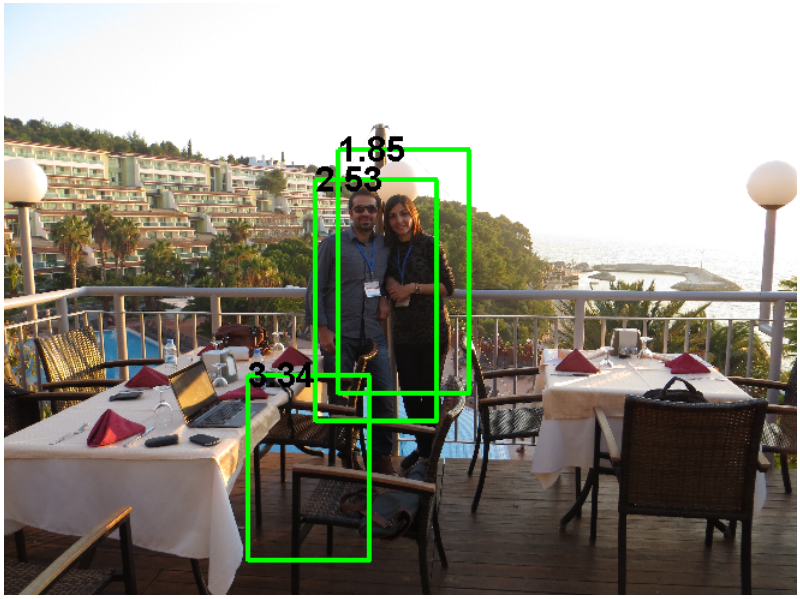
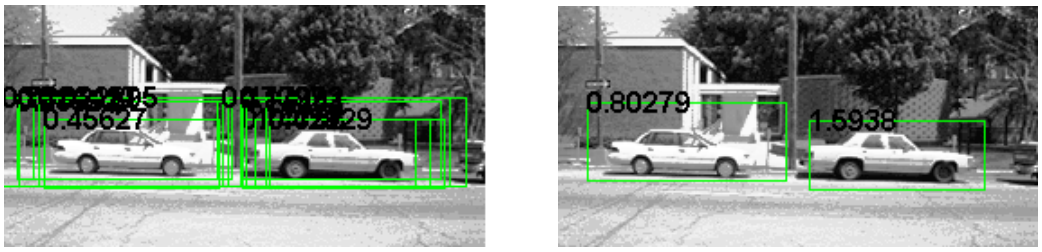


Figure 3.9. A synthetic scenario that contains three intersecting detections with different overlapping ratios; O value between BBox1 (1.85) and BBox2 (2.53) is greater than %50, BBox3 (3.34) intersects with the others in lower O value.

from there, in this case detections are compared with the groundtruths. Groundtruth are equal to the number of objects in image and it is a representation in BBox form that gives information about where the object instances are located. It is also named as *annotation*. After the *rescaled-real* forms of detections are obtained, annotations are checked with all these detections whether they intersect. The proportion of intersection area to union area gives us the O value and by this value we give the decision on correctness of our detections. If O value of detection with any annotation greater then threshold the detection is accepted as true positive (TP), that mean this found detection really represent a object instance in this image. If there is inadequate O value we label this detection as false positive (FP). Also if there is an annotation and no detection is found that corresponds to it, this unfound object instance number gives us the false negative (FN). FN can be called as the number of missed objects.

To measure how a detector performs in practice and localizes the detections on image, we draw precision-recall curve. The aim of precision-recall curves is to show the two performance metrics together: Precision (#True positives / #True Predicted positives) and Recall (#True positives /#True Actual positives). Therefore plotting a precision versus recall on a



(a) **(b)**

Figure 3.10. An UIUC test image, (a) after multiple scales dense scan (raw detection result), (b) fusion of these multiple detections with non-maximum suppression.

log-log scale gives us this curve, in our experimental results we plot recall along x-axis and precision along y-axis. The larger the area under the curve, the better the performance of the algorithm.

CHAPTER 4

IMPLEMENTATION OF THE PROPOSED METHOD

After the preparation of standard HOG + SVM infrastructure, the explanation of how it is applied on omnidirectional images is presented in this chapter. In order to obtain a complete descriptor of an omnidirectional image, we propose modifications on standard HOG + SVM detection method.

When considering the all phases of regular HOG + SVM method which were introduced in the previous chapter, the real differences occurs in HOG descriptor formation stage. Especially Gradient computations for omnidirectional camera with a camera calibration technique and conversion of gradients for omnidirectional sliding window are the additional steps in feature extraction process. These additional parts are demonstrated in Fig. 4.1 to show where exactly the modifications are placed in feature extraction step.

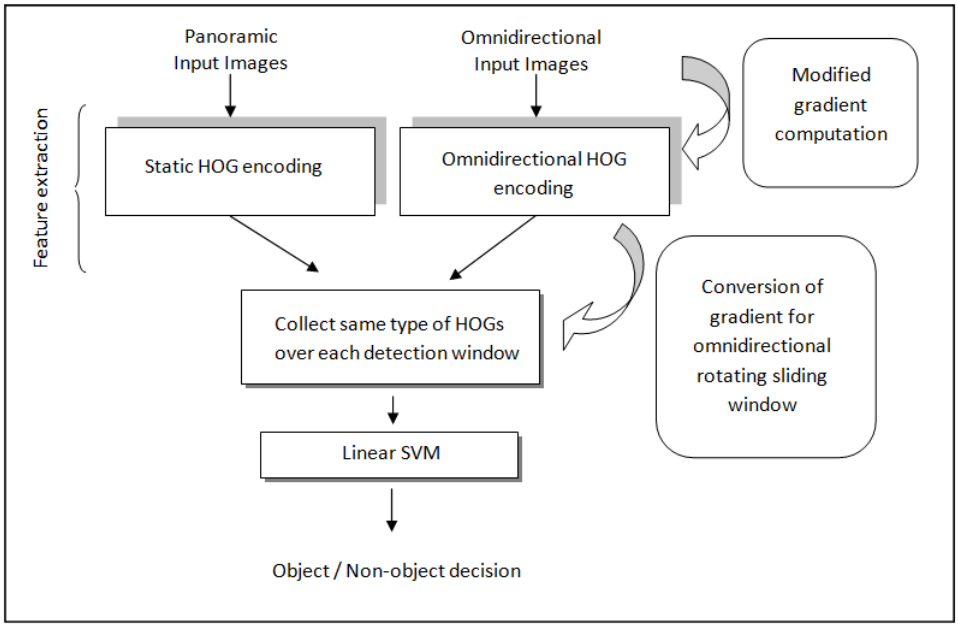


Figure 4.1. In which steps of feature extraction life-cycle we applied the modified HOG descriptor is shown.

These two modifications enable us to carry out a direct classification on omnidirectional image. In this way, we can use one type of perspective image trained classifier on

both omnidirectional and panoramic images. In other words, since we train our both detectors with perspective camera training images, thanks to the modified HOG we can give the collected HOG feature to the same classifier.

4.1. Processing of Omnidirectional Images

Due to their non-linear imaging geometry, working with omnidirectional cameras requires geometric transformations. At first sight, converting an omnidirectional image to a panoramic or several perspective images may seem to be a practical solution. However, it has two major drawbacks: The conversion, which is a non-linear warping, can be computationally expensive for large video frames especially when an omnidirectional image is converted to numerous perspective images to properly fit sliding windows. More importantly, the interpolation required by the image warping introduces artifacts that affect the detection performance.

Among a small number of omnidirectional object detection studies (cf. Section 2.3), none of them developed a method peculiar to omnidirectional cameras. On the other hand, last decade witnessed some effort on computing SIFT features in omnidirectional images. Starting from [13], researchers tried to avoid warping omnidirectional images and instead they assumed a unitary sphere S^2 as the underlying domain of the image function. When these studies (which consider the convolution step of SIFT) are examined, several approaches can be observed. Below, we describe these approaches briefly.

1. The simplest approach would be backprojecting the image onto a sphere surface S^2 and convolving it with a spherical Gaussian function G_S [6]. Since this approach requires resampling of the whole image, authors in [13] project the kernel G_S into image plane instead of backprojecting the image onto S^2, and the convolution is carried directly on the image plane. This avoids image resampling but since the mapped Gaussian kernel changes at every image location it leads to an adaptive filtering. This computational complexity makes the solution unsuitable.

2. Another approach processes omnidirectional images on the sphere after an inverse stereographic projection [25]. Scale space is computed with Gaussian kernels on the sphere, while, the convolution is performed using the spherical Fourier transform. It was stated in [3] and [37] that this operation leads to aliasing issues due to bandwidth

limitations.

3. The processing on the sphere is achieved through a suitable differential operator that adapts to the non-uniform resolution, while using the original image pixel values. In [5], scale space representation is computed using the heat diffusion equation and differential operators (Laplace-Beltrami operators) on the non-Euclidean (Riemannian) manifolds. Moreover, authors in [3] tested this approach by evaluating the matching performance of SIFT on rotated and translated images. Lastly, authors in [41] compared the original SIFT with the version modified by Laplace-Beltrami operators on the Riemannian manifolds and mentioned that the modified version has a better performance. They also extended the method to all central catadioptric systems. Later, this approach was extended to radially distorted images as well [37].

Exploiting the experience gained by the summarized previous work, we decided to compute the gradients on Riemannian manifolds and adapted the HOG computation step (section 4.2.1)of our algorithm accordingly.

