UNIVERSITY OF SOUTHAMPTON

Person Detection using wide angle overhead cameras

by

Imran Ahmed

A thesis submitted in partial fulfillment for the degree of Doctor of Philosophy

in the

Faculty of Engineering and Applied Science

Department of Electronics and Computer Science

February 2014

UNIVERSITY OF SOUTHAMPTON

ABSTRACT

FACULTY OF ENGINEERING AND APPLIED SCIENCE
DEPARTMENT OF ELECTRONICS AND COMPUTER SCIENCE

Doctor of Philosophy

by Imran Ahmed

In cluttered environments, the overhead view is often preferred because looking down can afford better visibility and coverage. However detecting people in this or any other extreme view can be challenging as there are significant variation in a person's appearances depending only on their position in the picture. The Histogram of Oriented Gradient (HOG) algorithm, a standard algorithm for pedestrian detection, does not perform well here, especially where the image quality is poor. We show that with the SCOVIS dataset, on average, 9 false detections occur per image. We propose a new algorithm where transforming the image patch containing a person to remove positional dependency and then applying the HOG algorithm eliminates 98% of the spurious detections in the noisy images from our industrial assembly line and detects people with a 95% efficiency.

The algorithm is demonstrated as part of a simple but effective person tracking by detection system. This incorporates simple motion detection to highlight regions of the image that might contain people. These are then searched with our algorithm. This has been evaluated on a number of SCOVIS sequences and correctly tracks people approximately 99% of the time. By comparison, the exampled algorithms in the OpenCV are less than approximately 50% efficient.

Finally, we show our algorithm's potential for generalization across different scenes. We show that a classifier trained on the SCOVIS dataset achieves a detection rate of 96% when applied to new overhead data recorded at Southampton. Using the output from this stage to generate labelled 'true positives' data we train a new model which achieves a detection rate of 98%. Both these results compared favourably with the performance of a model trained with manually labelled images. This achieves a detection rate of greater than 99%.

DECLARATION OF AUTHORSHIP

I, **IMRAN AHMED**....., declare that the thesis

"**Person Detection using wide angle overhead cameras**"

and the work presented in the thesis are both my own, and have been generated by me as the result of my own original research. I confirm that:

- this work was done wholly or mainly while in candidature for a research degree at this University;
- where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated;
- where I have consulted the published work of others, this is always clearly attributed;
- where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work;
- I have acknowledged all main sources of help;
- where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself;
- none of this work has been published before submission, **or** [delete as appropriate] parts of this work have been published as: [please list references]

Signed: .....

Date:.....

FIGURE 1

Contents

List of Figures

List of Tables

Acknowledgements

I am indebted to my supervisor, Dr. John N Carter, for his undivided attention and most of all his patience in dealing with my circumstances throughout my PhD study. I truly admire his unique way of teaching and supervising as well as his friendly behaviour.

Regarding my PhD funding, I would like to extend my special thanks to the Institute of Management Sciences, Peshawar and the Higher Education Commission of the Pakistani Government for sponsoring my studies in the United Kingdom.

I also wish to acknowledge all people who helped me through this research either directly or indirectly including: Professor Mark S. Nixon and Dr. Sasan Mahmoodi for their valuable feedback during the mini-thesis examination, Miss Sophie for editing my PhD thesis. Thanks are also due to staff at University of Southampton, Lauren Dampier, Jane Kennard and Sonal Mehta for their supportive attitude and all people from the CSPC research group.

I am also thankful to my colleagues and friends who have made my time at Southampton an enjoyable and rewarding experience, including Haji M. Zia, Qari Bashir, Haji Khan Muhammad, Qari Hamayun, Qari Akhtar, Abdul Khaliq, Jasur Habibi, Mia Nadeem and Haji Mujahid.

Most of all, I would like to express my gratitude to almighty God, for allowing me to think and approach things positively during my study.

Last but not least, thanks to all of my family members and friends for their constant support.

Dedicated to my late father and brother.

Chapter 1

Introduction

In this research we have explored a new dimension for detecting people, this is the use of an overhead view. The main focus of this chapter is to explain how, due to a change of perspective, people looks radically different in an overhead view compared with how they look in a normal or conventional view. We also discuss the significance of using an overhead view to detect people particularly in an occluded environment where different conventional view based detection techniques might fail or not work adequately.

In Section 1.1 we briefly explain the possible applications areas of person detection. The main focus of Section 1.2 is on a new problem of detecting people. To show why it is a new problem, we compare it with existing approaches of person detection. The motivation of using an overhead view over a conventional view is been discussed. The difficulties and challenges associated with person detection are discussed in Section 1.3. In Section 1.4 we summarise our main contributions to this field of work. The structure of the thesis is discussed in the last section.

1.1 Applications of Person Detection

Human detection in images is one of the active and challenging areas of research in the field of computer vision and pattern recognition. It is important because of its potential applications in different areas. These applications also differ in their real-time requirements. For example, such an application can be found in driving assistant systems where a person detection system is incorporated into automobiles; and this system is then used to alert drivers to situations involving the presence of pedestrians on the road. Recently, such a vision-based collision warning system has been introduced in cars with full auto brake control [1]. The person detection system can be useful in surveillance to oversee an area and alert security or annotate the image. In this context, another field of application is behaviour analysis [15][26]. In the industrial environment, person

detection and tracking can be used for monitoring of industrial work-flows [9]. For a human-computer interaction where a robot is used, it is necessary for a robot to infer the correct position of a person or other objects in the scene in order to avoid collision. The human detection system can be useful in biometric systems where we can integrate the person detection system with face, nose or ear biometrics. In such a system the computer first scans the image for the detection of a person and then uses biometric information to extract the details about that person. Such systems can be installed at airports for identity verification.

1.2 Problem Statement

Many researchers have developed methods and techniques for detecting people in images. A recent good review about person detection methods has been discussed in Dollar et al.[34]. In all of these person detection methods horizontal or normal view images of people are used. Some of these types of images are shown in Figure 1.1.

(a) Facing to the front. (b) Facing to the right. (c) Facing to the rear. (d) Facing to the left.

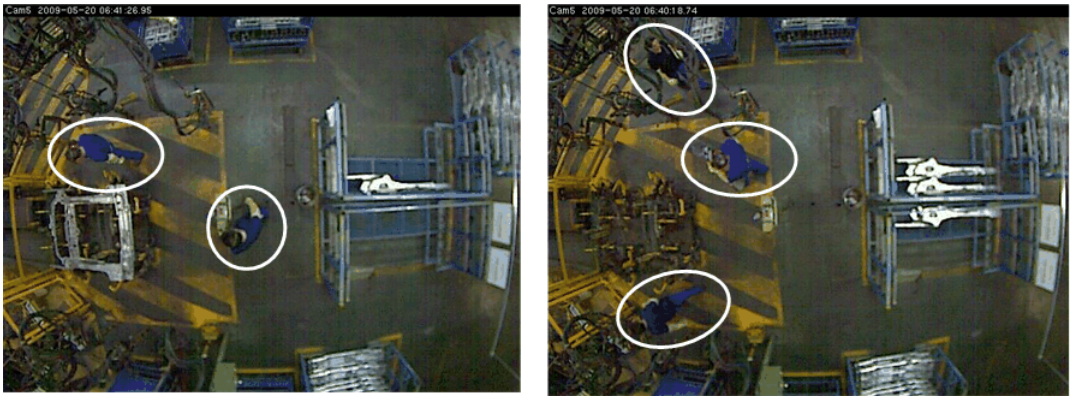
FIGURE 1.1: Normal View: A view parallel to the ground and a vertical pose. (from[29]). The central rotation of people along with vertical axis can be seen in these images while facing in different directions.

These images are taken from the INRIA [1] pedestrian dataset [29]. People in these images have a variety of poses and backgrounds which makes it more challenging than the previously used pedestrian benchmark, MIT [2] Pedestrian Database [89] . Although a great deal of work has been done to detect people in these kind of images there are still situations where these techniques might fail or do not work properly; one such example is shown in Figure 1.2(a). Here for the purpose of visualisation we use white ellipses to highlight the positions of people in the scene. This is a sample example in this occluded environment where we can see people partially. These people are occluded with different parts of racks and wires of machinery. Most of the time in this environment we are unable to see these people completely, particularly when they are working behind the heavy machinery and racks filled with steel rods. The overall environment is very cluttered and even for a manually monitoring system it is very hard to locate the person. In this type of situation we can take advantage of using an overhead view with a wide

[1] http://pascal.inrialpes.fr/data/human/
[2] http://cbcl.mit.edu/software-datasets/PedestrianData.html

angled camera; (See Figure 1.2(b)). Compare Figure 1.2(a) and Figure 1.2(b) taken at the same time but with two different cameras. Using an overhead camera with a wide angle lens we can not only clearly see the people who were occluded in the side view but we can also see much more in the scene compared to the normal view.

(a) Normal View: The people are shown in white ellipses which are occluded with machinery. Existing Person detection algorithms might fail or have a worst performance in this kind of images.



(b) Overhead View: The people are shown in white ellipses which can be seen easily.

FIGURE 1.2: Normal View vs Overhead View of the same scene at the same time but with different camera positions and perspectives.

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

FIGURE 1.3: Overhead View: Effect of rotation around vertical axis and movement of limbs. In each pair of images people are at the same position but facing to the different directions.

The person is a deformable object and has a great degree of variations in pose due to movement of limbs and rotation around vertical axis. These effects can be seen in Figure 1.1; for example starting from the first pair of images in Figure 1.1(a), a person is facing to the front which rotates gradually in the next pair of images, shown in Figure 1.1(b). In the third pair of images as shown in Figure 1.1(c) people are facing to the rear while

again the last pair in Figure 1.1(d) exhibits the side view property. In all of these images, articulation of different body parts can also be seen. These effects can also be seen if we view people from an overhead camera with a wide angle lens. In Figure 1.3, four pairs of images are shown from an overhead camera. In each pair the person is at the same position but with a rotation along the vertical axis. For example in the first pair of images (Figure 1.3(a)) the person is just below the camera in the same position but facing in a different direction. Similarly in the next three pairs of images the person is shown at the same position but with front and back view. The movement of different body parts can also be seen in these images.

In conventional imaginary, as in Figure 1.1, body rotation about the vertical axis and movement of body parts are the major causes of differences of appearance while perspective effects are negligible, manifesting themselves as simple size changes. Furthermore it is usually assumed that the person is mostly visible and is in an upright pose. These assumptions can be seen easily in these images. In scenes viewed from an unorthodox position, camera properties, such as field of view, can be a major cause of variation and may have a much greater effect than local rotation and even limb movement (see Figure 2.26).

This research concerns the use of an overhead camera to detect people in an industrial scene, shown in Figure 2.26. This is an example of an extreme viewpoint where the above assumptions about human appearance do not hold. It can be seen that the appearance of people is radically different in the two types of images.

(a) Just below the camera (b) Upward (c) Downward rear side (d) Downward front side

(e) Left side (f) Left diagonal (g) Right diagonal (h) Right side

FIGURE 1.4: Overhead view and perspective: Variation in size and orientation of people due to perspective change.

Perspective changes in the overhead view, have a much greater effects. This is because:-

- In an extreme or overhead view the appearance primarily depends on the position of the person relative to the optical centre of the image, as well as local rotation of the person about their vertical axis and limb movement.

- When directly under a camera a person's upper body obscures their legs. This is something that is not seen from a conventional or oblique views.

- As the person moves further away from the viewpoint, there is less self occlusion and the view of the body is elongated. Figure 2.26 shows some example images of people in this scenario. The left most image exhibiting self occlusion is taken from almost directly overhead. We can see only the head and shoulders of a person at this position. In the rest of images the person is shown in different positions away from the viewpoint and exhibits elongation effect as they are taken from off axis. In the right three images we can easily observe how arms are visible which are obscured in the first image. Overall, these images show how perspective changes effect the appearance of the people from the viewpoint.

- Another fact using the overhead view is that the orientation of the person also changes depending upon the position and radial distance from the optical centre of the image. For example, in Figure 2.26, the second and third images are taken from top and bottom of the scene respectively, where the orientation of the person is completely in the opposite direction. These effects can also be seen when the person is at left and right most edges of the scene, respectively as shown in Figure 2.26(e) and Figure 2.26(h). Similarly in Figure 2.26(f) and Figure 2.26(g) the person is shown at the same radial distance in left and right diagonal positions of the scene, respectively but facing to the opposite directions. This also means that, in using the overhead view in contrast to the normal view, it is very difficult to establish certain geometric configuration or association about different body parts of the person due to orientation change.

The motivation behind people detection from an overhead view (particularly in an occluded and cluttered environment) is that, we can see different objects of the scene more clearly and distinctly than from the front view. Figure 1.2(a) and Figure 1.2(b) show the two sample images from the SCOVIS database [36]. The two images are taken from the side and overhead view at the same time. In these images people are more visible in the overhead view than from a normal or horizontal view.

1.3 Challenges in Person Detection

In general, person detection is a difficult problem because the human body is a deformable object and can have a lot of variations in body poses. Moreover wearing different clothes, illumination, occlusion, view point variations, cluttered background and scale variations makes it a much more complex and difficult problem. Some of these associated challenges using an overhead view are described as follow:

1. The appearance of the human body is strongly influenced by the clothing they wear. Other sources of variations are body shape (slim or fat) and gender structure. Figure 2.26 and Figure 1.5(a) show people with different clothes. Moreover most of the workers' clothes have the same colour as parts of the surroundings.

2. Another challenge in person detection systems is body articulation. Variation in the body appearance is caused by different positions of organs of the body relative to each others. Some examples of body articulations can be seen in Figure 2.26; for example, in Figure 2.26(a) we can see mostly the head and shoulder of the person and it looks a square or circular object which makes it significantly different to the rest of the examples. In these examples different variations of poses due to the movement of legs, arms and other body parts can also be seen.

3. Shading or lighting variation has a major influence on the appearance of a person because of large variations in the values of pixels of the image. It is one of the fundamental and most common problems in computer vision systems. The illumination effect can be seen from two sample images in Figure 1.5. In our dataset there are rapid lighting changes due to machinery in operation and camera shake due to transportation of heavy machinery.

(a) Image with high illumination (Brightness) (b) Image with low illumination (Darkness)

FIGURE 1.5: Images with different illumination: The same object at the same position may be interpreted differently by the recognition system because of the brightness or darkness in the image.

4. The cluttered and dense background may cause confusion for the detection system while classifying background and foreground objects. People detection becomes more challenging particularly in an industrial environment [77, 110] due to sparks and vibrations in the scene because of using welding machine, dense structured background (e.g. different racks with heavy machinery), occlusions of workers in most of the positions in the image and moving objects (e.g forklifts and welding machines). One such example can be seen in Figure 1.6.

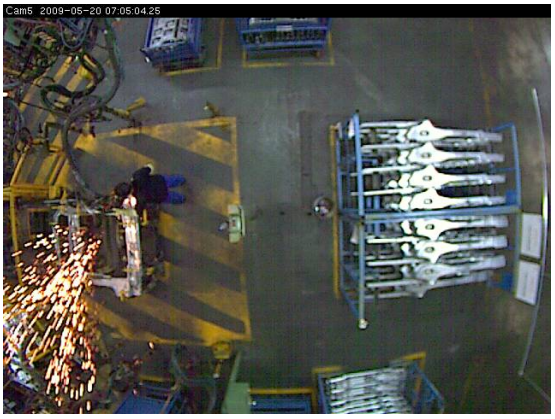


FIGURE 1.6: Cluttered background: Sparks in the background due to welding machine.

5. Another major difficulty in person detection system is occlusion. The partial object representation or missing body parts often result in poor recognition with a lot of false positives. In our overhead dataset people are performing different activities like carrying and/or handling different parts of the car, jointing these parts with welding machine, interacting with racks, and so on. During this most of the time the people were occluded by machinery, wires or racks. An example is shown in Figure 1.7 where people carrying different metallic parts of the car and the lower body are occluded by these rods. This this kind of occlusion makes the person detection process very difficult and challenging.

6. Another challenge is that a human can use high level context and the background information but the computer still lacks this property. Figure 1.8 highlights one such example. In this figure two regions of image are highlighted by drawing two labelled white boxes. The box labelled as 1 has an actual real person in it while box 2 does not contain any person. Humans use common sense and background information for reasoning to distinguish between a person and another object. For example we know already that a human has a vertical structure with an approximate average 2 feet width and 5 feet height. This kind of vertical structure could apply for certain other objects like a tree, a wooden plank or a tower, among many others. One such resembled object is shown as a white labelled box 2, in Figure 1.8. We can categorise this object as a human or non-human but a computer may find it hard including the algorithm we develop in this thesis, See Figure 4.22 of Chapter 4.

(a) The lower body of both people are occluded by a steel rod.

(b) The left most person is mostly invisible due to occlusion by rack and steel rod.

FIGURE 1.7: Two different situations of occlusion.

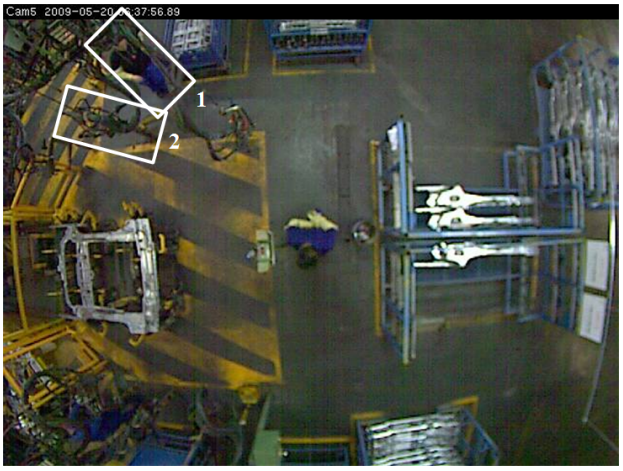


FIGURE 1.8: Context knowledge: Box 1 contains a person while box 2 is perceived as a human by the computer due to a lack of common sense knowledge.

1.4 Summary of Contributions

This thesis addresses the following three main contributions of my research work:

- My *first contribution* is that we have explored a new dimension of research in which the person is viewed from an overhead camera with a wide angle lens. Due to perspective change a person can have different orientations and sizes at different positions in the scene relative to the optical centre. We exploit this property of the overhead camera and develop a novel algorithm which uses the variable size bounding boxes with different orientations, with respect to the radial distance of the center of the image. In these overhead view images we neither used any assumption about the pose or the visibility of a person nor imposed any restriction about the environment. When compare the results of our algorithm with a standard Histogram of Oriented Gradient (HOG) algorithm, we achieve

not only a huge gain in overall detection rate but also a significant improvement in reducing spurious detections per image.

- My *second contribution* is to develop a simple people-tracking-by-detection algorithm to demonstrate how well our proposed person detection algorithm works. We use a simple motion-detection framework in our tracking algorithm. Instead of searching all positions of an image, a search for the existence of a person is made using only detected motion blob regions. The expected position of the person in the next frame is estimated using the history of the already detected object. We compare our proposed tracking algorithm with the existing four standard tracking algorithms of OpenCV. The results show that our algorithm outperforms all of these. The proposed algorithm has the additional advantage of handling big missing data gaps, occlusion and detecting a stationary person in the image.

- My *third* and exciting contribution is about the generalisation frame-work of person detection. We have shown experimentally that our algorithm trained with one set of overhead data can work perfectly in another completely different overhead dataset with very little manual intervention and produce remarkable results. We use a trained model with SCOVIS overhead dataset and tested it using the SOTON overhead dataset with more than 10,000 positive and negative images and achieved very impressive results. We then trained and tested SOTON data using positions detected by the SCOVIS model and got comparable results. We have demonstrated that our proposed algorithm not only locates the person in the given test image but also has the potential of automatic labelling of the centroid of the person. This suggests that the said generalised frame-work can be applicable to any overhead dataset without using any manual labelling of people in images.

1.5 Outline of Thesis

In this chapter we discussed introducing a new direction of person detection which is an overhead view. We compared how a person looks significantly different in a normal and an overhead view, respectively. We explained the limitation of normal view in an extremely cluttered environment with occluded people and the advantage of using an overhead view in such a situation. We also discussed the different effects of perspective change in the overhead view in the context of viewing people. The challenges associated with a person detection system have also been discussed using examples. The rest of this thesis is organised as follow:

Chapter 2 highlights some of the most prominent techniques used in person detection. We categorise people detection methods into two broad categories i.e classical or normal view based methods and overhead view based methods. We briefly discuss

different normal view based techniques used for person detection. There has been very little work done using an overhead view with a limited pose of a person just underneath the camera.

Chapter 3 describes detection of people from an overhead view. Most of the recent person detection algorithms using a normal view are based on the Histogram of Oriented Gradient. We therefore use this algorithm as a baseline and as a feature descriptor with our overhead view based images to detect people. We explain the implementation of the HOG algorithm step by step using our dataset. We also discuss in detail our overhead dataset. We optimize different parameters of HOG algorithm and discuss associated result of each parameter. The explanation about person classification results and spurious detections per image results are also discussed in this chapter.

Chapter 4 presents the limitations of the HOG algorithm using the overhead dataset. We demonstrate that the orientation of person and size varies with respect to the centre of the image when viewed from overhead. Therefore we propose a novel algorithm which uses variable size bounding boxes with different orientations based on the position of the person in the scene. The implementation of our proposed algorithm is discussed step by step. We compare average person images using overhead and normal view datasets, respectively. The criteria of using a variable size bounding box is also discussed. We also compare the results of our proposed algorithm with the standard HOG algorithm. The attempts made to reduce computational complexity and processing time per detection window are also discussed in this chapter.

Chapter 5 describes detecting people in images using our proposed algorithm and standard HOG algorithm. We first generally discuss the image-scanning process of both algorithms and then discuss overall results. At each stage, from image scanning to final detection results, we compare our proposed algorithm with the standard HOG algorithm. We also describe how additional background training boosts the performance of the detector. To fuse multiple classification scores per image we introduce our simple but effective clustering algorithm. The parameters associated with the clustering algorithm and the selection of the optimum parameters are also discussed in this chapter. We also explain the criteria of deciding the validity of a detected position. The chapter concludes with the discussion of the results and execution time per image.

Chapter 6 studies the application of person detection using our proposed simple tracking by detection algorithm. We explain how a simple blob-based motion detection framework integrate with our tracking algorithm speeds up the process. To evaluate the overall detection performance we compare our algorithm with four standard built-in algorithms of OpenCV. The experimental results using our five

test sequences are discussed. The limitation of Opencv algorithms and advantage of our tracking algorithm are also discussed.

Chapter 7 presents a generalised person detection framework. We recorded and labelled another new SOTON overhead dataset. The criteria of deciding bounding box size is then illustrated. We demonstrate how good the classification and detection results are using SOTON training and test data. We also show experimentally how a previously trained SCOVIS model produce excellent results while testing completely different images from the SOTON dataset. The last experimentation part is about the use of automatic labelling of people in images. The chapter concludes with a discussion about advantages of generalisation of person detection.

Chapter 8 summarises our main contributions and key results. This chapter also provides a discussion about the short-comings of our research work. At the end we suggest some possible future dimensions related to existing work.

Chapter 2

Person Detection

In this chapter we discuss some of the popular approaches used in person classification or person detection in images. These are grouped into two different categories i.e, Normal or Horizontal view and Overhead or Birds eye view based approaches. In a normal view based person detection system it is a common assumption that the person in the image is visible and has an upright or vertical pose. This is not true in a birds eye view. In the following two sections we describe the research work associated with these two categories. The justification for choosing the Histogram of Oriented Gradient algorithm as the basis for my analysis of overhead view based images is discussed in the last Section.

2.1 Normal View based Person Detection

While the Histogram of Oriented Gradient (HOG) algorithm has proved to be the best at detecting people in the images, there have been many other techniques. In the literature numerous methods and techniques have been used to detect people [38, 61, 79, 118]. In [47] different issues related to pedestrian protection systems have been discussed. [55] is a survey of pedestrian detection for an advanced driver assistance system where different learning algorithms and approaches used for human detection have been discussed. The authors also proposed an architecture for an on-board pedestrian detection system. We refer readers to [34] for a recent survey and a performance evaluation of most promising descriptors on different bench mark datasets. Hussain et al.[60] undertakes a discussion about different techniques used for reducing dimensionality of feature vector.

These approaches are either based on a sparse-representation or a dense-representation of an object. In sparse-representation, an object can be represented based on a key-point or image patch, image template, or part detector. In [74, 100], an explanation of different key-point based methods is given. Pinto et al.[91] discussed different state-of-the-art key-point detectors and evaluated their performance. The main factor in sparse-representation is that not every pixel of the image contributes to the representation of

an object; however, the opposite is the case in dense-representation, when every pixel contributes to the representation of an object.

In [96] different window based and part based models have been discussed. In window based methods a feature description is used for the complete person while in the part based methods individual feature descriptor is used for each body part of the person. In the computer vision and pattern recognition field a variety of part-based models [44, 42] have been used for object detection in general and person detection [93, 120, 41, 62] in particular. These are usually based on two fundamental steps. In the first step low level features or classifiers are used to form individual components or parts of the body of a person, and the second step is to integrate these parts/components to build a final descriptor.

2.1.1 Before Histogram of Oriented Gradient

Gavrila et al.[52] in 1999 proposed a person detection method based on template matching using the chamfer distance [14] and described its real time implementation within on-board vehicles. Different templates were used in off-line training. Figure 2.1 shows an example of the hierarchy of templates.

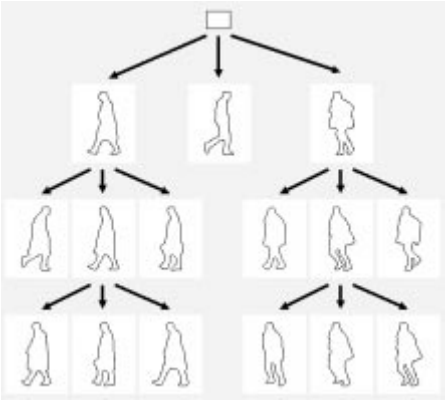
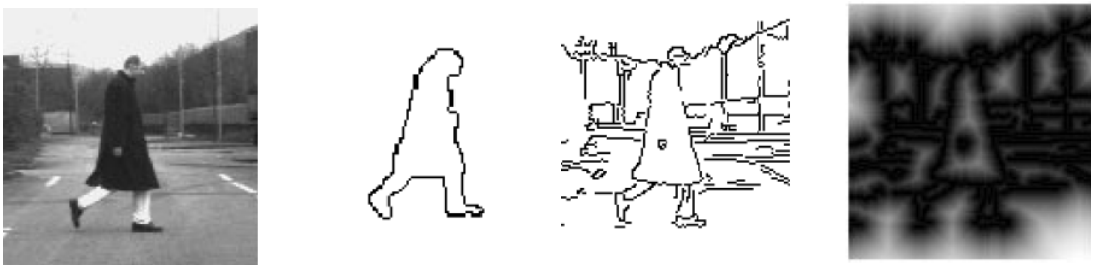


FIGURE 2.1: Hierarchy of templates used in [52].



(a) Original Image (b) Template (c) Edge Image (d) Distance Transform Image

FIGURE 2.2: This figure illustrates how to use template matching (from [52]).

The input test image as shown in Figure 2.2(a) is first converted into binary edge image Figure 2.2(c) and then into a gray level image (Figure 2.2(d)) after performing distance transform. A distance transform image is produced by computing distance of every pixel to the nearest edge pixel. Their hierarchical template-based classifier system matches the Distance Transform image as shown in Figure 2.2 with template shapes (Figure 2.2(b)). The matching of candidate template is done by traversing the tree from root to leaf using the depth-first strategy. The candidate test template is then validated by comparing the similarity measurement with a threshold. They use their own recorded database for pedestrian containing 1000 pedestrian shapes and obtained 5500 templates using 5 different scales (in the range 70-102 pixels height). To evaluate the performance of their system 700 test images of pedestrian are used and resulted in a detection rate of about 75%-85% with two or less false detections per image.

Papageorgiou and Poggio [89] in 2000 introduced a feature descriptor based on a type of wavelets [19, 56, 97] called Haar Wavelets. Haar Wavelets compute the pixel differences of two or more rectangular areas. Different masks as shown in Figure 2.3 are used for different configurations. A polynomial Support Vector Machine is used to learn and classify feature vectors. They used 1,848 examples of people (924 frontal and rear people images and their mirror images) with 11,361 non-people examples in training. They tested with 123 images containing people. All of these images are normalized to 64×128 pixels. This dataset has been referred as MIT pedestrian database in the literature. They achieve a detection rate of 90% with 1 false positive out of 10,000 samples.

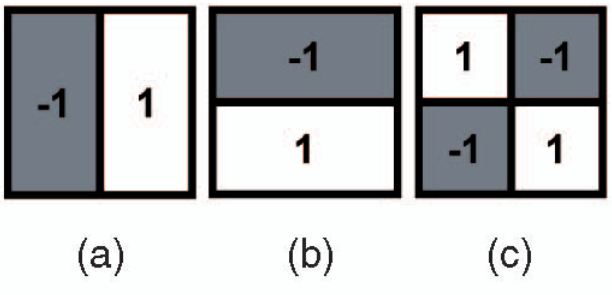


FIGURE 2.3: Different masks used in [55].

In 2001, Mohan et al.[76] presented example-based detectors by taking four components (the head, legs, left arm and right arm) of the human body using haar wavelet transform based features. Each of these components is trained and classified separately using an SVM and after applying geometric constraints their outputs are integrated into a final classifier. In their work they have mentioned that it is very important to place geometric constraints on the position and scale of body component directions s variations in these parameters might lead to false alarm and missed detections. The images were obtained from pictures of people taken in Boston, Cambridge and Massachusetts, with different cameras and under different lighting conditions. They used 889 positive examples and

3,106 negative examples for training the classifiers. The performance of the system is evaluated using 123 images [89] of people. They achieve a positive detection rate of about 90% at 10^{-4} false detection rate.

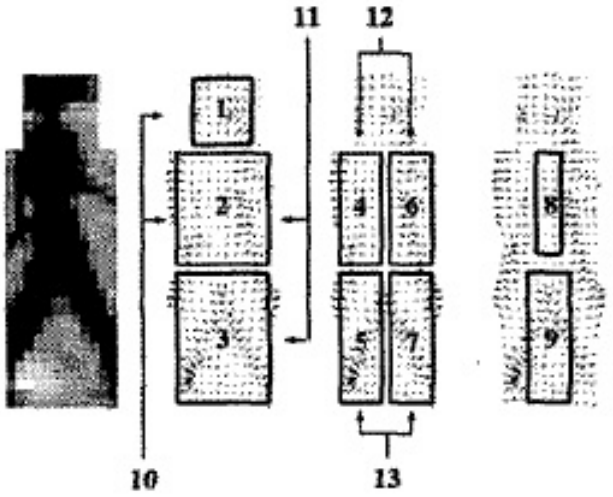


FIGURE 2.4: Part based classification using 13 overlapping regions (from [101]).