4.2. The proposed HOG computation

In the sliding window based object detection approach, a window is moved horizontally and vertically on different scales of an image. No rotation is applied since there is an assumed orientation of the object, for instance pedestrians should be upright. In a similar manner, to detect objects in omnidirectional images, we rotate the sliding window around the image center. In addition, to achieve a mathematically correct detection method, we modify the image gradients. The operations that we perform can be divided into two steps:

1. Modification of gradient magnitudes using Riemannian metric.

2. Conversion of gradient orientations to form an omnidirectional (non-rectangular) sliding window.

4.2.1. Modification of Gradient Magnitudes Using the Riemannian Metric

4.2.1.1. Sphere Camera Model

We use the sphere camera model [22] which was introduced to model central catadioptric cameras. The model comprises a unit sphere and a perspective camera. The projection of 3D points can be performed in two steps (Fig. 4.2). The first one is the projection of point **Q** in 3D space onto a unitary sphere, resulting in point **r**, and the second one is a perspective projection from the sphere to the image plane, resulting in point **q**. This model covers all central catadioptric cameras with varying ξ. $\xi = 0$ for perspective cameras, ξ = 1 for para-catadioptric cameras (the ones using a paraboloidal mirror), $0 < \xi < 1$ for hyper-catadioptric cameras (the ones using a hyperboloidal mirror).

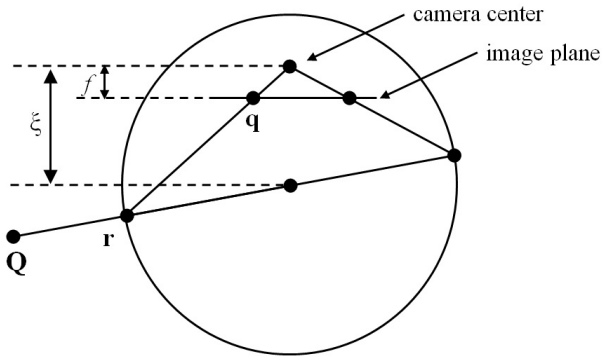


Figure 4.2. Projection of a 3D point onto the image plane in the sphere camera model.

A point on the sphere **r** = (X, Y, Z) can also be represented by two angles (θ, φ), the former is the vertical angle and the latter is the azimuth (Fig. 4.3a). In para-catadioptric case (ξ = 1), if we place the image plane at the south pole (which only differs the scale), $f = 2r = 2$ and the perspective projection within the sphere model corresponds to the stereographic projection (Fig. 4.3b).

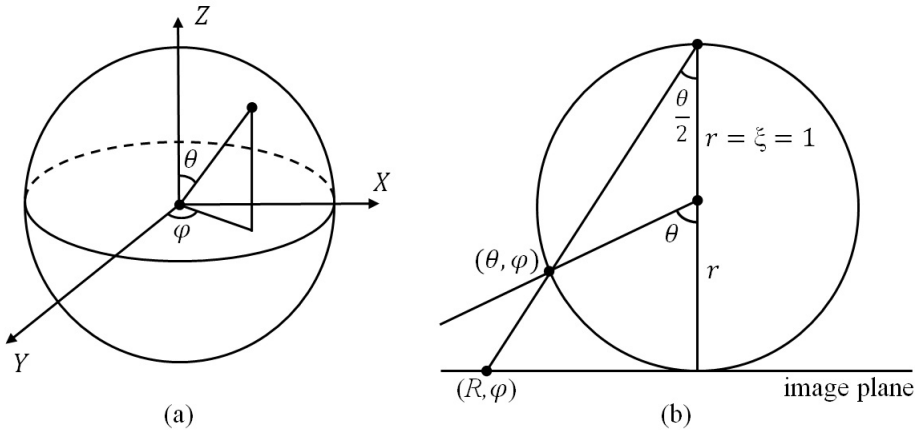


Figure 4.3. (a) A 3D point on the sphere is represented by two angles (θ, φ). (b) Consider the unitary sphere ($r = 1$). Image plane is placed at the south pole ($f = 2$). A 3D point is first projected onto the sphere surface and then projected onto the image plane, where in this case $\xi = 1$.

4.2.1.2. Differential Operators on Riemannian Manifolds

Let us briefly describe how the differential operators on the Riemannian manifolds are defined. Suppose *M* denotes a parametric surface on $\Re^3$ and g_{ij} denotes the Riemannian metric that encodes the geometrical properties of the manifold. In a local system of coordinates x^i on *M*, the components of the gradient are given by

$$\nabla^i = g^{ij} \frac{\partial}{\partial x^j} \tag{4.1}$$

where g^{ij} is the inverse of g_{ij}.

A similar reasoning is used in [3] and [41] to obtain the Laplace-Beltrami operator, which is the second order differential operator defined on and used for scale space representation for SIFT. In this paper, we are working on the first derivatives. Let us briefly go over the para-catadioptric case and derive the metric that allows us to compute the derivatives on the sphere directly using the image coordinates.

Consider the unitary sphere S^2 with radius = 1 (Fig. 4.3a). A point on S^2 is represented in Cartesian and polar coordinates as

$$(X, Y, Z) = (\sin\theta \sin\varphi, \sin\theta \cos\varphi, \cos\theta) \tag{4.2}$$

The Euclidean line element in Cartesian coordinates, dl, can be expressed in polar coordinates as

$$dl^2 = dX^2 + dY^2 + dZ^2 = d\theta^2 + \sin^2\theta d\varphi^2 \tag{4.3}$$

The stereographic projection of the sphere model sends a point on the sphere (θ, φ) to a point in polar coordinates (R, φ) in the image plane (plane $\Re^2$), for which φ remains the same and $\theta = 2\tan^{-1}(R/2)$ in a para-catadioptric system (Fig. 4.3b).

Using the identities, $R = x^2 + y^2$, $\varphi = \tan^{-1}(y/x)$ the line element reads

$$dl^2 = \frac{16}{(4 + x^2 + y^2)^2}(dx^2 + dy^2) \tag{4.4}$$

giving the Riemannian inverse metric

$$g^{ij} = \frac{(4 + x^2 + y^2)^2}{16} \tag{4.5}$$

We refer the reader to [3] and [5] for a detailed derivation of catadioptric Riemannian metric. With this metric, we can compute the differential operators on the sphere using the pixels in the omnidirectional images. In particular, norm of the gradient reads

$$|\nabla_{S^2} I|^2 = \frac{(4 + x^2 + y^2)^2}{16}|\nabla_{\Re^2} I|^2 \tag{4.6}$$

We see that the para-catadioptric gradients are just the scaled versions of the gradients in Euclidean domain. Therefore, we simply multiply our gradients with metric g^{ij}.

At the center of the omnidirectional image, $(x, y) = (0, 0)$, Riemannian and Euclidean gradients are the same. At an image location where $\sqrt{x^2 + y^2} = 2$, which corresponds to a 3D point at the same horizontal level with the sphere center (mirror focal point), the Riemannian metric is equal to 4. Therefore the gradients are magnified as we move from the center to the periphery of the omnidirectional image. This metric was extended to all central catadioptric systems by Puig et al. [41].

4.2.2. Conversion of Gradients for Omnidirectional Sliding Window

After the image gradients are obtained with Riemannian metric, we convert the gradient orientations to form an omnidirectional (non-rectangular) sliding window. A rectangular object in a perspective image is warped in the omnidirectional image, therefore the gradients in the sliding window should be computed as if a perspective camera is looking from the same viewpoint.

Figure 4.4. Two cars in the omnidirectional image are indicated with black frames. The one close to the camera covers a larger area and it should be searched with a more bent sliding window, the other one is far away and it should be search with a more straight sliding window

The shape of the omnidirectional sliding window varies according to the size and location of the object in the omnidirectional image. As depicted in Fig. 4.4, a car close to the camera is severely bent. However, a window covering the car at a distance is close to a rectangle. The difference can not be represented with a scale ratio, therefore we are not able to train one object model for detection in omnidirectional images. Since, it did not seem plausible to train many omnidirectional HOG models, we chose to train our object models with perspective image datasets. The modifications we made for HOG computation in omnidirectional sliding window enables us to compare it with the perspective camera HOG model.

Fig. 4.5a shows a half of a synthetic para-catadioptric omnidirectional image (400x400

pixels) where the walls of a room are covered with rectangular black and white tiles. Conventional HOG result of the marked region (128x196 pixels) in this image is given in Fig. 4.5b where gradient orientations are in accordance with the image. However, since these are vertical and horizontal edges in real world, we need to obtain vertical and horizontal gradients. Fig. 4.5d shows converted gradients for the region marked in Fig. 4.5c, which is an example of the proposed HOG computation.

Since the Cartesian coordinates in the detection window (Fig. 4.5d) corresponds to a non-linear distribution of pixels in the image (Fig. 4.5c), we employ bilinear interpolation with backward mapping both for gradient orientations and gradient magnitudes.

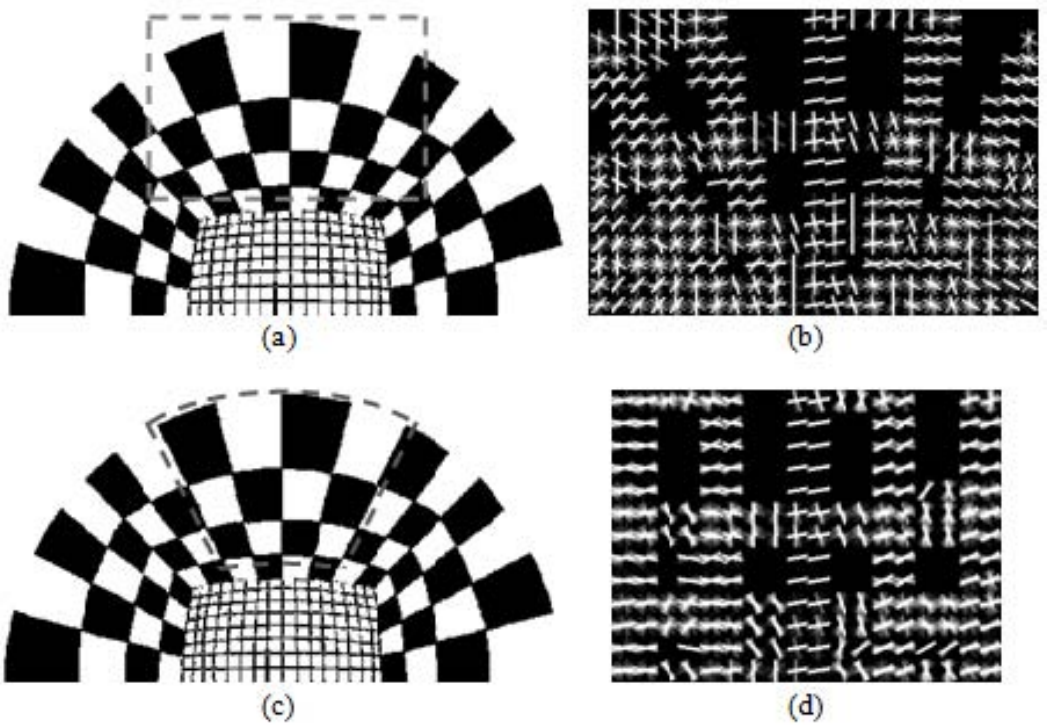


Figure 4.5. Description of how the gradients are modified for an omnidirectional sliding window. Result in (b) is the regular HOG computed for the region marked with dashed lines in (a). Modified HOG computation gives the result in (d) for the region marked in (c). Vertical and horizontal edges in real world produce vertical and horizontal gradients in the modified version.