Shashua et al.[101] in 2004, use 13 manually labelled overlapping body parts as shown in Figure 2.4. The candidate region is selected from an image. This region of interest is then divided into a fixed configuration of nine overlapping sub-regions, and the histogram of gradients is computed for each sub-region. Each of these local vectors is divided further into 2×2 sub-regions with eight orientation bins weighted by gradient magnitudes. Thus, each local vector consists of 2×2×8 = 32 element local feature vectors (per sub-region), normalised to unit length (normalisation reduces the effects of illumination changes). The authors concatenated four additional local vectors (selected pairs of sub-regions) as shown in Figure 2.4 with these nine local feature vectors forming a total of 13 local sub-regions representing the candidate region. Each of these local feature vectors for sub-region is fed into discriminant function (Ridge Regression) and local discriminant results are integrated by Adaboost for second-stage classification. Their own recorded dataset consist of 25000 positive and same number of negative training samples while the test set consisted a total number of 15244 positive and negative samples. They achieved a detection rate of 90% at a false rate of 5.5%.

Mikolajczyk et al.[75] in 2004, present a human detection system based on the probabilistic assembly of local features of body parts according to geometric constraints. They use the decomposition of the body into seven parts (Legs, profile upper body, frontal upper body, profile head and face, frontal head and face, frontal face, profile head). They use joint like-hood model for assembling body parts using Laplacian based filters and gradient based features. To evaluate the detection performance, MIT pedestrian database [89] was used and achieved a detection rate of 87 % with a false positive rate of 1 in 100000 samples which corresponds to 1 false positive per 1.8 images.

In 2005, Wu et al.[119] introduce silhouette oriented features called edgelet features. These features are calculated at pixel level for different edges. The input image is converted into edge image using 3×3 Sobel filter. The orientation of each pixel in the range 0-180 degree is quantized into six bins which corresponds to integers in range 0-5. They use edgelet regions from 4 pixels to 12 pixels length. These edgelets include lines, curves and their symmetric pairs. For a given window size of 24×58 pixels (full body size of the person sample), there are overall 857,604 features. In Figure 2.5 the process of extracting edgelet features are shown. They use four body parts (legs, torso, head-shoulder and full body) to train AdaBoost. They use 1742 human samples, 925 of which are taken from MIT pedestrian dataset and rest of the samples are collected from the internet. Also there were 6000 negative samples taken from 7000 negative images were used in the training set. They use the CAVIAR[22] test set containing 54 images with 271 people for evaluation. Their combined part detector achieved a detection rate of 90 % with 20 false detections in these images.

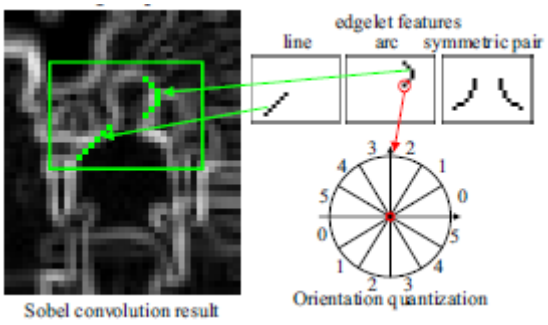


FIGURE 2.5: Edgelet features (from [119])

2.1.2 The Histogram of Oriented Gradient Algorithm and its derivatives

The Histogram of Oriented Gradient algorithm was proposed by Dalal and Triggs [29] in 2005 for detecting people in images and videos [30]. It has shown a very significant improvement in performance over previous work, achieving a TPR of almost 100% on the MIT database [89]. 90% [89] was the best result obtained previously.

The HOG algorithm (as shown in Figure 2.6) is briefly described below.

- Each input colour image is normalised for gamma and colour.
- The gradient of this resultant image is calculated by convolving a first order derivate mask in horizontal and vertical directions, respectively across the whole image.
- The next step is to divide each image into overlapping blocks and cells. Each detection window is divided into blocks. Each block is further divided into four

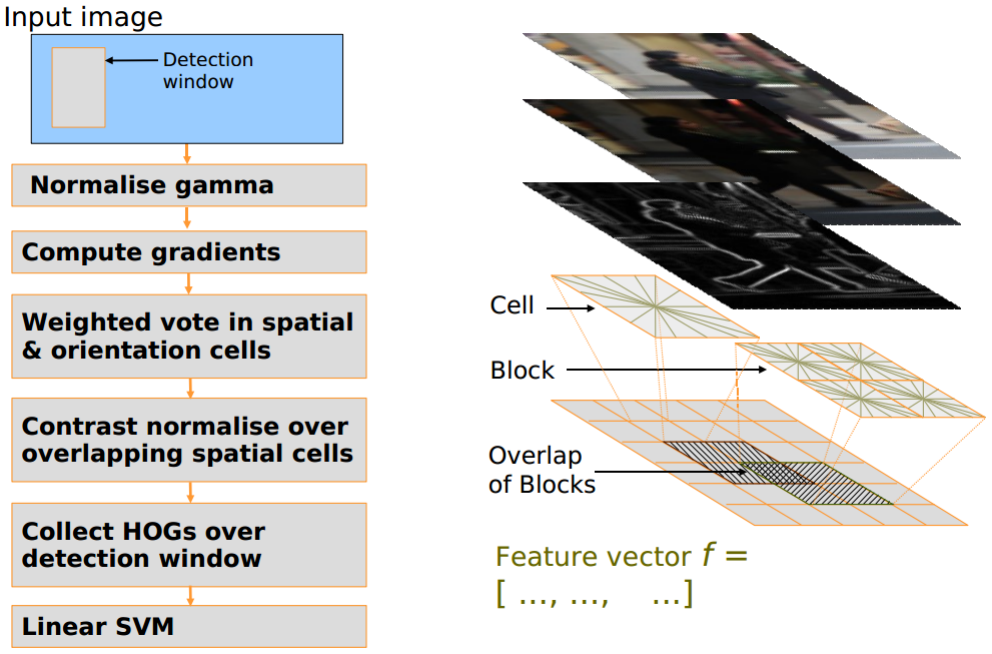


FIGURE 2.6: Steps for Histogram of Oriented Gradient Algorithm (from [29]).

regions called cells. The weighted votes (gradient orientation and magnitude) of all pixels within each cell are accumulated into equally spaced orientation bins. These n bins histograms from each cell are concatenated into a block.

- In the next step cells within each block are normalised using the $L2$ normalisation scheme to make these small cells invariant to illumination.
- Finally these normalised blocks are concatenated to form a feature vector.
- These features are then used to train and use with linear SVM for classification.

A detailed implementation of this algorithm is discussed in Chapter 3 using an overhead dataset of people. They initially test HOG descriptor trained with 509 training and 200 test images of pedestrian using MIT pedestrian database. These images contain only front or back views with a relatively limited range of poses of people. The people in these images are normalized to approximately same size 64×128 pixels in each image. MIT data set does not contain person-free images for background training. Due to these limitations in MIT pedestrian database, Dalal and Triggs produced a new and significantly more challenging INRIA dataset having 1208 pedestrian samples for training and 563 for testing. The negative training set contains 1218 background images and the negative test set consist of 453 person-free images. The overall results using the HOG algorithm as reported by the original authors are almost 100% true positive per window at 10^{-4} false positive per window using MIT database, while using the INRIA dataset they achieve a 90% true positive per window at 10^{-4} false positive per window. In their paper they have shown experimentally that HOG based features significantly

outperforms other features including PCA-SIFT, Haar wavelets and shape-context based features to detect people in images. Because of its excellent performance most of the techniques after HOG use variations of this algorithm.

Zhu et al.[125] (2006) improve the performance of HOG at the cost of large computational training time. They use variable size blocks (with sizes between 12×12 pixel to 64×128 pixel and aspect ratio ((1:1), (1:2), (2:1)) , integral image representation [27, 32] and cascade of rejectors using Adaboost. Integral image calculates the sum of pixels of a region or image in just four memory access operations. Adaboost (adaptive boosting) [45] is a learning algorithm used to construct a strong classifier by iteratively adding weak learners or classifiers. Zhu et al. use nine integral images corresponding to nine bins. They reported that 12×12 pixel blocks as used by the original authors[29] do not carry much local information about the object as compared to variable size blocks with different locations and aspect ratios. They increase blocks per detection window from 105 blocks (used by Dalal and Triggs) to 5031. As per reported by authors, their method is up to 70x speedy than [29]. Wojek et al.[117](2008) speed up processing using the Graphical Processing Unit (GPU) for HOG features computation. The process of scanning all detection windows across the whole image is very exhaustive and time consuming. [104, 12] use the image modality (using infrared images) to increase the performance by selecting the specific regions to search for a person. As reported by the authors warm objects (a person in this case) appear lighter than cold objects (the background environment) which are dark. This potential area in the image is only then searched for the presence of a person. Zhang et al.[124] (2007) proposed a multi-resolution framework to speed up the performance of person detection using HOG features. They use HOG features with different parameters at four different low-to-high resolution sequences with different sizes of blocks as shown in Figure 2.7. The image at lowest resolution is one-eights of the original size image.

Resolution	cell size	block size	detector size	#orientations	block stride	detector stride
1	(3, 3)	(6, 6)	(8, 16)	9	(2, 2)	(1, 1)
2	(4, 4)	(8, 8)	(16, 32)	9	(2, 4)	(2, 2)
3	(6, 6)	(12, 12)	(32, 64)	9	(4, 4)	(4, 4)
4	(8, 8)	(16, 16)	(64, 128)	18	(4, 8)	(8, 8)

FIGURE 2.7: HOG Parameters at different resolution (from [124]).

Pedersoli et al.[90] (2007) speed up the performance of HOG based features calculation by introducing square blocks using MIT pedestrian database [89] and used the block sizes varying from 32×32 pixels to a lower resolution 2×2 pixels. These different resolution blocks are shown in Figure 2.8. They have quoted that the method is four time faster than [125]. Geronimo et al.[54](2007) use both HOG and Haar Wavelets based features for person classification using real Adaboost [95] to speed up the performance with same detection rate as quoted by Dalal and Triggs.

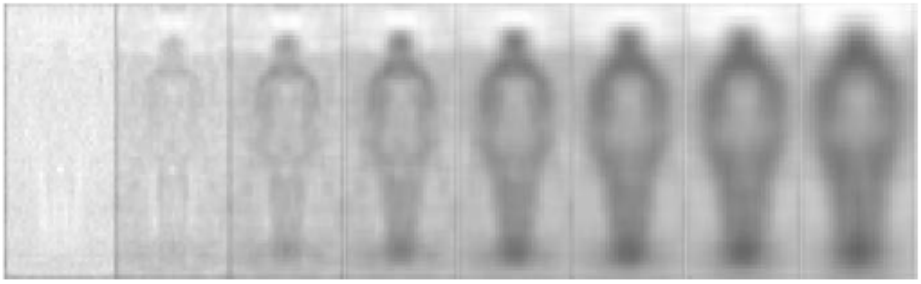


FIGURE 2.8: Square sized blocks at different resolutions. Left most 2×2 pixels and right most at a resolution of 32×32 pixels (from [90]).

Jia and Zhang [63](2007) present a real-time person detection system based on Viola's object detection framework[111] and HOG-based features. According to the authors, their proposed system has the discriminative power of [29] and real time properties of [111]. He et al.[59](2008) use HOG features at different scales [72] to form the Scale Space HOG. Scale Space is a special type of multi-scale representation that contains a continuous scale parameter and it preserves the same spatial sampling at all scales. The scale space representation of image I can be defined as in Equation2.1.

$$L(\sigma, I) = I * G(\sigma_i), \qquad i = 0, 1, 2, 3 \tag{2.1}$$

where $G(\sigma)$ is Gaussian kernel with variance σ and $*$ is convolution operator. For each sample image four images are produced at different scales ranging from 0 to 3 as shown in Figure 2.9. HOG features at each scale are calculated and then concatenated to form a feature vector. The results were similar to that of original HOG algorithm.

FIGURE 2.9: Scale Space representation: Images with four scales from 1 to 3 (from [59]).

Chuang et al.[18](2008) use HOG features by adding human shape properties (contour distances, symmetry, gradient density). To evaluate the performance they used their own recorded video sequences and a total number of 2390 positive and 14496 negative samples used in the training. The same number of samples used in the test set. A cascaded Adaboost classifier having 500 weak classifiers is then constructed. They compare their method with original HOG and achieved a gain of 5 percentage points in detection performance.

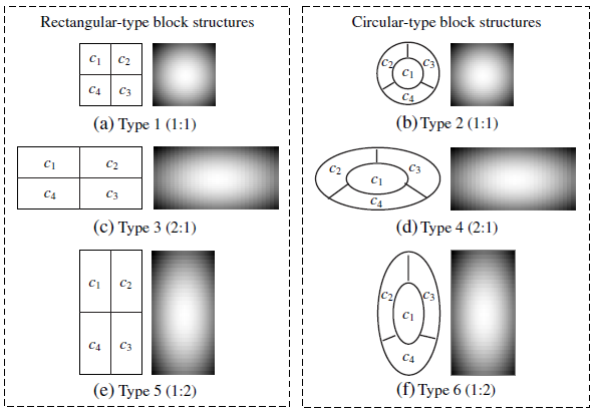


FIGURE 2.10: Six different blocks and their corresponding multivariate Gaussian-weighted windows (from [112]).

Inspired by Zhu et al.[125], Wang et al.[112] during 2009, enhanced the HOG method by introducing three variable size circular blocks, thus defining a total of 10062 blocks (i.e 5031 rectangular blocks and 5031 circular blocks) per 64×128 pixels bounding box. These six different types of blocks are shown in Figure 2.10. These blocks are made rotation-invariant. The dominant orientation(s) of each block is determined by voting its pixels into a 36-bin orientation histogram. Figure 2.11 illustrates how a block is chosen on the basis of their dominant orientation. The Adaboost algorithm is used to select most discriminative blocks. Each 320×240 pixels image was down-sampled by a factor of 8/9. Thus five different resolutions are formed. Figure 2.12 demonstrates the process of down-sampling an image and use of cascade Adaboost. They improve the detection performance from 90% to 94%.