CHAPTER 5

EXPERIMENTAL RESULTS

In this chapter we try to compare the performance of the proposed method (omni-directional sliding window) with the regular method on our omnidirectional and panoramic image set. First, implementation details that we applied in this thesis are introduced in the following paragraphs.

Our experiments consider the detection of standing humans and cars viewed from either side. The same classifiers are used as mentioned in section 3.2.3, for human detection we trained a 128x64 model using INRIA person dataset as described in [11], for car detection we trained a 40x100 model using UIUC [1] and Darmstadt [36] sets together totalling 602 car side views.

While training both object models, as defined in detail in section 3.2, the number of the negative samples in the dataset were increased by collecting so-called 'hard-negatives'. These are the false-positive detections of the initial model that was trained with the original positive and negative samples of the dataset. Therefore, second stage training with this negative set, improved both in terms of number and quality, is used for the final classification.

Also, before concentrate on experimental details, we will give a brief explanation on used software and equipment. In this study, all implementations are conducted with MATLAB R2012b. In addition for enhancing the running time performance, MATLAB allows us to rewrite the any time consuming part of our MATLAB methods with using C / C++ codes. This compilation work is conducted by MATLAB and the compiled file is named as Mex file. Hence, some extraction and modification parts of our MATLAB code is performed by Mexing. From the side of used equipment, omnidirectional images are captured by Canon G6 with a convex parabolic mirror apparatus [1].

[1] http://0-360.com/

5.1. A Fair Annotation and Overlapping Computation for Omnidirectional Sliding Approach

Within the scope of working with omnidirectional images, there is a necessary arrangement that is especially related with annotation generation and overlap computation of proposed sliding windows.

5.1.1. Annotation Generation

We have two types of test images; the first one is the omnidirectional image that is used for proposed (omnidirectional rotating sliding window) and regular (rectangular sliding window) method and the other one is panoramic images which is used for converting into panoramic from omnidirectional method. A panoramic image and an omnidirectional image have a different sliding window framework. Because of their peculiar shape; omnidirectional image is scanned in a circular pattern, whereas a panoramic image can be scanned in a rectangular (horizontal / vertical) pattern. The default sliding window detector for regular method, introduced in section 3.2.2, can be applied on panoramic image. Although we can represent the panoramic sliding window detection in a standard integrated form $[x_i, y_i, s_i, w_i]$ because of its rectangular shape, the same representation is not suitable for omnidirectional one. Moreover, we need a doughnut slice shape detection representation form like $[er_i, ma_i, s_i, w_i]$. Where w_i and s_i correspond to same parameter like being in the standard integrated form; differently er_i and ma_i correspond the *end-radius* value and *middle-angle* value of i-th omnidirectional sliding detection respectively. An example is given in Fig. 5.1. Radius values are in pixels and give the distance from image center coordinate, the start radius value is calculated from the end radius value. Angles are in radians ranging from 0 to 2π, also start angle and end angle is derived from middle angle value which divides the omni sliding window into equal two parts vertically.

After defining the omnidirectional sliding window representation, we have to use this formation while generating the annotation for omnidirectional images. We need to create three kinds of annotation corresponding to the mentioned three methods.

Annotations of the proposed HOG approach (e.g. Fig. 5.4 a) are doughnut slices, annotations of the regular HOG approach are rectangles rotating around the omnidirectional

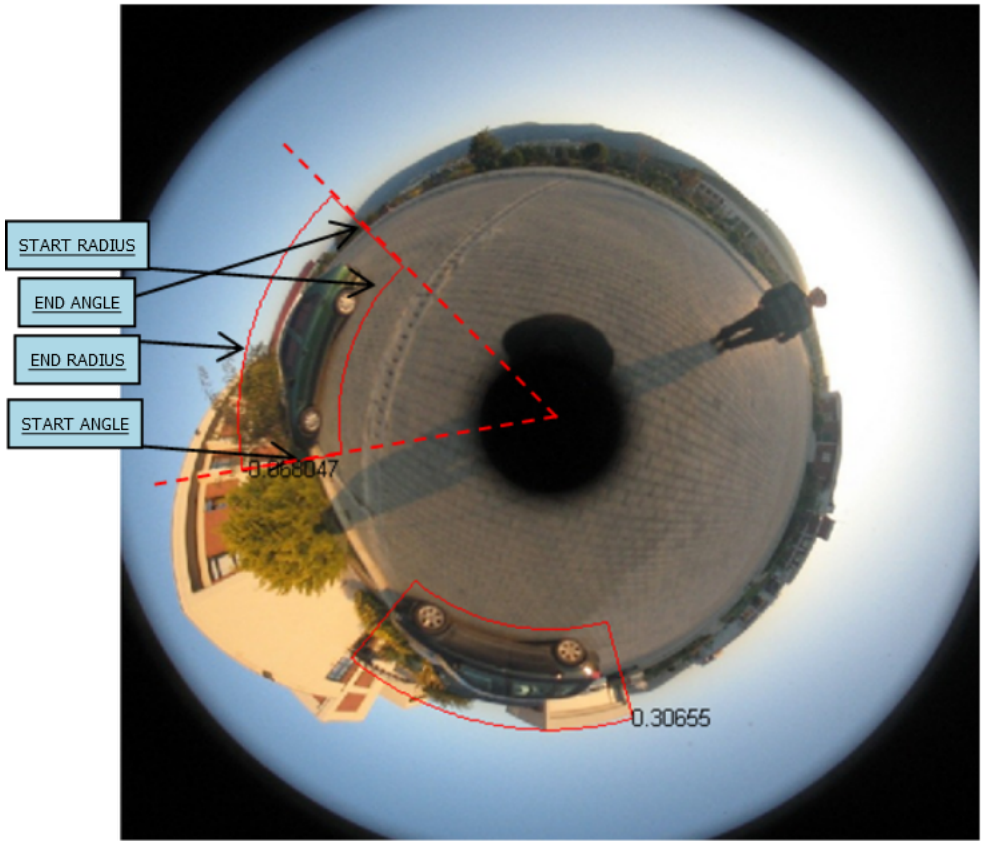


Figure 5.1. On this omnidirectional image sample, dashed line shows the angles and the other lines denote the radius. This representation shows our omnidirectional sliding window form.

image center. Finally, annotations of the HOG on panoramic image approach are upright rectangles.

Creating these annotations manually for each method will cause inequality for the same object instance. For a fair comparison, the annotations are separately prepared from a template annotation. We just generate the omnidirectional sliding window form annotation for proposed HOG approach on an omnidirectional image as a template; and then owing to a mathematically conversion process, annotation of the regular HOG on the same omnidirectional image and annotation of the HOG on panoramic image is derived from the previously determined template. Another important point stands out while determining this template annotation. We should prepare this annotation according to our positive training dataset; we left a similar space from left, right, bottom and top side of the objects.

Also we should underline that, for providing a fair scanning area on both panoramic

and omnidirectional images; we generate our omnidirectional image in size of [750 x 750] and convert them into our panoramic images in size of [645 x 257]. By this sizing we equalize the radius of an omnidirectional image with a height of a panoramic image, later on our detectors scanning equal amount of area on the both types of images.

5.1.2. Overlap Computation of Proposed Sliding Window

We defined the evaluation methodology and NMS for regular sliding windows in Section 3.2.4.2, but these are suitable methods for detection with rectangle sliding window. We need to define a new overlap computation method for the proposed sliding window that rotates around the omnidirectional image center. Firstly we had concentrated on a trapezoidal shape but it did not give an accurate area computation because our formation has a curved shape. Based on this reason, we assume an omnidirectional image like a circle and assume an omnidirectional sliding detection like a slice of this circle.

Therefore, we calculate an omnidirectional sliding detection area by the area computation formula of circle: $A = \pi \times r^2$. In this formula A denotes the whole area of omnidirectional image and r is the radius value of this image. We calculate the desired detection area by dividing A into small parts according to *mid-angle* and *end- radius* information of detections. After area computation technique is determined, how we give a decision whether these omnidirectional rotating shapes intersect or not. With the same principle in intersection of rectangular shapes (sec. 3.2.4.2); instead of maximum coordinate values of left-top corners maximum radian value of the *start-angles* gives the intersecting area's left boundary, in a similar manner instead of minimum coordinate value of right-bottom corner we look into minimum *end-angle* value for right boundary. This technique is suitable for cases like intersection of *BBox3* with *BBox4* or *BBox6* with *BBox7* in figure 5.2. However if we apply same technique on intersection case of *BBox1* with *BBox2* or *BBox5* with *BBox7*, a computational trouble is occurred. This wrong intersection area computation results from the base line depicted with dashed line in this figure, because start point (0)is same with the end point (2π) in a circle geometry. Normally, *end-angle* of a BBox must be greater than its *start-angle*, but if it was on the base line there is an opposite situation. So, we first determine if one of the compared BBox is on the base line, if it is on the baseline then we determine the position of the other compared one. According to its position add 2π to the *end-angle* or *start-angle* value. After this addition operation, we reach the accurate covered angle of intersection area

by subtracting the *start-angle* from *end-angle*. Finally we check the intersection area which can not exceed any of the compared BBoxes. If the intersection area was greater than any of the two compared BBoxes, it means there is no intersection between these BBoxes. Codes about NMS algorithm for omnidirectional sliding windows is presented in Appendix C.