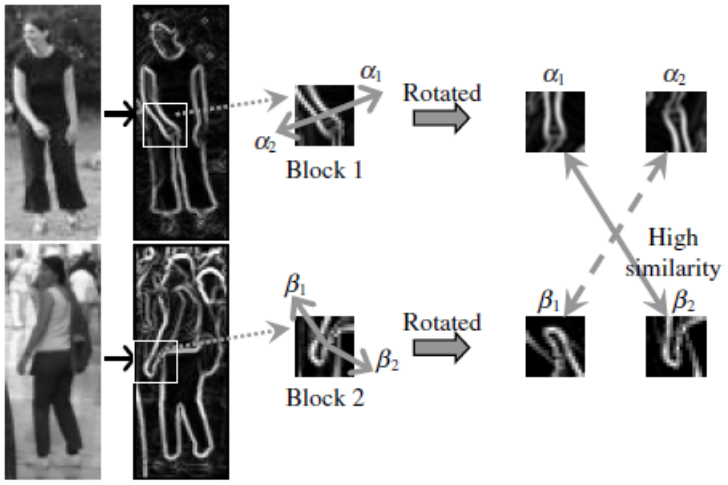


FIGURE 2.11: Rotation invariant block. Two different blocks in two different images looks similar to each other after being rotated on their dominant orientations to achieve the rotational invariant for each block (from [112])

Schwartz et al.[98](2009) combine texture and colour information with HOG features. These features are generated at very high dimensional space (more than 170,000 dimensions as compared to 3780 dimensions used in original HOG). They use Partial Least

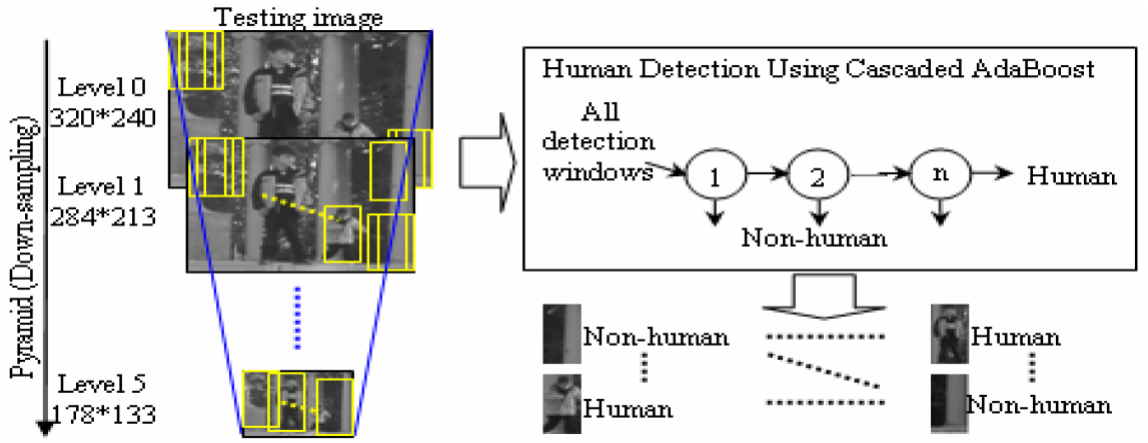


FIGURE 2.12: Work flow of image down sampling and use of cascaded Adaboost (from [112])

Squares based features selection method to reduce the dimensionality of features space. They achieved 94% detection rate at 10^{-5} false positive rate using INRIA pedestrian data set. Watanabe et al.[114] propose Co-occurrence Histograms of Oriented Gradients. The basic idea of Co-occurrence HOG is to handle gradient orientations in pairs instead of individually because as reported by authors and shown in Figure 2.13 these pairs have more vocabulary than using a single gradient orientation. Different from HOG, all orientations of the gradient image in a range $0^0 - 360^0$ were divided into eight orientations per 45^0. In this way the orientation of each pixel is represented by eight labels. Therefore the Co-occurrence HOG matrix becomes 8×8=64 elements. These features are used for classification using linear SVM. The results achieved were a detection rate of 88% at 10^{-6} false positive rate using INRIA pedestrian data set.

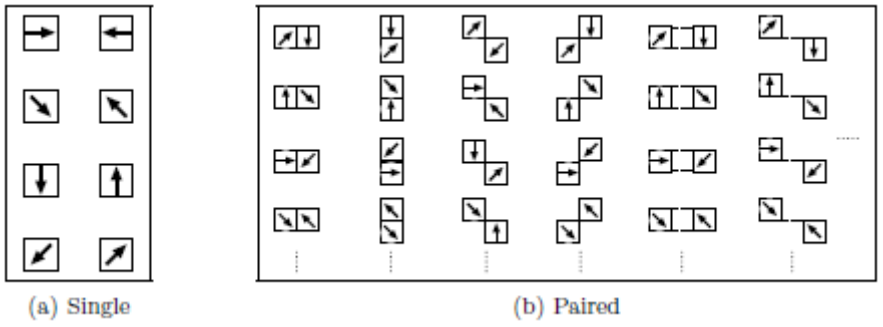


FIGURE 2.13: Vocabulary of gradient orientations. Though (a) a single gradient orientation has only eight varieties and through (b) a pair of them has many more varieties than the single one (from [114]).

Wang et al.[113](2009) combine HOG with Local Binary Pattern (LBP) for a person detection system and achieved a detection rate of 94.7% at 10^{-5} false positive rate using INRIA database. The Local Binary Pattern was originally proposed by Ojala et al.[84] for texture classification and later on was used in various pattern recognition fields

including face detection [58], face recognition [6] and more recently, for person detection [78, 82, 106]. Figure 2.14 shows an example use of LBP features. In this figure the red box in the first image is the region used for extracting LBP. The central pixel values is 20 and its nearest 8 pixels values are also shown in the second image. The value above the central pixel value is considered as 1, and the value below is considered as 0. Applying this process decomposes this region into a binary image as shown in the last image in Figure 2.14.

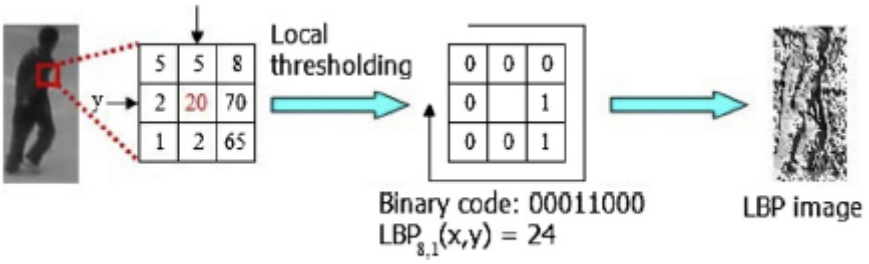


FIGURE 2.14: An example of Local Binary Pattern (from [106])

However, there are still some significant algorithms which are not based on histogram of oriented gradient direction.

In 2007, Gavrila et al. [52, 49, 53] has extended their work using the probabilistic approach [50] to estimate the posterior probability of the object class with Bayesian model and used for pedestrian tracking from a moving vehicle [51]. The results as quoted by the author in [50] were a detection rate of 80% with on average more than 3 false positives per image when tested on their own recorded dataset containing 2254 pedestrian in 1306 images. Lin et al. [71](2007) combined global template approach and local part based template in human detection. Figure 2.15 shows an example of the hierarchy of different part based templates used in this approach. They achieve a detection result of 90% with 10 false detections when tested on USC pedestrian dataset-B [119] contains 54 grayscale images with 271 humans.

During 2008, Leibe et al. [68] use Implicit Shape Model (ISM) [67, 69] framework for person detection and is shown in Figure 2.16. Instead of using shape templates or low level features, this category uses the key-point based features where a codebook for human detection is used. A code book is actually a combination of different local appearance patches or sub-regions of an image. These patches are selected automatically using different corner detector algorithms like the Harris Corner Detector [31]. They use 206 test images containing 595 annotated pedestrians from TUD pedestrians dataset [5] and achieve 80% detection rate with 119 false positives.

Inspired by the work of Gavrila [50], Nguyen et al. [37] in 2009 proposed a weighted template matching method by weighting the strong edge points during distance trans-

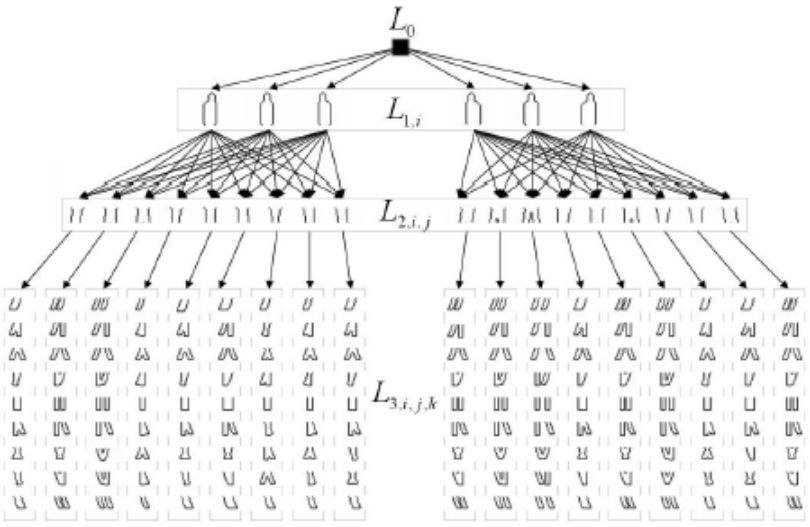


FIGURE 2.15: Hierarchy of part template and their association (from [71]).

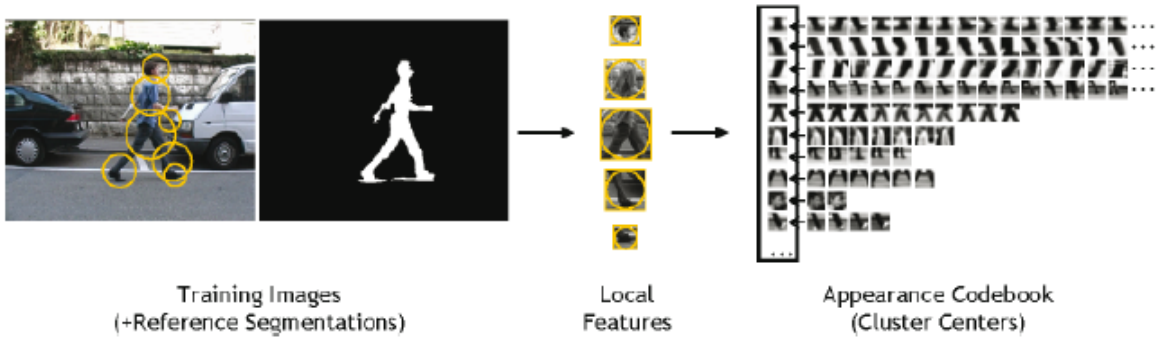


FIGURE 2.16: The training process of person detection system using the Implicit Shape Model (from [68]).

form. The results of template matching are then verified by Bayesian framework. The performance of their system at MIT database was a true positive rate of 90% with a 30% false positive rate. They compare the performance of Gavrila's method [50] on the same database and achieve a marginal improvement. In [80] they decompose the full body template into sub parts and then use these part templates with a Bayesian-based model for human detection. This is a two-phase approach; in the first phase, body parts are determined using template matching and validation of detected parts is performed in the second phase. Figure 2.17 shows an example of decomposing full body template into part templates. In Figure 2.17 four types of part templates are shown. These body parts are 1) top body part which covers head shoulder and torso, 2) lower parts includes both legs, 3) left body templates and 4) right body templates. The method is evaluated on USC-C dataset [119] and achieve a true detection rate of 80% with a 20% false positive rate. In [105](2009) the same authors use template matching technique to test with images of INRIA dataset and achieve a true positive rate of 80% with 10% false positive rate while at the same true positive rate result was 50% false positive rate using the method proposed by Gavrila [50].

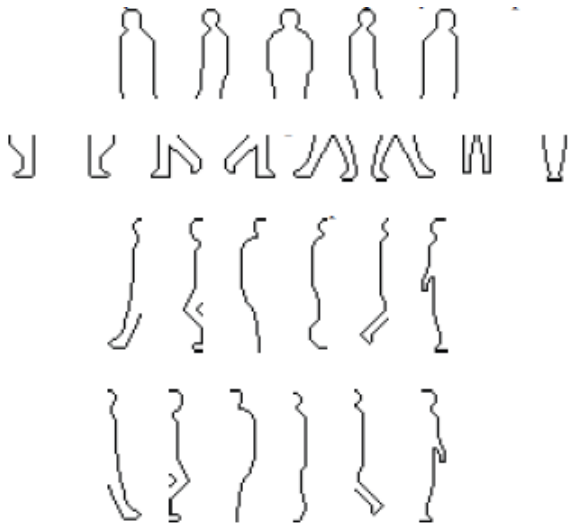


FIGURE 2.17: Four different part templates (from [80]).

2.1.3 Recent work in person detection

In 2010, Lee et al.[66] use body ratio estimation and HOG features for classifying human/non human. DaimlerChrysler pedestrian data set [21] is used to train a descriptor using original HOG algorithm. The evaluation is done using their own recorded video test sequences. They first extracts moving blobs through background subtraction and then applies human body ratio (width:height) estimation technique to determine the possible presence of a human candidate. These segmented regions are then tested for a person/no person classification. In total four test sequences with 1575 sample images are used and achieved on average a detection rate of 80%.

During 2011, Beiping and Wen[10] present a person detection method based on HOG features. They train HOG descriptor using INRIA data set and tested using three different recorded video sequences. From test sequences they use consecutive frame differencing to extract motion regions. These motion regions are searched for person detection. The results they achieved were a detection rate of 90% with a total number of 116 false positives in 1270 images. Xin [121](2011) use combined method of Haar and HOG based features extraction to speed up the process upto 10 frames per second using images with resolution 320×240.

The work of Nguyen et al. [37] is further extended [81] in 2011 using variational mean field approximation to handle occlusion. They use template matching to detect individual body parts of human. The hypothesis for these initial detections along with spatial layout information is modelled by graphical representation as shown in Figure 2.18. The initial detections are refined by Bayesian estimation, and to approximate such estimation a mean field method is used. This method was evaluated on the OneStop-MoveEnter1cor sequence of the CAVIAR-INRIA dataset [22]. In total 200 images containing 1614 an-

notated people are taken as a test set. They achieved 90% precision at 20% recall. In their paper the authors have shown improvement in performance over method proposed by Lin et al. [71].

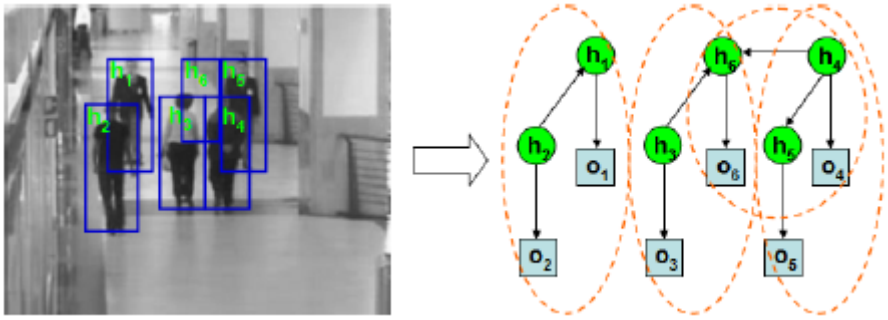


FIGURE 2.18: Left: Initial detection hypotheses, Right: the graphical model in which ellipses represent local groups (from [81]).

Morzinger et al.[77](2011) use multiple self-calibrated cameras for person detection. They used the cascade confidence filtering technique for automatic ground plane estimation and muti-camera correspondence estimation. They use images from (SCOVIS dataset) [1]. To evaluate performance 1000 frames (every 10th frame in a 40 seconds image sequence) are used where visible people are labelled manually. By using multi-view of two cameras they achieve precision of 80% with a recall of 50%. The advantage of the using mult- camera system is that it prevents the system from losing the target object, particularly in the case of occluded objects in the scene. The disadvantage of the multi-camera system is that it is costly to install multiple cameras. Furthermore camera calibration and cross-camera correspondence are difficult tasks.