We test our overlapping methodology by synthetic cases like depicted in Fig. 5.2 and so on, for this scenario the lower scored ones *BBox5* and *BBox4* are suppressed in NMS process. Consequently, thanks to the explained steps of computation, we reach a fair area computation.

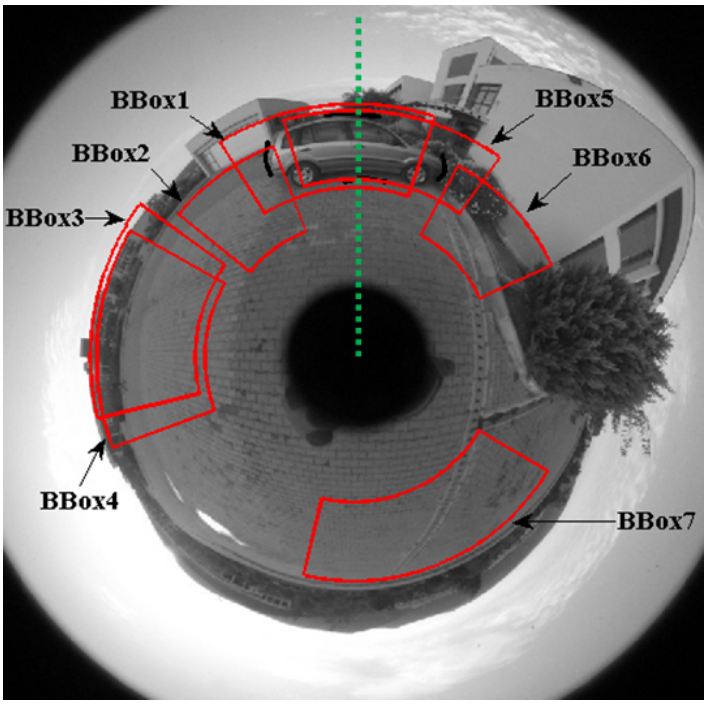


Figure 5.2. BBoxes in this scenario denote the omnidirectional sliding windows, each intersection between them creates different cases that help to evaluate the proposed overlapping computation. This computations is conducted depends on the dashed base line.

5.2. Evaluation of the Proposed HOG Computation Using Synthetic Omnidirectional Images

Let us first compare the results of the proposed HOG computation and the regular (unmodified) HOG computation. Since the computed HOG features are given to an SVM trained with an image dataset of corresponding object type, we aim to obtain higher SVM scores with the proposed omnidirectional HOG computation.

5.2.1. Experiment with Synthetic Images of Humans

We artificially created 210 omnidirectional images containing humans. While creating this set, we followed an approach similar to [25], where images in INRIA person dataset are projected to omnidirectional images using certain projection angle and distance parameters. Fig. 5.3 shows an example omnidirectional image, where the regular HOG window (rectangular, 128x64 pixels) and the proposed omnidirectional HOG window (non-rectangular) are shown. The HOG features computed with the two window types are compared with their resultant SVM scores. Since the locations of projections in these images are known, no search is needed for this experiment. However, vertical position of the window affects the result. For both approaches, we chose the position that gives the highest mean SVM score. Table 5.1 summarizes the result of the comparison, where we see that the mean score (also minimum, maximum and quartiles) for the proposed approach is higher than that of regular HOG window.

5.2.2. Experiment with Synthetic Images of Cars

For synthetic car images, same methodology in the previous subsection is used. 602 perspective car images from UIUC [1] and Darmstadt [36] datasets are projected to omnidirectional images.

40x100 pixel regular HOG computation and the proposed non-rectangular HOG window are applied on artificial omnidirectional images. Using the vertical positions that give the highest SVM scores, the two window types are compared in Table 5.2. The result is in accordance with the human detection experiment: mean SVM score, together with minimum,

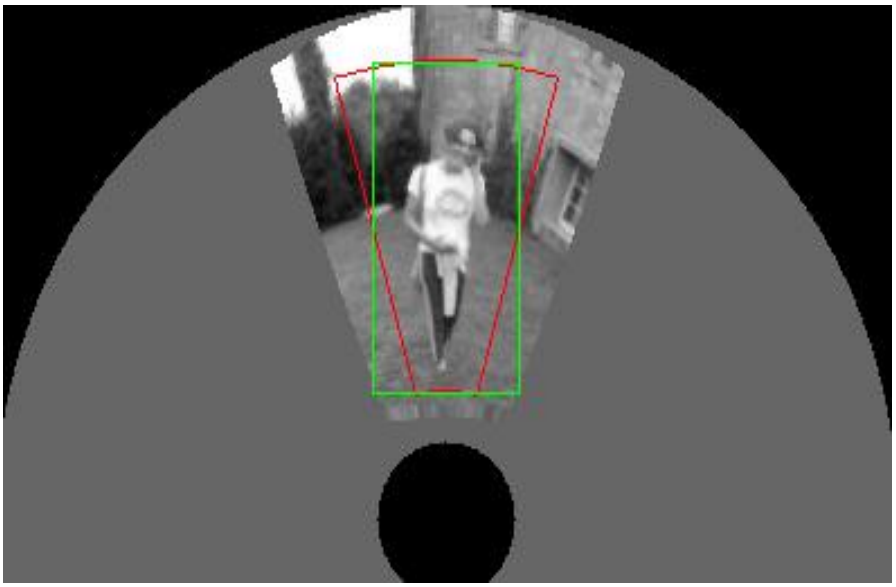


Figure 5.3. Depiction of the regular HOG window (green rectangle) and the proposed window (red doughnut slice) on an omnidirectional image artificially created by projecting a perspective image from INRIA person dataset.

maximum and lower/upper quartiles, for the proposed approach is higher than the regular method.

5.3. Experiments with Real Omnidirectional Images Containing Humans

In this subsection, we present the results for a set of images taken with our catadioptric omnidirectional camera which uses a paraboloidal mirror. We compared the proposed HOG computation not only with the regular HOG window, but also with the approach that first converts the omnidirectional image to a panoramic image and then performs regular HOG computation. Although it was explained in Section 4.1 that working on panoramic images is not a theoretically correct approach, we wanted to test its performance. Warping to a panoramic image takes less than a second if the image size is not larger than one mega-pixel (would be faster if conversion process is embedded in the camera), therefore if the performance of detection on panoramic image is higher it can still be considered as an option for

Table 5.1. Comparison of the regular and proposed HOG window by their SVM scores for human detection

	Min. SVM score	Lower quartile	Mean SVM score	Upper quartile	Max. SVM score
Regular HOG window	-1.01	1.16	1.69	2.20	3.21
Proposed HOG window	-0.42	1.51	1.93	2.45	3.64

Table 5.2. Comparison of the regular and proposed HOG window by their SVM scores for car detection

	Min. SVM score	Lower quartile	Mean SVM score	Upper quartile	Max. SVM score
Regular HOG window	-1.81	-0.38	-0.09	0.24	1.17
Proposed HOG window	-1.55	-0.17	0.19	0.55	1.79

practical applications.

Fig. 5.4 shows the results for one of the images in the set. Positive detections, after NMS, are superimposed on the images with the proposed HOG window, the regular HOG window on omnidirectional image and HOG after panoramic conversion. Only the detections with SVM scores greater than 1 are shown. The corresponding SVM score of each window is given at the upper left corner. For the humans in the scene, the average SVM scores for the proposed HOG, the regular HOG and HOG on panoramic image approaches are 2.94, 2.11 and 2.41 respectively.

To evaluate the overall performance of these three approaches, we plot precision-recall curves explained in sec. 3.2.4.3 for the whole dataset which consists of 30 real om-

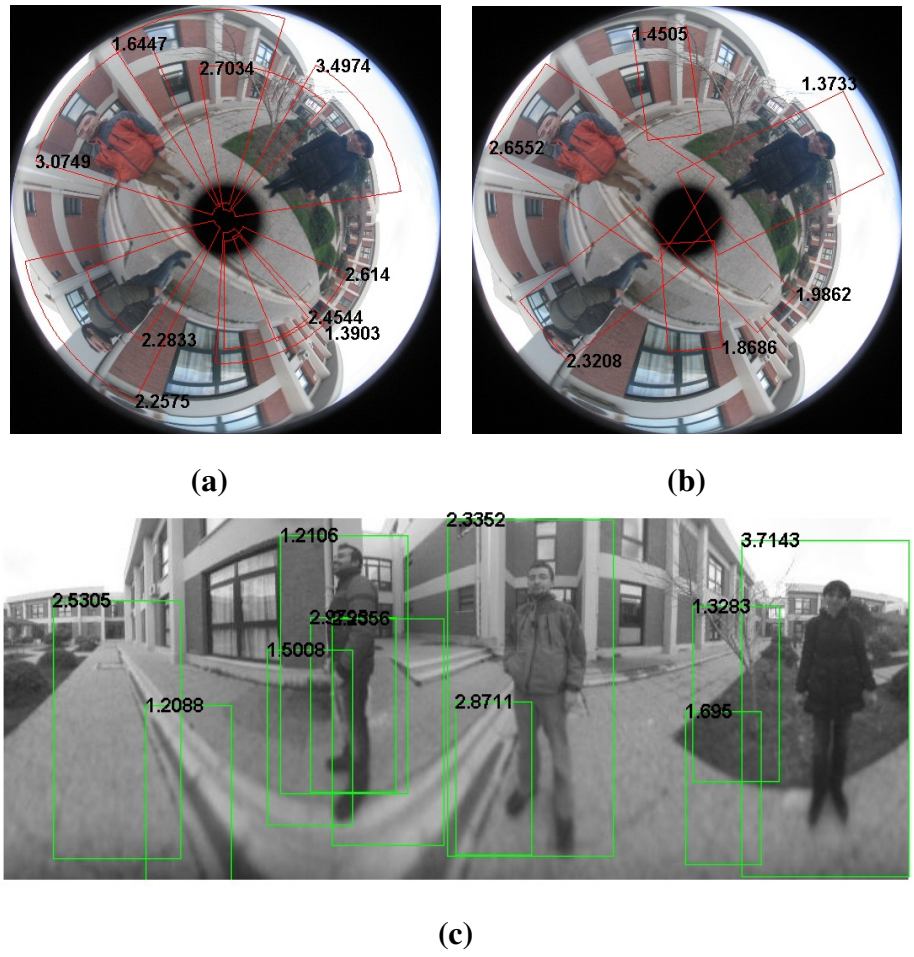


Figure 5.4. Human detection results on an omnidirectional image with SVM scores (given at upper left corners) greater than 1. (a) Proposed sliding windows. (b) Regular (rectangular) sliding and rotating windows. (c) Regular sliding windows on panoramic image.

nidirectional images taken in different scenes including indoor and outdoor environments (Fig. 5.5). A total of 66 humans were annotated in these images. One can observe that the performance of the proposed HOG computation is better than the others. Only for a limited range regular HOG approach performs better, however that is for low recall values. When recall >0.5, the proposed approach is distinctively superior.