During 2012, Nguyen [106] use LBP features combined with global features (template, keypoints at edges of templates) to detect people. These feature combinations are shown in Figure 2.19, where keypoint based features are extracted from different body parts' templates and combined with LBP to form the features vector. They compare the performance of their method with state-of-the-art using INRIA dataset. The person classification results were good but the results of detecting people from images were not better than the original HOG algorithm.

In 2012, Pang et al.[87] propose two fast ways of extracting HOG features, which results in an improvement of efficiency. One way is to reuse block-based features by concatenating these to construct HOG features for intersecting detection windows. The second way is to utilise sub-cell based interpolation. The basic idea in both methods is not to use redundant blocks or cells while constructing features for overlapping detection windows in image.

Wei et al. [115](2013) integrate Haar-like features, HOG based features, Adaboost algorithm, Support Vector machine and head-shoulder information in their proposed

[1] http://www.scovis.eu

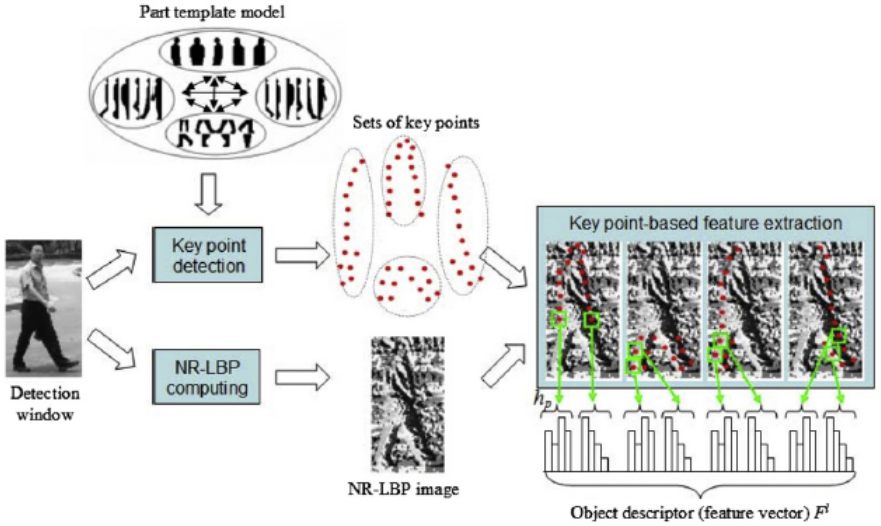


FIGURE 2.19: Combining keypoint based features with LBP to generate final feature vector (from [106]).

algorithm. They achieve a detection rate of 95% with 2.4% false positive rate using their own person dataset.

2.2 Person Detection from an Overhead View

There have been little work in detecting people from an overhead view. Most have been constrained to a narrow field of view when the person immediately below the camera.

In 2000, Cohen et al. [20] use a highly constrained environment with background subtraction method for head segmentation using a very simple and static background. n this constrained environment they impose a range of other, different restrictions; for example, one of the assumptions is that the appearance of the person should not be change significantly. They use people samples just below the camera for the purpose of using hair colour and texture-based features. Their main focus was to extract the segmented heads of people for onward processing. Figure 2.20 shows some sample images; it can be seen from this figure that a very limited pose (just below the camera) of the person has just been considered. Aradhye et al. [7] during 2005, use overhead images to present a person recognition system based on hair colour and texture based features. These features include hair line, segment, orientation, length, shape and colour.

During 2005, Snidaro et al. [102] proposed a people tracking system for ambient intelligence which is based on a network of smart sensors that cooperatively aim to understand the movements inside a monitored building. Background differencing is performed to extract moving regions or blobs. Different features are then extracted for each blob including area, density, centroid coordinates, mean color values and colour histograms. These features are used along with the mean shift algorithm [25] and a Kalman filter

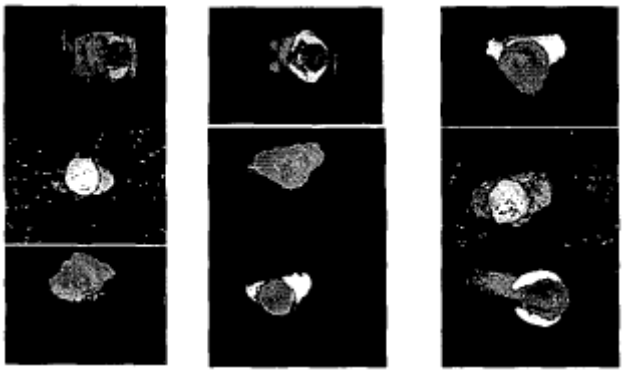


FIGURE 2.20: Head segmentation using simple background subtraction (from [20])

[116] to track the people. Fascioli [40] presents a vision based pedestrian crossing system for zebra crossing. A sample image used in their system is shown in Figure 2.21. They used stereo vision approach to extract difference image where the moving objects appears brighter than backgrounds. A search for these brighter regions are made and the adjacent overlapping brighter regions which likely to represent the same object are merged into a single blob. The stability of the detected blobs are maintained using history information from the previous frame. As shown in Figure 2.21 the direction of pedestrian and vehicles on the zebra crossing area is different, therefore the discrimination between a person and a vehicle is performed by considering the orientation of the movement of detected blob.

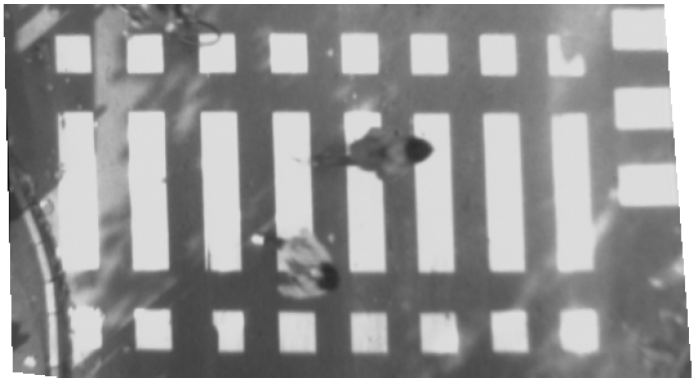


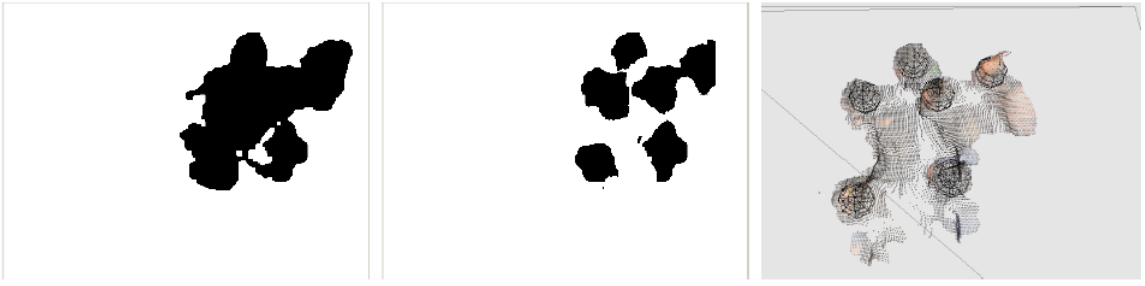
FIGURE 2.21: Image acquired by the system used in [40]

Velipasalar et al. [109] propose an automatic head counting system of people. They used a simple and constrained environment as shown in Figure 2.22 using the knowledge of entry and exit point within the monitoring region of the scene. A background subtraction process is performed to get motion blobs. Their system has two counters for entering and exiting people. The information about door location is used to determine the direction of the moving blob. When a blob crosses the entry/ exit region, its size is used to find the number of people forming that blob. Depending on the direction, the corresponding counter is then incremented by the number of people. Yahiaoui et al. [122] present a stereo vision based system for passengers counting in buses. Their main objective is to

FIGURE 2.22: A sample image for automatic head counting system used in [109].

provide a precise passengers counting system adapted to buses environment. Yu et al. [123] propose a foreground/background model for detecting and counting moving people in the video sequence. For the every frame in the video sequence the canny edge detector algorithm is applied using a threshold to get edge image. As quoted by the author most of the time the edges of the background images remains static compared to the edges of moving foreground object. Based on this idea they construct an edge model to extract foreground from the background. A region of person is then extracted from foreground edge object. The similarity of the size ratio of moving objects(people) is used between the consecutive frames to classify a person.

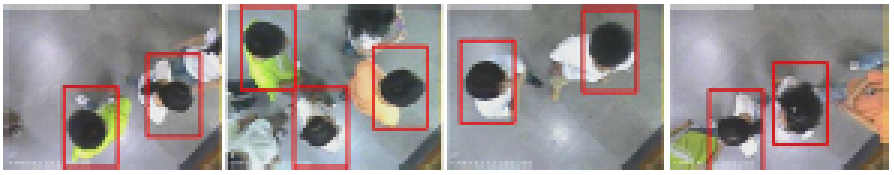
FIGURE 2.23: Original Image used in [107]



(a) Blob detection: 1 blob (b) Height thresholding: 5 blobs (c) Head detection: 6 heads

FIGURE 2.24: An example of detecting heads from the image. (from [107])

During 2011, Oosterhout et al. [107] propose a head detection system for people counting using stereo camera. The output from the stereo camera is the colour map and depth



(a) Sample images with people. Mostly only head can be seen with a very simple background.

(b) Sample background images.

FIGURE 2.25: The top view images with narrow lens camera. These images have simple background (from [87]).

map image. An adaptive background model is then used to determine the dynamic portion of both maps. They use an algorithm which matches a spherical crust head shaped template on the foreground regions of the depth map. Figure 6.3(a) shows the original image and the process of detecting heads from original image is shown in Figure 6.3(b).

Garca et al. [48] in 2013, present a people counter system based on head tracking. The gray scale image is converted into foreground edge image after applying background model and Canny edge detector. The authors then used different circular shapes masks to search for head detection. Rauter Michael [92] present a method of human detection in depth images using stereo camera. To localize the head region in the image a gradient climbing algorithm which is similar to Mean Shift algorithm is used. This results a list of head candidates.Instead of using any background/foreground subtraction they used depth feature descriptor.

Ozturk et al. [86] propose a method of human tracking and determining the body and head orientation of a person. In the first step motion regions or blobs in the scene are extracted by a background subtraction algorithm. In the next step color histogram and edge-orientation histogram models for each blob are generated to describe the appearance of the human. The orientation of the body is measured by matching the head to shoulder region of the body with predefined shape templates.

Recently Pang et al.[88] propose two fast methods of extracting HOG features. In their work, they record top view based images against a very simple background, as shown in Figure 2.25(b), with the sample of people just below the camera as shown in Figure 2.25(a). It can be seen that mostly the heads of the people are visible because in all of these images the people are in a narrow range of positions under the camera.

2.3 Why Histogram of Oriented Gradient ?

When I started my PhD in 2010, there were a number of different methods for detecting people in images. From these I chose to use the HOG features as my overhead view based algorithm to detect people for the following reasons:-

- HOG based algorithms have been used extensively and have proved to be superior as shown in two independent surveys. In the survey by Dollar et al. [33] in 2009, they evaluated the performance of seven promising pedestrian detectors using ten challenging pedestrian datasets and concluded that HOG algorithm remains competitive, especially when properly benchmarked (using per image metrics). In another survey, Enzweiler et al. [38] in December 2009, have shown that HOG-based features combined with a linear SVM significantly outperformed all other approaches.

- The advantage of shape template based matching methods is that, these techniques are computationally fast and could be used in real-time systems but the drawback of these type of methods are that they generate a high number of false positives. These methods detect only upright visible people in the images and the performance degrades as the human poses become more diverse in the images. The diversity of pose is much greater in an overhead view. Some examples are shown in Figure 2.26. Furthermore these methods do not work well in case of a cluttered background with occluded people in the images.

- The advantage of part-based approaches over window based approaches is that the former can handle partial occlusion better than a holistic detector. However, part-based models tend to require higher resolution images than most window based approaches [96]. Another problem with the part-based approach is that in images with a cluttered background or occluded people, it might generate too many detection hypotheses and a cascade of rejector mechanism is generally used in this case. To assemble each body part in order to make a final decision, a robust assembly method is also needed. These methods works on the basis of a very constrained geometry of the associated body parts of the person. These associations are very straight forward in case of people in normal view images but has become much complex in case of images used from overhead view as shown in Figure 2.26.

The power of the HOG algorithm is that object appearance and shape can be easily characterised by the distribution of intensity gradients or edge directions. HOG is not only a good person detector but has also been proved as a good object detector. In a survey in 2012, Dollar et al.[34] (2012) extended their previous work [33] and compared

(a) Just below the camera (b) Upward (c) Downward rear side (d) Downward front side

(e) Left side (f) Left diagonal (g) Right diagonal (h) Right side

FIGURE 2.26: Overhead view and perspective: Variation in size and orientation of people due to perspective change.

the performance of sixteen representative state-of-the-art person detectors on seven different publicaly available datasets. They use different variations in parameters, e.g. scale, positions and occlusion to evaluate the performance and resulted that HOG based hybrid features outperforms to all others. This vindicates the choice of the Histogram of Oriented Gradient algorithm as the start point for this work.

Chapter 3

Detecting people from an overhead view

The Histogram of Oriented Gradient (HOG) algorithm was proposed by Dalal and Triggs [29] to detect people in normal or conventional views with the assumption that people should be visible and upright in the images. This chapter mainly focuses on using the HOG algorithm with our dataset in which people are viewed from overhead using a wide angled lens camera. We are neither using any assumption about the visibility of person nor restricting the pose. The main conclusion of this chapter is that the HOG features gave encouraging results when used for person classification but a poor performance when scanning the whole image to detect people.

The detailed implementation of using the HOG algorithm is discussed in Section 3.1. The description about the dataset is discussed in Section 3.2, and the classification and learning algorithms are explained briefly with examples in Section 3.3. We illustrate different parameters of the HOG algorithm along with associated results in Section 3.4. The classification and detection results with tuned parameters are discussed in Sections 3.5 and 3.6 respectively.

3.1 Extracting Features using the HOG algorithm

In this section we use the standard HOG algorithm to extract features from images in our overhead view dataset. In the features extraction process we use the default parameters proposed by the original author. The complete algorithm is written using OpenCV [13, 65] to have full control over various parameters of this algorithm. We are not using any built-in implementation of this algorithm in OpenCV. Different phases of this algorithm are explained step by step, as follow:

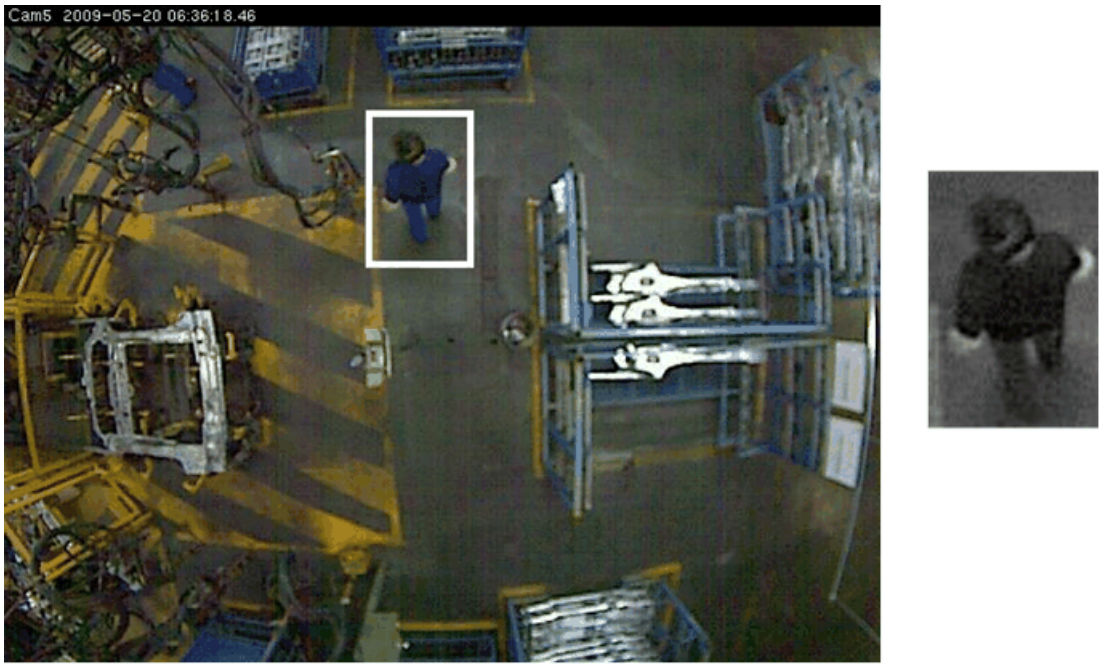


FIGURE 3.1: Cropping out of bounding box image and conversion into gray-scale image.

In the first step an appropriate sized bounding box image (the image region contains a person in it) is cropped and then converted into gray scale image I. This process is shown in the Figure 3.1.

The next step is to compute gradients in horizontal and vertical directions. Dalal and Triggs in [29] reported that using simple first-order derivative without smoothing gives the best performance than using other complex edge detectors like Sobel [83]. Equation 3.1 and 3.2 shows the horizontal and vertical gradient computations respectively of the gray scale image $I(x, y)$. The central differences in the x- and y- directions at pixel location(x,y) are represented by $d_x(x, y)$ and $d_y(x, y)$ respectively. The gray scale image, x magnitude image $|d_x|$ and y magnitude image $|d_y|$ are shown in Figure 3.2(a), Figure 3.2(b) and Figure 3.2(c) respectively.

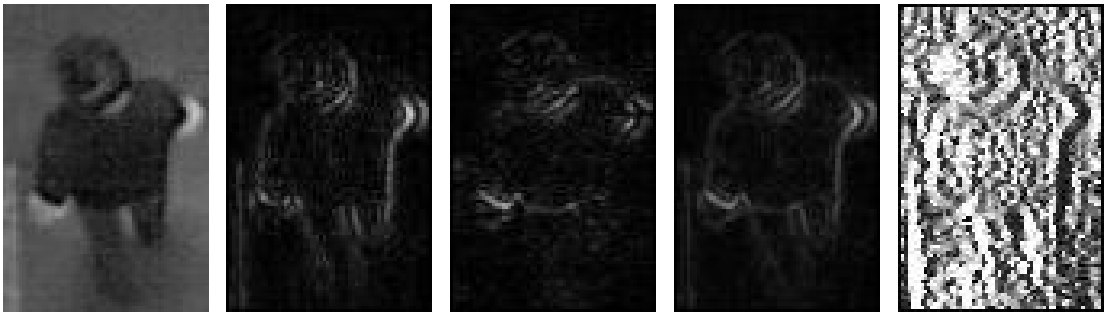
$$d_x(x, y) = I(x + 1, y) - I(x - 1, y). \tag{3.1}$$

$$d_y(x, y) = I(x, y + 1) - I(x, y - 1). \tag{3.2}$$

$$m(x, y) = \sqrt{d_x(x, y)^2 + d_y(x, y)^2}. \tag{3.3}$$

$$\theta(x,y) = \begin{cases} \arctan\ (\frac{d_y(x,y)}{d_x(x,y)}) - \pi & \text{if } d_x(x,y) < 0 \text{ and } d_y(x,y) < 0 \\ \arctan\ (\frac{d_y(x,y)}{d_x(x,y)}) + \pi & \text{if } d_x(x,y) < 0 \text{ and } d_y(x,y) > 0 \\ \arctan\ (\frac{d_y(x,y)}{d_x(x,y)}) & \text{otherwise} \end{cases} \tag{3.4}$$

The gradient magnitude of each pixel is computed using Equation 3.3 and is shown in Figure 3.2(d). The angle or orientation in the range (0-180) degree for each pixel is computed using Equation 3.4. The resultant image is the gradient image, and is shown in Figure 3.2(e).



(a) Gray-scale (b) X Magnitude (c) Y Magnitude (d) Magnitude (e) Gradient

FIGURE 3.2: This figure shows the gray scale image (a), x magnitude image (b) y magnitude image (c) Magnitude image (d) and gradient image (e).

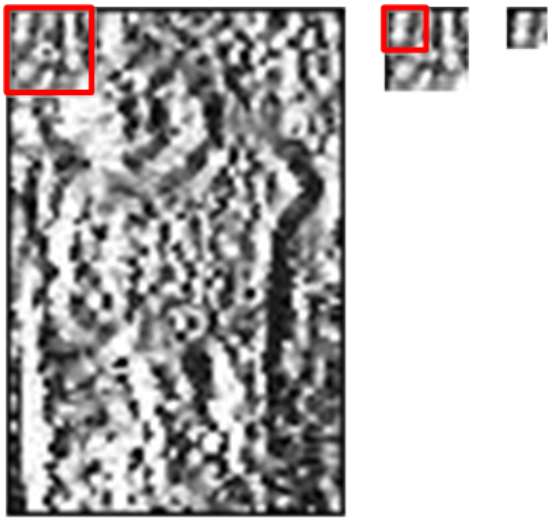


FIGURE 3.3: Extracting block and cell from gradient image : This figure shows how a block (16×16 pixel sub region of bounding box) is extracted and a cell (8×8 pixel sub region of block) is extracted from a block.

This bounding box or detection window is then divided into 16×16 pixel overlapping blocks with four pixels steps in both direction. Each block is further divided into four 8×8 pixel regions called cells. Figure 3.3 shows gradient image, extracting 16×16 pixel block and 8×8 pixel cell. An example of a 8×8 pixel cell with the orientation of each pixel in shown in Figure 3.4.

Dalal and Triggs [28] used two kinds of block geometry; a reactangular block and a circular block. In circular block, the cells are defined into grids of log-polar shape. We use the dafault rectangular block of HOG algorithm, because as reported by the original authors, the performances of using both type of blocks are very similar, but circular block is slower to compute owing to its log-polar grid.

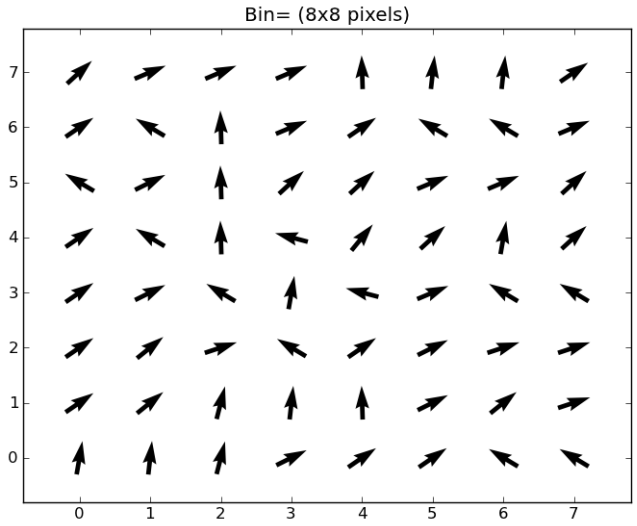


FIGURE 3.4: Orientation of pixels in a single cell (8x8 pixels).

The next step is to calculate local weighted histogram of oriented gradient by accumulating votes of each pixel of cell into nine bins. These votes are weighted by gradient magnitude. This process is represented mathematically in Equation 3.5, where gradients and magnitude of a single cell is represented by $h(\phi_p)$ and m_p respectively.

$$h(\phi_p) = h(\phi_p) + m_p \tag{3.5}$$

The original author used un-signed gradients within a range $(0 - 180^0)$. The size of the bin is calculated by dividing total angle range (e.g, 180 here) to total number of bins (e.g, nine here). Thus 180/9 gives a bin size of 20. The angle of each pixels is then associated with a particular bin. For example bin 1 has an angle range of 0-20 degrees, and bin 9 has an angle range of 160-180 degrees. This process is shown in Figure 3.5.

The next step is to concatenate all these weighted local histograms of oriented gradients into blocks. We know that in the real environment the illumination is not constant and it varies because of different factors like increase or decrease of lighting, shadowing of other objects, sun lighting and so on. Each $8\times$ 8 pixels cell gradient histogram is replicated four times per block and we know that the gradient is affected by the change in illumination; therefore normalisation is needed. Normalisation actually takes the maximum range of

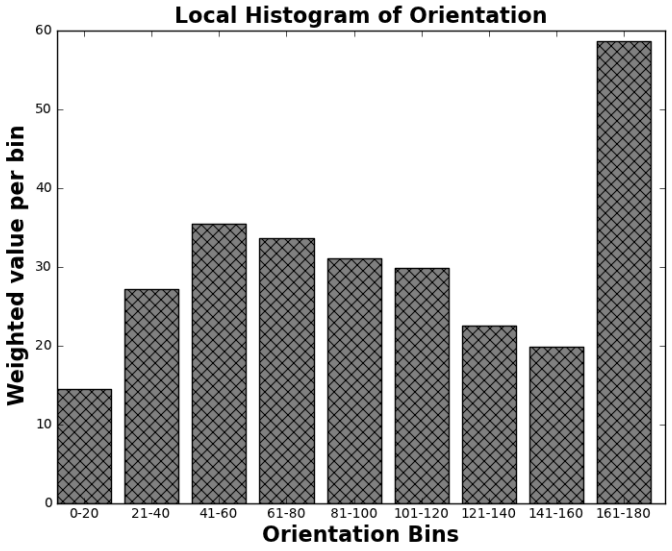


FIGURE 3.5: Histogram of local bins: This figure shows assignment of angles weighted by magnitude into nine orientation bins. Each bin is 20 degrees wide.

a signal and stretches it up to the maximum possible range. Therefore the cells within each block are normalised to make them invariant to illumination. Block normalisation is performed using Equation 3.6 with $L2$ normalization:

$$V = V_2 / \left(\sqrt{||V_2||^2 + \epsilon} \right) \tag{3.6}$$

where V is normalised and V_2 is unnormalised descriptor vectors respectively. Here ϵ is a small constant to avoid division by zero.

Due to block overlapping, each cell replicates four times in different blocks. In such situation, the final feature vector contains some redundant information about normalized cells. Overlapping ensures consistency across the image without the loss of local variations. The block overlapping effect has been demonstrated in Figure 3.6. In this example the 16×16 pixel blocks are shown in green square. Each block has two horizontal and two vertical cells of 8×8 pixel. Here, with eight pixels stride or 50% block overlapping, each cell has been used four times throughout the feature extraction process. This means that each block uses 50% redundant information which makes features vector much more robust against the illumination effect.

Therefore, for a detection window of size say, 64×96 pixel with 16×16 pixel block and 8× 8 pixel cell, the total number of features is calculated as:

Detection window = (7×11 blocks)×(2×2 cells)× (9 bins) = 2772 features.

These features are used for training and testing using a Support Vector Machine (SVM).

In this section we have discussed briefly about the HOG algorithm. We use this algorithm as a baseline of our experiments using the overhead dataset explained in the next section.

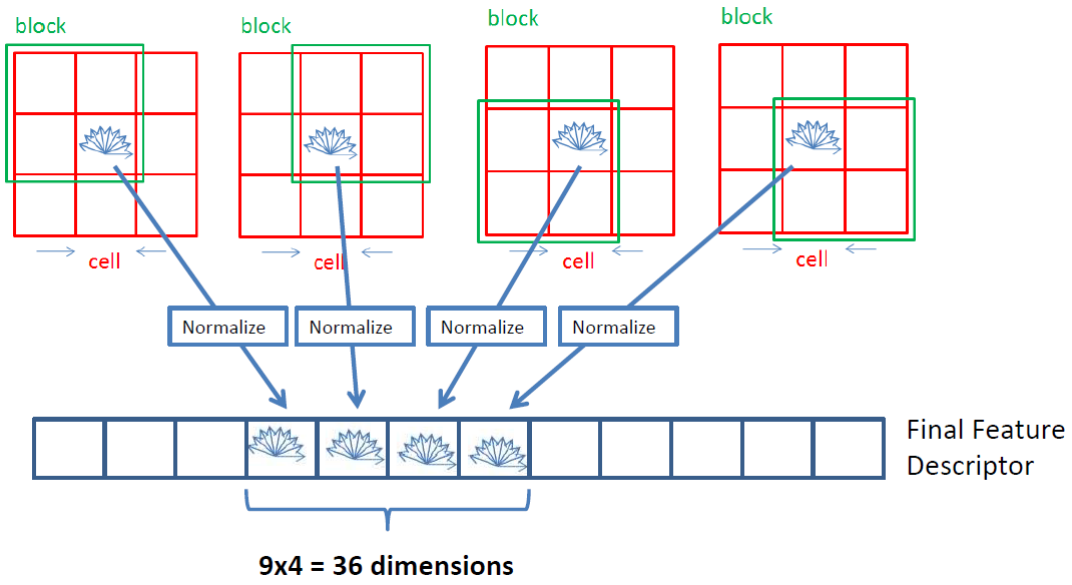


FIGURE 3.6: Overlapping Blocks: Each cell has been visited four times during features calculation process. (from [43])

3.2 Dataset

The images used in this work are taken from the Self-Configurable Cognitive Video Supervision (SCOVIS) [1] project [36], where a real-world industrial dataset was recorded. It is about an automobile construction environment of one of the NISSAN Motor Iberica SA (NMISA) sites, in Barcelona, Spain. This is a totally uncontrolled environment. Therefore there are variations in lighting and oscillation of camera due to the motion of the construction robots. In this environment there are some special traits on the floor which limits the working area, but for the motion of humans there is no restriction. The clothing colours are not restricted and there are many cases where the moving objects including humans have the same colour as the background. Images of size 640×480 pixels were captured at 20 frames per second from an AXIS 212 PTZ camera. This was mounted about five meters above the working area. In particular, the specification of the PTZ 212 camera are, Fujinon lens, F1.8, fixed iris, vertical viewing angle: 35^o-105^o and horizontal viewing angle: 44^o-140^o. The images were recorded at a low light level with high gain and JPEG compression. The average compression factor is 20. The size of each image is about 50 KB.