A detection window is considered to be a True-positive if it overlaps an annotation by 50%, where the fair overlap is computed as explained in sec. 5.1.

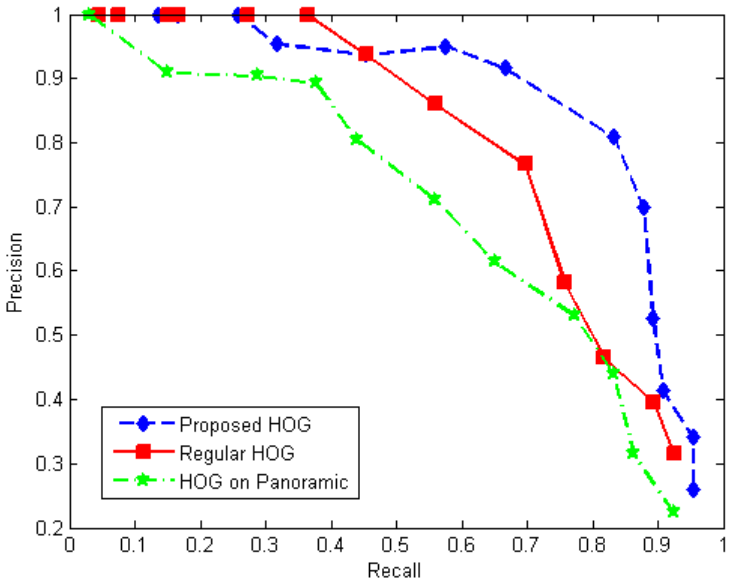


Figure 5.5. Precision-Recall curves to compare the proposed HOG computation, the regular HOG and HOG after panoramic conversion approaches for human detection. The data points in the curve correspond to the varying threshold values for the SVM score, which change from 0 to 5. As the threshold increases, all approaches reach Precision = 1.

5.4. Experiments with Real Omnidirectional Images Containing Cars

After carrying out experiment on images with humans in previous section, we repeated the comparisons between the evaluated methods for car detection.

Fig. 5.6 shows the results for a single image as an example. Similar to Fig. 5.4, scores are superimposed on detections. For the overall performance comparison of the proposed HOG computation, the regular HOG computation and HOG after panoramic conversion approaches for car detection, we plot precision-recall curves (Fig. 5.7) for the our dataset that includes 50 real images taken in different scenes. A total of 65 cars were annotated in these images. Similar to the human detection experiment, overlap ratio was computed with Eq. 3.1 and annotations were prepared separately for the three mentioned methods.

When we compare the results in Fig. 5.7 with the ones in Fig. 5.5, one observation would be that now the proposed method is better than the regular HOG everywhere. This is due to the fact that car is a wider object than human. The regular HOG computation is

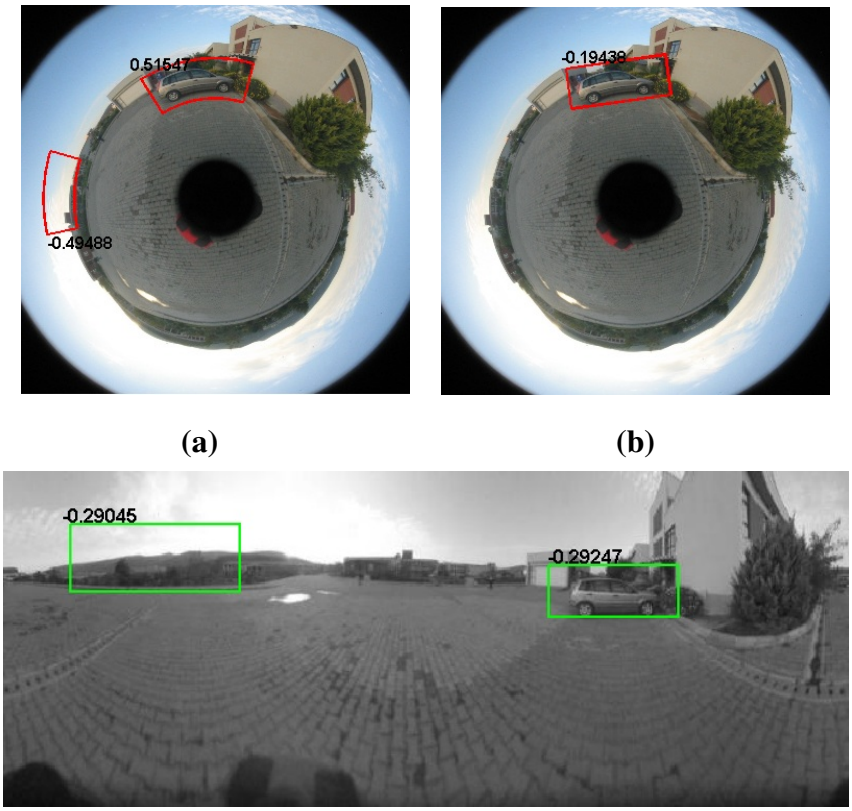


Figure 5.6. Results of car detection on an omnidirectional image with SVM scores (given at upper left corners) greater than -0.5. (a) Proposed sliding windows. (b) Regular (rectangular) sliding and rotating windows. (c) Regular sliding windows on panoramic image.

affected more as the width/height ratio of the object model increases because it tries to fit a rectangle to the object in the omnidirectional image, which is bent more.

A second major observation would be the increased performance of detection on panoramic image approach. Its performance is comparable to the proposed method as opposed to human detection results. This can also be explained by the fact that car has a 'wide' model with a width/height ratio of 2.5 whereas human has a width/height ratio of 0.5. It is harder for taller objects, like standing humans, to maintain the original width/height ratio in panoramic images since lower parts of the panoramic image suffer from an unnatural distortion. We create the panoramic image on a cylindrical surface that is rotating around the viewpoint (mirror focal point for catadioptric cameras). As we move down on the projection surface, same amount of viewing angle starts to cover a larger height in the panoramic image.

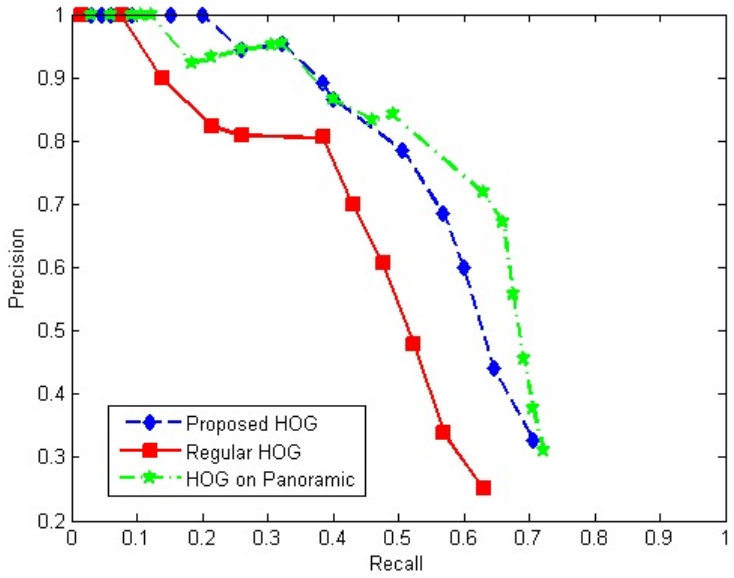


Figure 5.7. Precision-Recall curves to compare the proposed HOG computation, the regular HOG and HOG after panoramic conversion approaches for car detection. The data points in the curve correspond to the varying threshold values for the SVM score, which change from -1.0 to 1.5

This phenomenon is explained in Fig. 5.8 and can be observed in Fig. 5.4. Since the cars in panoramic images are affected significantly less than humans, regular HOG on panoramic images has a performance comparable to the proposed method for car detection.