Figure 3.7 shows the description of the environment. This is an industrial environment of car assembling where people perform different activities like carrying, handling, welding, walking and standing. The main parts of the scene are numbered within white circles. These are Working framework, Welding plant, Racks containing parts of the car, and Folklift truck respectively. For most of the time people in the scene carrying different parts of the cars to/from racks shown as labelled region 4 and 5 in Figure 3.7. These

[1] http://www.scovis.eu

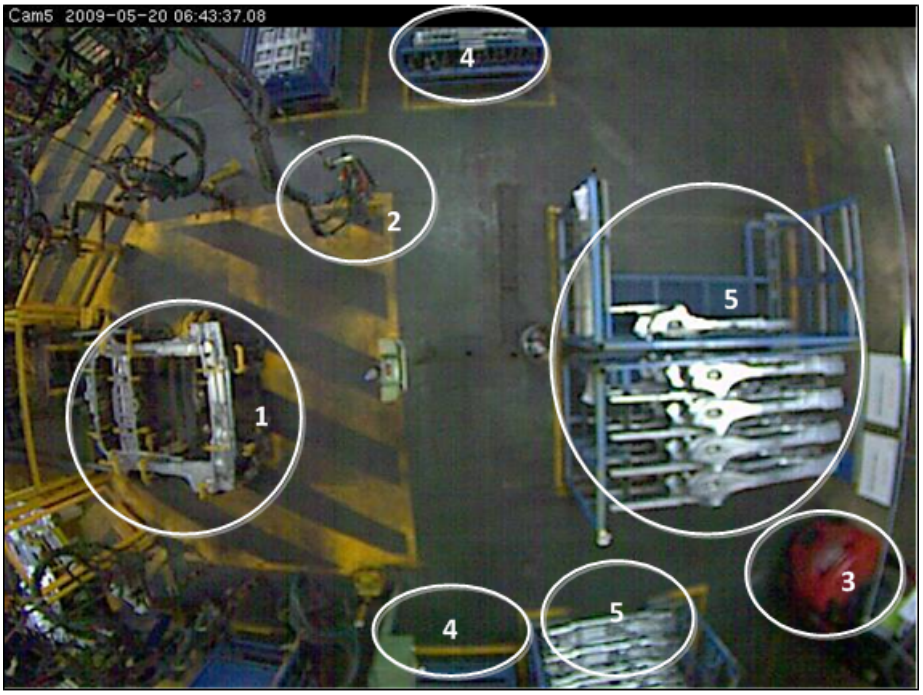


FIGURE 3.7: Description of the scene: Main regions of the scene are numbered within white circle 1. Working Framework, 2. Welding Plant, 3. Folklift truck 4-5. Racks.

racks are carried and removed by a folklift truck labelled as 3. The car assembling parts from these racks are brought to the working frame work (labeled as 1) by the workers. The size of this assembly framework is about 4x4 metres. Sometimes different parts are jointed together using a welding machine shown labelled as 2. These jointed parts are then picked up by a Robotic machine mounted on the top of assembly framework.

In these data, more than 50% of the images of people are occluded in some manner. Some sample images are shown in Figure 3.8. The first pair of images as shown in Figure 3.8(a) are labelled as un-occluded. In this pair the person is almost directly under the camera in the first image and away from the centre of the camera in second. In the next pair, as shown in Figure 3.8(b), the lower bodies of the people are occluded in more than 50% of cases with rods and racks. Similarly, in Figure 3.8(c), occluded people are shown while bending and handling over the racks. In the last pair shown in Figure 3.8(d), two samples are partially viewed persons whose heads are clearly not visible. Thus using the overhead view with these type of extremely occluded and challenging images makes the task of detecting people much more difficult.

In total, 6327 images with people and 6000 images without people were used. In the former, there are a total of 13016 people with upto three visible in each image. The positive instances were selected and labelled manually while the negative examples were chosen randomly. While labelling the data we also recorded subjective information about each person's visibility or occlusion.

(a) Normal People (b) 50% occluded people (c) 25% occluded people (d) 25% invisible people

FIGURE 3.8: Some sample images of people from overhead data set.

3.2.1 Image annotation

For labelling positive images, we developed a simple program written in C++ which reads each image name from a text file and displays it on the screen with a mouse-interactive interface as shown in Figure 3.9. In this figure the small white circle inside red box is the mouse pointer and the red box is shown just for visibility or display purposes and it has nothing to do with bounding box size. Clicking on the centre of the person records the screen co-ordinates along with the current image name. Three main pieces of information about each image are recorded: these are image name, centroid position (x,y) of the person image and occlusion level. We here use three occlusion levels (25%, 50%,75%) on left, middle and right mouse click respectively. In the case of the normal person with no occlusion we press the left mouse button with keyboard key,"N". In case of multiple persons per image we need multiple clicks per image. For example, in the case of three persons we use three mouse clicks on that image to record the centroids of the person and left, right or middle mouse button decides the occlusion level. The next image is loaded on by pressing the space bar button of the keyboard while the ESC button of the keyboard is used to terminate the application.

3.3 Learning and Classification

In this section we briefly discuss about learning and classification techniques. As our problem is binary classification, therefore here we just used two simple methods which are K Nearest Neighbour (KNN) and Support Vector Machine (SVM).

3.3.1 K Nearest Neighbour

KNN [83] is a simple and non-parametric method for classifying objects. KNN performs classification based on similarity measures (e.g distance). We could use different distance measurement schemes such as Euclidean, Manhattan/City block distance, Hamming distance, and others. KNN is also called the 'supervised learning algorithm'. The training data are in the form of vectors and an associated class label. This class label could be (+ and -) or (-1 and 1). By providing a number K, which could be any integer

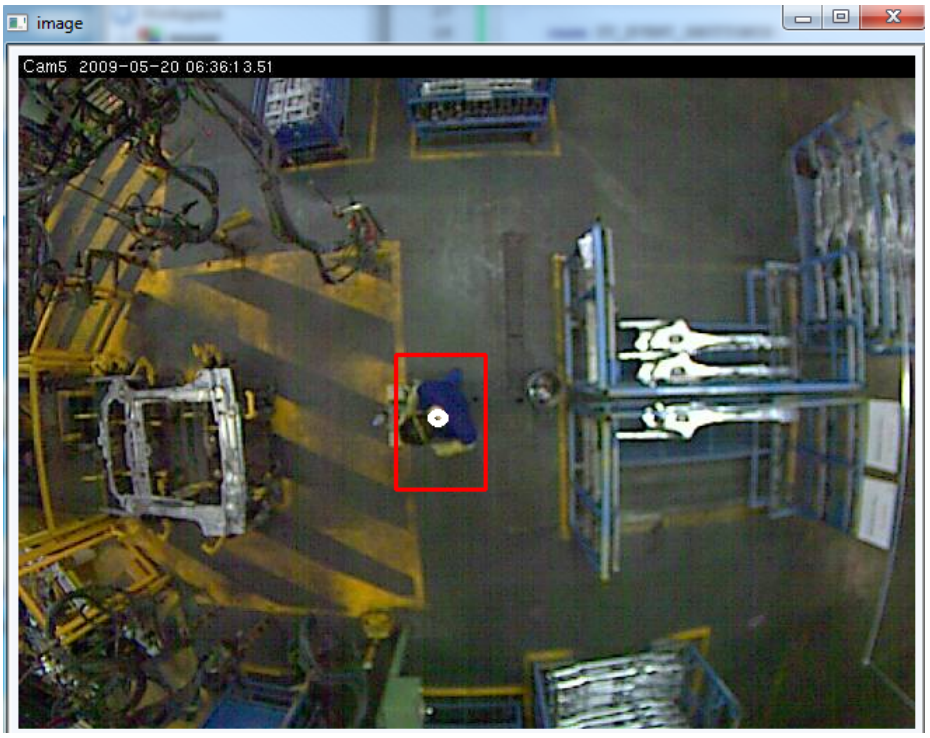
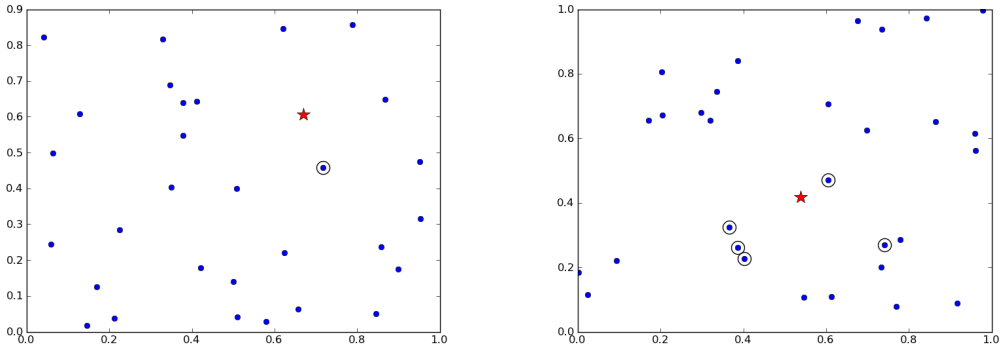


FIGURE 3.9: An application for image annotation: The interactive red box with the white circle in is the mouse pointer. On mouse click it records the coordinates of the screen with occlusion information and name of the image.

(1,2 etc), this number decides about the number of neighbours based on distance metric to influence the classification. If the number of classes is 2, then K is usually odd. When K=1 then it means simply nearest neighbour. A simple example of using KNN is demonstrated in Figure 3.10. In this figure we use K=1 and K=5 in the first and second examples respectively. The blue dots represent distribution of data. The red star in this case is the test datum. The nearest neighbour is shown in Figure 3.10(a) while five nearest neighbours are shown in Figure 3.10(b) as points in a small circle. It first decides the nearest neighbour on the basis of the distance of test data from all other points and then associates a class name to this test point. KNN is considered as a slow or lazy algorithm because it does not use any model or training data points to do any generalisation. Instead it keeps all the data in main memory and during testing it needs all the data stored in main memory. This means that KNN performs decisions on the basis of all training and test data in the memory. Having more data reduces the processing time.

3.3.2 Support Vector Machine

The Support Vector Machine [57, 11] is a binary classification algorithm developed by [108]. It is a supervised learning technique with a special property to simultaneously minimise the classification error and maximise the geometric margin; hence they are



(a) K=1 (b) K=5

FIGURE 3.10: An example of a KNN classifier.

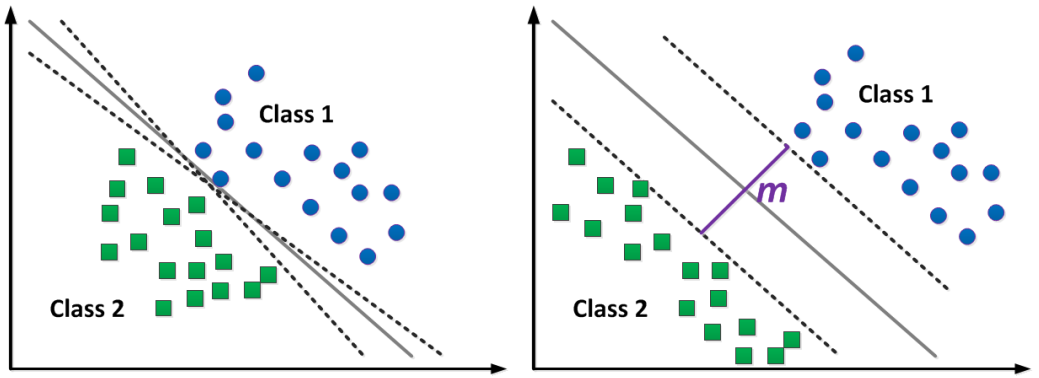
also known as maximum margin classifiers. SVM works well for classification due to the following properties and advantages:

- It can cope with high dimension input space.
- It assumes that the input space contains few irrelevant features.
- It uses a subset of training data in the decision boundary called support vectors. So it is memory efficient.
- It is versatile. One can use different kernel functions according to the nature of data for the decision function.
- The SVM is a simple and easy to use tool and has a good generalisation performance.

We can have many decision boundaries for a two-class, linearly separable classification problem as shown in Figure 3.11(a) which is not considered as good as it is in SVM. We should maximise the margin m as shown in Figure 3.11. The decision boundary should classify all the points correctly.

The SVM has a regularisation parameter c that controls the trade-off between margin maximization between classes and minimization of the misclassification error on the training data. There are different options available for choosing a kernel K in SVM. They are listed below:

- Linear Kernel: $K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i^T \mathbf{x}_j) + c$
- Polynomial Kernel: $K(\mathbf{x}_i, \mathbf{x}_j) = (1 + (\mathbf{x}_i^T \mathbf{x}_j))^d$
- Radial Basis(Gaussian) kernel: $K(\mathbf{x}_i, \mathbf{x}_j) = \exp(\frac{-1}{2\sigma^2}\|\mathbf{x}_i - \mathbf{x}_j\|^2)$
  or $K(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\alpha\|\mathbf{x}_i - \mathbf{x}_j\|^2)$



(a) Example of bad decision boundaries. (b) Example of Large margin decision boundary.

FIGURE 3.11: An example of Linear SVM. (based on [64])

- Sigmoid Kernel: $K(\mathbf{x}_i, \mathbf{x}_j) = \tanh(\gamma(\mathbf{x}_i, \mathbf{x}_j) + \phi)$

The Linear Kernel is the simplest kernel function of SVM. It is the inner product of data points with an optional constant c. The rest of the kernels are mostly used in non- linear features space. Each of these has some parameters. For example, in the Polynomial kernel, d is the polynomial degree or dimensionality of the space in which we intend to map the points. Similarly in the Gaussian kernel there is adjustable parameter sigma σ which has a vital role in the performance of the kernel. If it is overestimated then the exponential will behave almost linearly and in such a situation higher dimensional projection could start to lose its non-linear power. On the other hand, if the sigma is underestimated, the kernel function will lack the regularization and in that case the decision boundary will be highly sensitive to the noise in the training data. The Sigmoid kernel is also known as the Hyperbolic Tangent Kernel or the Multilayer Perceptron (MLP) kernel. It has the parameters slope γ and the intercept constant ϕ. A common value for the parameter γ is 1/N, where N is the data dimension. The effectiveness of the SVM depends on the selection of the kernel, kernel parameters and soft margin c.

3.4 Tuning the Parameters of the HOG Algorithm

In this section we discuss different parameters of the HOG algorithm. As discussed earlier the nature of data used in the conventional view is different from the overhead view data. So different parameters need to be tuned before applying HOG. Here we assume each parameter is independent of others and for the purpose of comparison we start with the default parameters used by the original author, except we use a 64×96 pixel window. We chose this size by visual inspection. These parameters are shown in Table 3.1.

To evaluate optimum value of each parameter we randomly chose 6000 manually labelled

TABLE 3.1: Default HOG parameters.

Gradient mask	Bins	Block size	Cell Size	Norm	Magnitude	Block overlapping
1-D centred	9	16x16	8x8	Yes	Yes	Yes

images with people which were split into equal numbers of training and test. These labelled images with people in are referred to as positive samples or positive detection windows. Each of these image has only one person. The same number of negative samples or detection windows are chosen randomly from person-free images. We test the effect of variation of these parameters over the overall classification performance. The results of this binary classification are referred to as True Positive Rate (TPR) and False Positive Rate (FPR). The selection of different parameters except for the gradient mask and choice of the classifier is made using the Detection Error Trade-off (DET) curve. A DET curve [73] is a graphical plot of error rates for binary classification systems, plotting Miss rate vs. FPR. Another similar graphical tool is the Receiver Operating Characteristics (ROC) curve which represents the trade-off between TPR and FPR. TPR, FPR and Miss Rate are defined as follow:

$$\text{TPR} = \frac{\text{Correct Classification}}{\text{Total Positive Samples}}$$

$$\text{FPR} = \frac{\text{Incorrect Classification}}{\text{Total Negative Samples}}$$

$$\text{Miss Rat} = 1 - TPR$$

We plot FPR on the x-axis and Miss-Rate on the y-axis. The FPR on each point on the x-axis which is associated with training data starts with 6000 person-free samples and increases with a step size of 1000 additional samples. An example of DET curve can be seen in Figure 3.13.