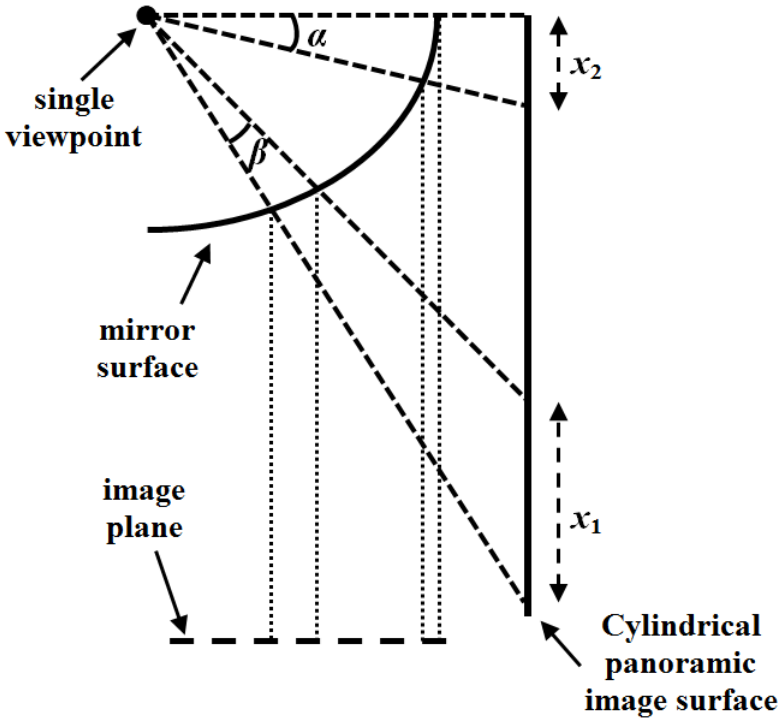


Figure 5.8. Formation of panoramic image on a cylindrical surface that is rotating around the single viewpoint (mirror focal point). As we move down on the projection surface, same amount of viewing angle starts to cover a larger height in the panoramic image ($\alpha = \beta$, $x_1 > x_2$).

CHAPTER 6

CONCLUSION

We have described a method to perform human and car detection directly on the omnidirectional images. As a base, we took the HOG + SVM approach which is one of the popular object detection methods and used the sliding window based object detection framework. In order to detect objects in omnidirectional images, we rotate the sliding window around the image center. In addition, to achieve a mathematically correct detection method, we modify the image gradients with two steps. Firstly we modify the gradient magnitudes using Riemannian metric then convert the gradient orientations to form an omnidirectional (non-rectangular) sliding window.

After describing how the feature extraction step of the conventional method should be modified, we performed experiments to compare the proposed method with the regular HOG computation on omnidirectional and panoramic images. Results of the experiments indicate that the performance of the proposed approach is superior to the regular HOG approaches for human detection. As for car detection, HOG on panoramic images has a performance comparable to the proposed method, whereas the performance of regular HOG on omnidirectional image has gone worse. Performance increase on panoramic images is explained by the fact that width/height ratio of the car model is high. This is an advantage for detection on panoramic image approach but a disadvantage for applying regular HOG on omnidirectional images.

We can conclude that the proposed HOG computation is not only the theoretically correct approach but also its performance is superior (considering that objects with a width / height ratio $>$2.5 are rarely of interest). Please also note that the regular HOG on panoramic image approach has an extra step of image conversion which can be considered as a disadvantage.

Finally we can say that despite the benefits of omnidirectional system, none of the encountered studies handled the omnidirectional imaging system with a feature extraction method for a direct classification on it. With our proposed method we try to overcome this deficiency in the literature. We believe our direct approach will be a useful contribution to the community for objects surveillance and omnidirectional vision based applications.

CHAPTER 7

FUTURE WORK

In this work, we concentrated on HOG features for human and car detection. However, other features, especially the ones based on image derivatives can be modified in a similar fashion for a theoretically correct and effective use in omnidirectional cameras. Moreover, part based object detection methods can be modified to be used with omnidirectional cameras. Detection of parts may not need a modification for omnidirectional images, since parts are smaller and relatively less deformed when compared to the whole object. However, the topographical relations between the part would require modification based on the imaging geometry of the camera. Also, rotation invariant HOG for object detection on omnidirectional images will be effective for this work.

In fact, best performing object detection techniques today do not use a single source to extract features. Therefore, using combination of multiple detection methods and combination of features from different sources such as first derivative, second derivative, color, edge etc. will contribute to the success of this work.

From the other side, the number of images in the used test dataset will be extended in future time. Furthermore this proposed approach will be adapted for other object classes like types of vehicles (bicycle, truck, bus etc.) or types of animals (cow, dog, cat etc.) for a more comprehensive study.

REFERENCES

[1] Agarwal, S., Roth, D., Learning a sparse representation for object detection, Proc. of the Seventh European Conference on Computer Vision, Part IV, 113-127, (2002).

[2] Amaral, F. H., Costa, A. H. R., Object Class Detection in Omnidirectional Images, Workshop de Visao Computacional (WVC), (2009).

[3] Arican, Z., Frossard, P., OMNISIFT: Scale Invariant Features in Omnidirectional Images, IEEE 17th International Conference on Image Processing (ICIP), (2010).

[4] Bastanlar, Y., Temizel, A., Yardimci, Y., Sturm, P., Multi-view Structure-from-Motion for Hybrid Camera Scenarios, Image and Vision Computing, vol.30(8), p.557?572, (2012).

[5] Bogdanova, I., Bresson, X., Thiran, J.P., Vandergheynst, P., Scale Space Analysis and Active Contours for Omnidirectional Images. IEEE Transactions on Image Processing, 16(7), 1888-1901, (2007).

[6] Bulow, T., Spherical diffusion for 3D surface smoothing, IEEE Trans. Pattern Anal. Mach. Intell. (PAMI), 25, 1650-1654, (2004).

[7] Chen, C.H., Yao, Y., Page, D., Abidi, B., Koschan, A., Abidi, M., Heterogeneous Fusion of Omnidirectional and PTZ Cameras for Multiple Object Tracking, IEEE Transactions on Circuits and Systems for Video Technology, 18(8), 1052-1063, (2008).

[8] Cheng, C. C., Chou, C. C., Hsueh, P. W., Detection of Moving Objects with a Mobile Omni-directional Camera, In Proceedings of International Conference on Software and Computer Applications (ICSCA), (2011).

[9] Cheng, S. Y., Trivedi, M. M., Lane Tracking with Omnidirectional Cameras: Algorithms and Evaluation, EURASIP Journal on Embedded Systems, (2007).

[10] Cinaroglu, I., Bastanlar, Y., A Direct Approach for Human Detection with Catadioptric Omnidirectional Cameras, IEEE Conference on Signal Processing and Communications

Applications (2014).

[11] Dalal, N., Triggs, B., Histograms of Oriented Gradients for Human Detection, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), (2005).

[12] Dalal, N., Finding people in images and videos, Doctoral dissertation, Institut National Polytechnique de Grenoble-INPG, (2006).

[13] Daniilidis, K., Makadia, A., Bulow, T., Image Processing in Catadioptric Planes: Spatiotemporal Derivatives and Optical Flow Computation, International Workshop on Omnidirectional Vision (OmniVis), (2002).

[14] Demiroz, B. E., Ari, I., Eroglu, O., Salah, A. A., Akarun, L., Feature-based tracking on a multi-omnidirectional camera dataset, InCommunications Control and Signal Processing (ISCCSP), 5th International Symposium on (pp. 1-5), IEEE, (2012).

[15] Dollar, P., Wojek, C., Schiele, B., Perona, P., Pedestrian Detection: An Evaluation of the State of the Art, IEEE Transactions on Pattern Analysis and Machine Intelligence, 34(4), 743-761, (2012).

[16] Everingham, M., et al., The 2005 PASCAL Visual Object Classes Challenge, In Selected Proceedings of the First PASCAL Challenges Workshop, LNAI, Springer-Verlag, 2006 (in press).

[17] Fei-Fei, L., Perona, P., A Bayesian Hierarchical Model for Learning Natural Scene Categories, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), (2005).

[18] Felzenszwalb, P., McAllester, D., Ramanan, D., A Discriminatively Trained, Multiscale, Deformable Part Model, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), (2008).

[19] Ferrari, V., Fevrier, L., Jurie, F., Schmid, C., Groups of Adjacent Contour Segments for Object Detection, IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI) , 30(1), 36-51, (2008).

[20] Gandhi, T., Trivedi, M. M., Video based surround vehicle detection, classification and logging from moving platforms: Issues and approaches, In Intelligent Vehicles Symposium, IEEE (pp. 1067-1071), (2007).

[21] Gamallo, C., Mucientes, M., V Regueiro, C., A FastSLAM-based Algorithm for Omnidirectional Cameras, Journal of Physical Agents, 7(1), 12-21, (2013).

[22] Geyer, C., Daniilidis, K., A unifying theory for central panoramic systems and practical applications, 6th European Conference on Computer Vision, 445-461, (2000).

[23] Gould, S., Fulton, R., Koller, D., Decomposing a Scene into Geometric and Semantically Consistent Regions, International Conference on Computer Vision (ICCV), (2009).

[24] Gupte, S., Masoud, O., Martin, R. F. K., Papanikolopoulos, N. P., Detection and Classification of Vehicles, IEEE Transactions on Intelligent Transportation Systems, 3(1), 37-47, (2002).

[25] Hansen, P. Corke, P., Boles, W., Daniilidis, K., Scale Invariant Features on the Sphere. IEEE International Conference on Computer Vision (ICCV), (2007).

[26] Herceg, D., Markovic, I., Petrovic, I., Real-time detection of moving objects by a mobile robot with an omnidirectional camera, In Image and Signal Processing and Analysis (ISPA), 7th International Symposium on (pp. 289-294), IEEE, (2011).

[27] Iraqui, A., Dupuis, Y., Boutteau, R., Ertaud, J., Savatier, X., Fusion of omnidirectional and PTZ cameras for face detection and tracking, International Conference on Emerging Security Technologies, (2010).

[28] Joachims, T., Making large-Scale SVM Learning Practical, Advances in Kernel Methods - Support Vector Learning, B. Schölkopf and C. Burges and A. Smola (ed.), MIT-Press, 1999.

[29] Kanhere, N. K., Birchfield, S. T., Real-Time Incremental Segmentation and Tracking of Vehicles at Low Camera Angles Using Stable Features, IEEE Transactions on Intelligent Transportation Systems, 9(1), 148-160, (2008).

[30] Kang, S., Roh, A., Nam, B., Hong, H., People detection method using GPUs for a mobile robot with an omnidirectional camera, Optical Engineering 50(12), 127204, (2011).

[31] Karaimer, C., Bastanlar, Y., Car Detection with Omnidirectional Cameras Using Haar-Like Features and Cascaded Boosting (in Turkish), IEEE Conference on Signal Processing and Communications Applications (2014).

[32] Ke, Y., Sukthankar, R., PCA-SIFT: A More Distinctive Representation for Local Image Descriptors, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), (2004).

[33] Khoshabeh, R., Gandhi, T., Trivedi, M. M., Multi-camera Based Traffic Flow Characterization and Classification, IEEE intelligent Transportation Systems Conference (IEEE-ITSC), (2007).

[34] Kumar, P., Ranganath, S., Weimin, H., Sengupta, K., Framework for Real-Time Behavior Interpretation from Traffic Video, IEEE Transactions on Intelligent Transportation Systems, 6(1), 43-53, (2005).

[35] Layerle, J. F., Savatier, X., Ertaud, J. Y., Mouaddib, E. M., Catadioptric Sensor for a Simultaneous Tracking of the Driver's Face and the Road Scene, The 8th Workshop on Omnidirectional Vision, Camera Networks and Non-classical Cameras-OMNIVIS, (2008).

[36] Leibe, B., Leonardis, A., Schiele, B., Combined object categorization and segmentation with an implicit shape model, Proc. of the Workshop on Statistical Learning in Computer Vision, (2004).

[37] Lourenco, M., Barreto, J.P., Vasconcelos, F., sRD-SIFT: Keypoint Detection and Matching in Images with Radial Distortion, IEEE Transactions on Robotics, 28(3), 752-760, (2012).

[38] Lowe, D., Distinctive image features from scale invariant keypoints, International Journal of Computer Vision (IJCV), 60, 91-110, (2004).

[39] Maji, S., Berg, A.C., Malik, J., Classification using Intersection Kernel Support Vector Machines is Efficient, IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), (2008).

[40] Morris, B., Trivedi, M., Robust Classification and Tracking of Vehicles in Traffic Video Streams, Intelligent Transportation Systems Conference (ITSC), (2006).

[41] Puig, L., Guerrero, J. J., Scale Space for Central Catadioptric Systems: Towards a Generic Camera Feature Extractor, International Conference on Computer Vision (ICCV), (2010).

[42] Sabzmeydani, P., Mori, G., Detecting pedestrians by learning shapelet features, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), (2007).

[43] Scharfenberger, C., Chakraborty, S., Farber, G., Robust Image Processing for an Omnidirectional Camera-based Smart Car Door, Proceedings of Embedded Systems for Real-Time Multimedia, (2009), pp: 106-115.

[44] Scotti, G., Marcenaro, L., Coelho, C., Selvaggi, F., Regazzoni, C.S., Dual Camera Intelligent Sensor for High Definition 360 Degrees Surveillance, IEE Proc.-Vis. Image Signal Processing, 152(2), 250-257, (2005).

[45] Shotton, J., Winn, J., Rother, C., Criminisi, A., TextonBoost: Joint Appearance, Shape and Context Modeling for Multi-Class Object Recognition and Segmentation, European Conference on Computer Vision (ECCV), (2006).

[46] Sivic, J., Russell, B.C., Efros, A., Zisserman, A., Freeman, W.T., Discovering Object Categories in Image Collections, International Conference on Computer Vision (ICCV), (2005).

[47] Suzuki, M., Blij, J. V., Floreano, D., Omnidirectional Active Vision for Evolutionary Car Driving, Proceedings of The Ninth International Conference on Intelligent Autonomous Systems, (2006), pp: 153-161.

[48] Tang, Y., Li, Y., Bai, T., Zhou, X., Human Tracking in Thermal Catadioptric Omnidirec-

tional Vision, International Conference on Information and Automation (ICIA), (2011).

[49] Torralba, A., Murphy, K. P., Freeman, W. T., Sharing Features: Efficient Boosting Procedures for Multiclass Object Detection, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), (2004).

[50] Vapnik, V. N., The Nature of Statistical Learning Theory, Springer, (1995).

[51] Walk, S., Majer, N., Schindler, K., Schiele, B., New Features and Insights for Pedestrian Detection, IEEE Conf. Computer Vision and Pattern Recognition, (2010).

[52] Wang, M.L., Lin, H.Y., Object Recognition from Omnidirectional Visual Sensing for Mobile Robot Applications, IEEE International Conference on Systems, Man and Cybernetics, (2009).

[53] Wu, B., Nevatia, R., Detection of multiple, partially occluded humans in a single image by bayesian combination of edgelet part detectors, IEEE International Conference on Computer Vision (ICCV), (2005).

APPENDIX A

SOFTWARE FOR DETECTION FRAMEWORK

```
img=imread(image_Path);

BinNum = 9;
Angle = 180;        % parameters for omnidirectional HOG feature
CellSize =4;
SkipStep=4;

dimension=size(img);
Hi=dimension(1);
img_r=Hi/2;

counter=0;
Sr=1.14;
Ss=1;
Se=min(img_r/100,img_r/40);         % computing the number of scale steps
Sn=floor(log(Se/Ss)/log(Sr)+1);

winnum=PreallocatingWinNumOmni_car(img);    % Preallocating the array(
Test_sumoffeat_Windows=zeros(winnum,7776); window number ) for omni
Windows_end_r=zeros(winnum,1);              sliding computation
Windows_mid_angle=zeros(winnum,1);
Windows_Scale=zeros(winnum,1);
Windows_Pathname=cell(winnum,1);

end_r_last=80;
for ScaleRatio=1:Sn          % resizing the image by Scale Ratio
    if ScaleRatio~=1
        imG=imresize(img,(1/1.14)^(ScaleRatio-1),'bilinear');
        End_r_last=end_r_last*((1/1.14)^(ScaleRatio-1));
    else
         imG=img;
         End_r_last= end_r_last;
    end

%% estimation of how far away the sliding window moves in horizantal and
vertical direction.

dimension=size(imG);
r=dimension(1)/2;       % finding radius

strd_r=8;               % move by 8 px stride, vertical
strd_angle=8;           % move by 8 degree angle, horizontal
iy=floor((r-40-End_r_last)/strd_r+1);       % vertical sliding numbers by
                                              radius.
x=0:strd_angle:360      % horizontal sliding number by angle
mid_angle_number=x;
```

Figure A.1. Computation of the number of S_n (scales steps) and the number of window stride for omni directional car images

```
img=imread(image_Path);

BinNum = 9;
Angle = 180;    % parameters for regular HOG feature of human image
CellSize =8;
SkipStep=8;
filter_para=[0;0];
filter_hsize=1;
filter_delta=filter_para(2);
dimension=size(img);
Hi=dimension(1);
Wi=dimension(2);

counter=0;
Sr=1.14;
Ss=1;
Se=min(Wi/64,Hi/128);           % computing the number of scale steps
Sn=floor(log(Se/Ss)/log(Sr)+1);

winnum=PreallocatingWinNum(image_Path);    % Preallocating the array(
Test_sumoffeat_Windows=zeros(winnum,3780);   window number) depens on
Windows_Y=zeros(winnum,1);                   human descriptor
Windows_X=zeros(winnum,1);
Windows_Scale=zeros(winnum,1);
Windows_Pathname=cell(winnum,1);

for ScaleRatio=1:Sn
    if ScaleRatio~=1
        imG=imresize(img,(1/Sr)^(ScaleRatio-1),'bilinear');
    else
         imG=img;
    end

%% Sliding window in dimension of [128*64]
dimension=size(imG)
heig=dimension(1); %128 pixel
wid=dimension(2);  %64  pixel

strd=8; % move by 8 pixel stride in both vertical and horizontal
          directions because of the rectangular shape of panoramic image

iy=floor((heig-128)/strd+1);  % vertical sliding number
ix=floor((wid-64)/strd+1);    % horizontal sliding number by angle
```

Figure A.2. Computation of the number of S_n (scales steps) and the number of window stride for panoramic human images.

APPENDIX B

SOFTWARE FOR LEARNING PHASE

```
%---Positive Training İmages-----------------------------------------------
path_pos_train='C:\Users\yalin
bastanlar\Desktop\INRIAPerson\INRIAPerson\train_64x128_H96\pos';
dirData=dir(path_pos_train);

fileNames = {dirData(1:502).name};

  Training_sumoffeat=zeros((numel(fileNames)-2),3780); %make code more
                                                        efficient
for iFile = 3:numel(fileNames)
    path_name=([path_pos_train,'\',fileNames{iFile}]);

    feat = ImgHOGFeature(path_name, SkipStep, BinNum, Angle, CellSize,
[0;0]);

    newfeat=reshape(feat,1,[]);   % convert [36*105] into [1*3780]
Training_sumoffeat(iFile-2,:)=newfeat;   % collect feature of each image
end

PosTrainImgNum=numel(fileNames)-2;   % using for creating training target

%---Negative Training Image ------------------------------------------------
path_neg_train='C:\Users\yalin
bastanlar\Desktop\INRIAPerson\INRIAPerson\Train\neg';
dirData2=dir(path_neg_train);

fileNames = {dirData2(1:1220).name};
   Training_sumoffeat(PosTrainImgNum+1:(PosTrainImgNum+(numel(fileNames)-
2)),:)=zeros((numel(fileNames)-2),3780);
for iFile = 3:numel(fileNames)

        path_name=([path_neg_train,'\',fileNames{iFile}]);
       feat = ImgHOGFeature(path_name, SkipStep, BinNum, Angle, CellSize,
[0;0]);

    newfeat=reshape(feat,1,[]);
    Training_sumoffeat((iFile-2)+PosTrainImgNum,:)=newfeat;
end
 NegTrainImgNum=numel(fileNames)-2;   % using for creating training target

%-------------generating training targets (Y) ----------------------------
for i=1:PosTrainImgNum     %for positive images target
    Y(i,:)=1;
end

for i=(PosTrainImgNum+1):(NegTrainImgNum+PosTrainImgNum)      %for
negative images target
    Y(i,:)=-1;
end
 %-----------------------training (svm)-----------------------------------

model = svmlearn(Training_sumoffeat,Y,'-c 0.5');
```

Figure B.1. Generation of human object model from the INRIA person dataset.

```
%--------------Positive Training İmages-----------------------------------
path_pos_train='C:\Users\yalin bastanlar\Desktop\HOG for Other Objects_
DATASET_Cars\DATABASES\UIUC Image Database for Car
Detection\UIUC\PNGImages\TrainImages';
dirData=dir(path_pos_train);
fileNames = {dirData(503:end).name};

Training_sumoffeat=zeros(numel(fileNames),7776);
for iFile = 1:numel(fileNames)
    path_name=([path_pos_train,'\',fileNames{iFile}]);
    feat = ImgHOGFeature(path_name, SkipStep, BinNum, Angle, CellSize,
[0;0],4);

    newfeat=reshape(feat,1,[]);
    Training_sumoffeat(iFile,:)=newfeat;
end
 PosTrainImgNum=numel(fileNames);

%--Negative Training (with randomly taking 10 window part from each images)
path_neg_train='C:\Users\yalin bastanlar\Desktop\HOG for Other Objects_
DATASET_Cars\DATABASES\UIUC Image Database for Car
Detection\UIUC\PNGImages\TrainCarNeg_40_100';
dirData2=dir(path_neg_train);
fileNames = {dirData2(3:end).name}; olması icin

Training_sumoffeat(PosTrainImgNum+1:(PosTrainImgNum+numel(fileNames)*10),:)
=zeros(numel(fileNames)*10,7776);
for iFile = 1:numel(fileNames)
        path_name=([path_neg_train,'\',fileNames{iFile}]);

    for rastgele=1:10
    katsayi=(iFile-1)*10;
    feat = ImgHOGFeature(path_name, SkipStep, BinNum, Angle, CellSize,
[0;0],5);
    newfeat=reshape(feat,1,[]);
    Training_sumoffeat(PosTrainImgNum+katsayi+rastgele,:)=newfeat;
    end
end
 NegTrainImgNum=numel(fileNames)*10;   % using for creating training target

boyut=(NegTrainImgNum+PosTrainImgNum);

%***************** For HARD NEGATİVES*********************************
(Adding hard negative features to Training_sumoffeat)
load thirdPhaseFeats_UIUC;
aralik=length(SecondPhaseFeats(:,2));
Training_sumoffeat(boyut+1:boyut+aralik,:)=SecondPhaseFeats;
%*****************************************************************
%% -------generating training targets (Y) ------------------------

for i=1:PosTrainImgNum     %for positive images target
    Y(i,:)=1;
end

for i=(PosTrainImgNum+1):(boyut+aralik) %for negative images target
    Y(i,:)=-1;
end
model = svmlearn(Training_sumoffeat,Y,'-c 0.5');
```

Figure B.2. Retraining of UIUC car object model by Hard Negative Examples.

APPENDIX C

SOFTWARE FOR MULTIPLE DETECTION LOCALIZATION

```
function pick = nms_Mine(scale_vektor,y_vektor,x_vektor,svm_result_vektor,
overlap,pthname)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%% İbrahim Çınaroğlu , ibrahim.ceinaroglu@gmail.com %%%
%%%%%%% İzmir Institute of Technology, İzmir, Turkey     %%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if isempty(y_vektor)
  pick = [];
else
    %% Converting values into x1,x2.... with scale estimation
size=numel(y_vektor);
x1=[size,1]; y1=[size,1]; x2=[size,1]; y2=[size,1];
for i=1:size
                if scale_vektor(i)~=1
                curr_scale=1.14^(scale_vektor(i)-1);
                else
                    curr_scale=1;
                end

x1(i,1)=  floor(x_vektor(i)*curr_scale);
y1(i,1)= floor(y_vektor(i)*curr_scale);
x2(i,1)= x1(i,1)+63*curr_scale;
y2(i,1)= y1(i,1)+127*curr_scale;
end
s = svm_result_vektor;

  area = (x2-x1+1) .* (y2-y1+1);

  [vals, I] = sort(s);  % ascending sort
  pick = [];
  while ~isempty(I)
    last = length(I);
    i = I(last);
    pick = [pick; i];
    suppress = [last];
    pathcontrol=pthname{i};
    for pos = 1:last-1
      j = I(pos);
      if 1~=strcmpi(pthname{j},pathcontrol) % control the image for true
image comparison.
          continue;
      end
      xx1 = max(x1(i), x1(j));
      yy1 = max(y1(i), y1(j));  %choose top-left coordinate as max, choose
      xx2 = min(x2(i), x2(j));   right-bottom coordinate as min for
      yy2 = min(y2(i), y2(j));   intersection area
      w = xx2-xx1+1;
      h = yy2-yy1+1;
      if w > 0 && h > 0
        o = w * h / area(j);     % compute overlap
        if o > overlap
          suppress = [suppress; pos];
        end
      end
    end
    I(suppress) = [];
  end
end
```

Figure C.1. Nms algorithm for regular sliding windows (rectangular Bboxes).

```
function
[pick,EndRadius,StartRadius,StartAngle,EndAngle,CoveredAngle,boxArea] =
nms_Mine_OmniProp_car_Donut(imgpath_vektor,scale_vektor,end_r_vektor,mid_an
gle_vektor,svm_result_vektor, overlap,ProposedResult)

if isempty(end_r_vektor)
  pick = [];
else
HOGwidth=100; HOGheight=40;

EndRadius=end_r_vektor; %the top of the sliding window is pixels away from
StartRadius=EndRadius-HOGheight;               the omni image center
MidAngle=mid_angle_vektor;

k=numel(end_r_vektor);
CenterRadius=zeros(k,1);   % preallocating
CoveredAngle=zeros(k,1);
ImageCenterx=zeros(k,1);
ImageCentery=zeros(k,1);

for i=1:k
    if scale_vektor(i)~=1
           curr_scale=1.14^(scale_vektor(i)-1);
    else
           curr_scale=1;
    end

    ben=imread(imgpath_vektor{i});
    [height width rgb]=size(ben);

    EndRadius(i)=curr_scale*EndRadius(i);
    StartRadius(i)=StartRadius(i)*curr_scale;
    CenterRadius(i)=EndRadius(i)-HOGheight*curr_scale/2;
    CoveredAngle(i)=360*HOGwidth*curr_scale/(2*pi*CenterRadius(i));
    ImageCenterx(i)=width/2;
    ImageCentery(i)=height/2;

end
StartAngle=MidAngle-CoveredAngle/2;  % in degree, never be >360
EndAngle=MidAngle+CoveredAngle/2;    %in degree, never be negative
for r=1:k
 if StartAngle(r)<=0  StartAngle(r)=StartAngle(r)+360; end
 if EndAngle(r)>=360  EndAngle(r)=EndAngle(r)-360;  end
end
%% control each window whether it is on baseline or not(0 degree)
    Zonedifference=(2*pi*(EndRadius.^2))-(2*pi*(StartRadius.^2));
    boxArea=(Zonedifference.*CoveredAngle)/360;
    s=svm_result_vektor;

[vals, I] = sort(s);  % ascending sort
  pick = [];
  while ~isempty(I)
    last = length(I);
    i = I(last);
    pick = [pick; i];
    suppress = [last];
    for pos = 1:last-1
      j = I(pos);
```

Figure C.2. Nms algorithm for omnidirectional rotation sliding windows (doughnut slice Bboxes) (cont. on next page).

```
%% start angle checked & end angle checked , if in 0-360 degree zone

if       StartAngle(i)> EndAngle(i) % if first one is over baseline
    if   StartAngle(j)< EndAngle(j)  and second one is not over baseline
        if StartAngle(j) < 180
StartAngle(i)=StartAngle(i)-360;
        elseif EndAngle(j) > 180
            EndAngle(i)=EndAngle(i)+360;
        end
    end
else
    if   StartAngle(j)> EndAngle(j)
        if StartAngle(i) < 180
        StartAngle(j)=StartAngle(j)-360;

        elseif EndAngle(i)> 180
            EndAngle(j)=EndAngle(j)+360;  % for solving the complexity we
        end                               add 2 pi to start angle or end angle.
    end
end
      strtAngNew = max(StartAngle(i), StartAngle(j));
      strtRadNew = max(StartRadius(i), StartRadius(j));
      endRadNew = min(EndRadius(i), EndRadius(j));
      endAngNew = min(EndAngle(i), EndAngle(j));
%% obtaining original start and end angle values from negative and +360
version
for r=1:k
 if StartAngle(r)<=0  StartAngle(r)=StartAngle(r)+360; end
 if EndAngle(r)>=360  EndAngle(r)=EndAngle(r)-360;  end
end
      h = endRadNew-strtRadNew+1;

      if   h > 0 % radius kesin olarak buyuk olamalı, baska olasılık yok

          if (endAngNew > strtAngNew)        % compute overlap
              difference=(2*pi*(endRadNew^2))-(2*pi*(strtRadNew^2));
              AreaBboxDonutnew=(difference*(endAngNew-strtAngNew))/360;
              o = AreaBboxDonutnew / boxArea(j);
              if o > overlap
                  suppress = [suppress; pos];
              end
          end
          minCoverAngOfBbox=(min(CoveredAngle(i),CoveredAngle(j)))+1;

          if (endAngNew < strtAngNew)&& ((endAngNew+360)-strtAngNew)<=
(minCoverAngOfBbox)
              difference=(2*pi*(endRadNew^2))-(2*pi*(strtRadNew^2));
              AreaBboxDonutnew=(difference*((endAngNew+360)-
strtAngNew))/360;
              o = AreaBboxDonutnew / boxArea(j);
              if o > overlap
                  suppress = [suppress; pos];
              end
          end
      end
    end
    I(suppress) = [];
  end
end
```

Figure C.3. Nms algorithm for omnidirectional rotation sliding windows (doughnut slice Bboxes) (cont.).