The selection of a classifier and gradient mask is done by 2-fold cross validation on the basis of best classification results. The reason is that DET curve is used to set a threshold or to select a trade-off between TPR and FPR.

3.4.1 Selection of a classifier

We start classification with Nearest Neighbour (NN) and Support Vector Machine (SVM). The code for NN classifier was written in python while for SVM we use Lib-linear SVM [39]. Table 3.2 shows the result of NN classifier and SVM with different kernels. We use the default parameters of SVM for all the kernels. From the table it can be seen that using a linear SVM gives a better classification score compared to other kernels and NN. The highest TPR was obtained using a linear-SVM kernel. Comparing Linear SVM with NN, the overall results of both are the same: i.e, using NN leads to a decrease

of one percentage point in TPR, but also reduces FPR by one percentage point. The same trend exist in Linear SVM where we achieved an increase of one percentage point in TPR with FPR as well. The reason for selecting linear SVM is that it is computationally faster than other SVM kernels and NN.

TABLE 3.2: Comparison between NN and SVM classifiers using 2-fold cross validation.

	Fold1		Fold2	
	TPR	FPR	TPR	FPR
Nearest Neighbour	0.9	0.08	0.9	0.08
SVM- RBF	0.88	0.11	0.89	0.1
SVM- Poly	0.88	0.1	0.84	0.08
SVM- Sigmoid	0.88	0.12	0.92	0.13
SVM-Linear	0.91	0.09	0.91	0.09

Linear SVM has a cost parameter c (misclassified error) which needs to be tuned. We use different values of c in the range 0 to 2 and chose 0.015 on the basis of classifier best performance as shown in Figure 3.12. It can be seen that, at the chosen value, the classification rate is about 92%.

We further tested the classifier using linear SVM and $c = 0.015$ with scaling. Table 3.3 shows the results using with and without scaling. For scaling the data were normalised into range 0-1. It can be seen that scaling does not improve the performance of the classifier.

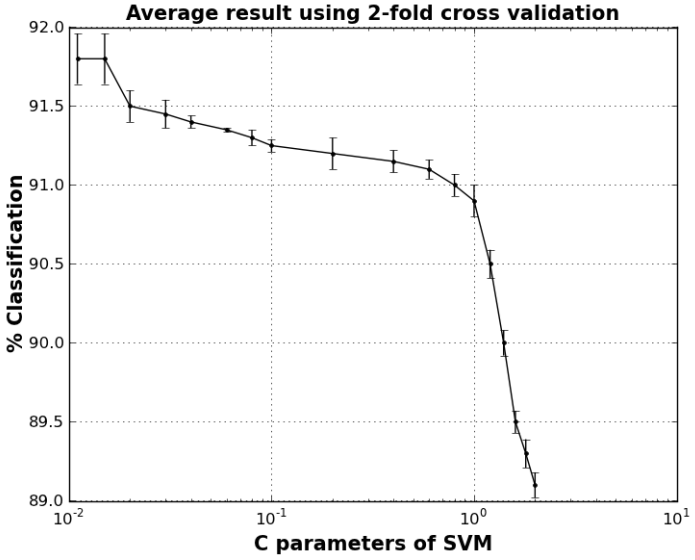


FIGURE 3.12: Selecting optimum value for cost parameter of SVM

In the rest of our experiments we use Linear SVM with $c = 0.015$ as a classifier.

TABLE 3.3: Performance of linear SVM with and without scaling

	Fold1		Fold2	
	TPR	FPR	TPR	FPR
Without Scaling	0.92	0.08	0.92	0.07
With Scaling	0.91	0.08	0.91	0.07

3.4.2 Selection of a gradient mask

We use following four types of horizontal and vertical gradient masks:-

- 1-D centred: $G_X = \begin{bmatrix} -1 & 0 & +1 \end{bmatrix}$, $G_Y = \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}$
- 1-D un-centred $G_X = \begin{bmatrix} -1 & +1 \end{bmatrix}$, $G_Y = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$
- Sobel 3x3 $G_X = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix}$, $G_Y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix}$
- Schar 3x3 $G_X = \begin{bmatrix} -3 & 0 & +3 \\ -10 & 0 & +10 \\ -3 & 0 & +3 \end{bmatrix}$, $G_Y = \begin{bmatrix} -3 & -10 & -3 \\ 0 & 0 & 0 \\ +3 & +10 & +3 \end{bmatrix}$

Using 2-fold cross validation, we found that the 1-D centred mask gave better results than others. Using a simple 1-D mask is computationally effective. We used a Sobel 3x3 mask which also performs smoothing, but the results shown in Table 3.4 suggests that using a complex mask or smoothing slightly reduces the performance. [29] also reported in their paper that using a simple 1-D mask with no smoothing gives a better performance. Using a complex masks and smoothing (Sobel Filter) degrades the performance of detector. A reason could be because the fine details of image are important and smoothing may decrease the edge contrast.

TABLE 3.4: Performance: Using different gradient masks

	Fold1		Fold2	
Mask	TPR	FPR	TPR	FPR
1-D centred	0.92	0.08	0.92	0.07
1-D un-centred	0.91	0.1	0.92	0.1
Sobel (3x3)	0.9	0.08	0.92	0.08
Schar(3x3)	0.9	0.08	0.92	0.08

3.4.3 Number of bins in histogram of angles

The cell is an 8x8 pixels region of the detection window. In each cell we quantise the gradient orientation voted with gradient magnitude into a number of bins. Each pixel in a cell has an angle value between (0-180, unsigned) or (0-360, signed). The number of bins per cell decides the range of angle per bin. For example, six bins with angle range 0-180 degrees means that first bin could accommodate angles between 0 and 30 degree and the last or sixth bin could accommodate angles between 160 and 180 degrees. Similarly, using six bins with unsigned angles between 0 and 360 degrees would mean the first bin could accommodate angles in a range between 0 and 60 degree and the last or sixth bin can have angles in the range 301 and 360 degree. Also increasing the bin size causes an increase in the feature vector size. We use three different sizes (6,9 and 18) of bins with signed and unsigned angles.

The results are shown in Figure 3.13. It can be seen that decreasing the bin size from 9 degrades the performance of the detector. Increasing the bin size from 9 to 18 doubles the feature vector size but does not improve the performance significantly. Overall, 9 bins gives the best performance, which validates the original author's reported results about number of bins. Here the quantisation of the gradient orientation (0-180, unsugned) into 9 bins is called the local histogram of oriented gradient.

The conclusion is that, as reported by the original author, the fine orientation coding is crucial for good performance; however increasing the bin size beyond 9 does not affect the results significantly. Furthermore, using unsigned gradients opposed to signed ones is seen to be more discriminative.

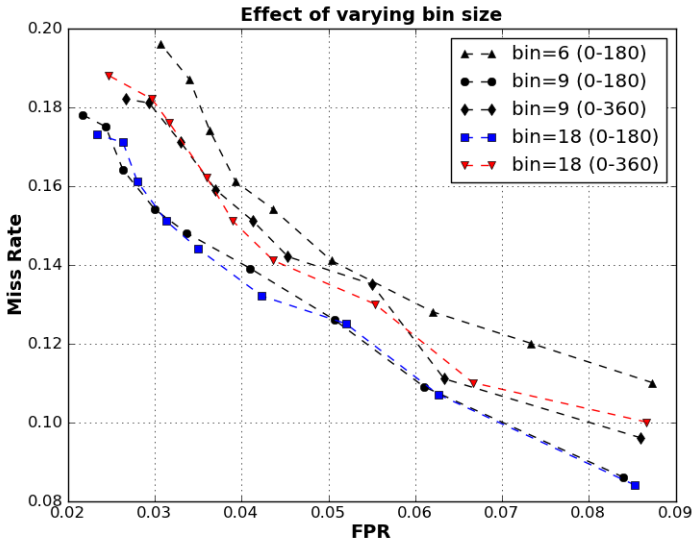


FIGURE 3.13: Bin Size: Increasing or decreasing bins from nine bins effects the performance. Using unsigned gradient improves the result.

3.4.4 Block normalisation

A block is a 16x16 pixels region of an image. Each block has four cells and local weighted histogram of oriented gradient is calculated at 8x8 pixel cell. Here weight means gradient magnitude. Gradient orientations are not affected by the intensity. Gradient magnitude lacks invariance to changes in illumination and foreground-background contrast. Therefore, once all of the cell histograms within a block have been obtained, they are normalised due to the variance of gradient strengths over a wide range. Hence we group cells into larger, spatially connected blocks and normalise each block separately. Once this block normalisation has been performed, all of these local histograms can be concatenated in a single feature vector.

Figure 3.14 shows the result with and without doing normalisation. It can be seen that using normalisation not only significantly improves the detection rate but also reduces the false positive rate. For example, at the starting point, the detection rate without normalisation is 0.85 and FPR is 0.13 while by the same data using normalisation TPR is about 0.92 with a FPR 0.08. This mean normalisation improves the detection rate with seven percentage points and reduces FPR by five percentage points. This is a significant improvement in overall results, and validates the argument of the original author [29]; that normalisation is an important factor for detection performance.

The conclusion is that the block normalisation is an important step of the HOG algorithm. As the gradient magnitude is not invariant to illumination, therefore, to equalise the luminance among adjacent cells of block, we perform normalisation. The results shown in Figure 3.14 provide the evidence that block normalisation can reduce the ill effect of illuminations and produce a better performance.

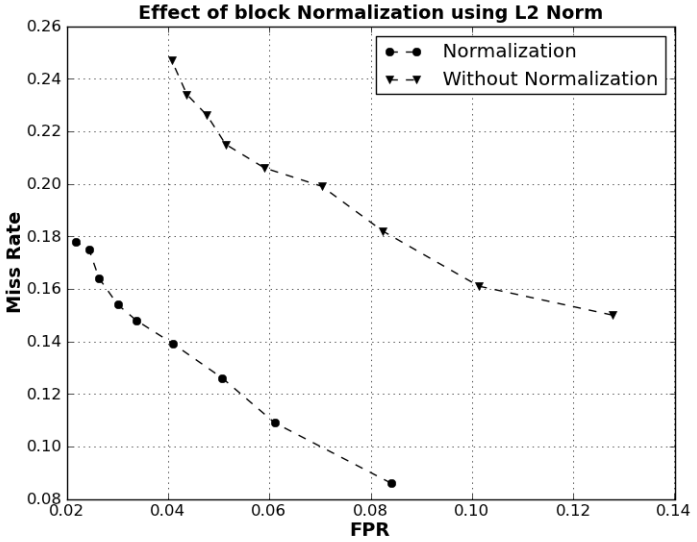


FIGURE 3.14: Block Normalisation: Normalisation significantly improves the performance of detector.

3.4.5 Gradient magnitude

During assigning gradients into bins, another important factor is to add gradient magnitude as well. The reason is that the gradient orientations are not affected by the pixel's intensity. Two low-and hight-intensity values can have the same angle but their magnitude will be different and orientation with larger magnitude is important. Therefore, adding magnitude with gradient orientation into cellular bins seems to be more robust than without adding the gradient magnitude. The results are shown in Figure 3.15. It can be seen from Figure 3.15 that not adding weights not only almost doubles the false positive rate but also increases the misclassification rate. On the basis of these results we can state that adding gradient magnitude and block normalisation makes the feature vector much more resistant toward changes in illumination.

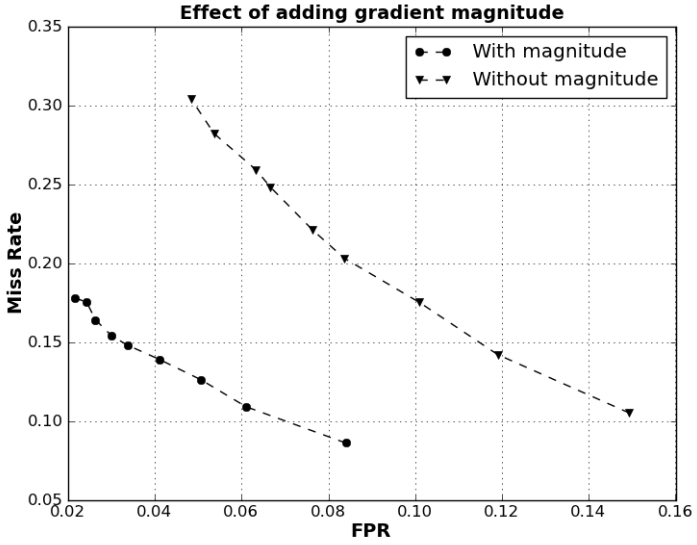


FIGURE 3.15: Effect of adding gradient magnitude: Adding magnitude significantly improves the overall performance of the classifier.

3.4.6 Block overlapping

Block overlapping is also an important parameter of the HOG algorithm. When overlapping the blocks we use some redundant information many times depending on the overlapping factor. For example, using a step size of eight pixels means an overlapping factor of 1/2 or 50% overlapping of pixels. This situation has already been discussed in Section 1.2 and shown in Figure 3.6. With 50% overlapping of pixels, blocks are visited from left to right and top to bottom when forming the final descriptor.

The results of different block overlapping factors are shown in Figure 3.16. Starting with stride 0 (No Overlapping) to a factor of 1/2 increases the overall performance of classifier. The advantage of no overlapping is that it results in a very low sized feature

factor but it reduces the overall performance considerably. Without using overlapping, the FPR increases almost three times more than if using the overlapping block. The miss rate also decreases after introducing overlapping blocks.

Overall, 50% block overlapping gives better result best results, which is again justifiable by the results produced by the original author.

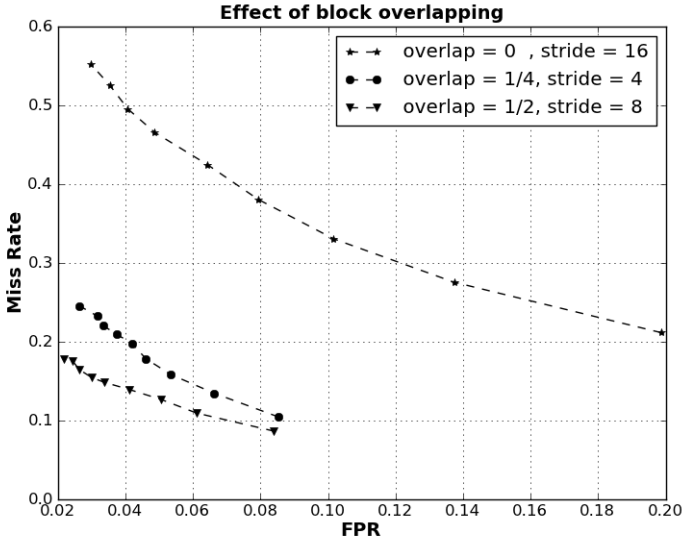


FIGURE 3.16: Effect of Block overlapping: Increasing overlapping from 0 to 1/2 increases overall performance considerably.

3.4.7 Selection of bounding box size

In our overhead view dataset the size of person varies depending on the distance from the optical centre. The person looks more square just below the camera in the centre of the image. The size of the person gradually increases the further away they are from the centre. Another noticeable fact is that, apart from size change, there is also change in the orientation of the person which also depends on the radial distance from the image centre and their position in the image. So using a fixed size bounding box doesnot necessarily work better. We therefore use a compromise 64×96 and 96×96 pixels sized bounding box. The reason for choosing the two sizes is that the average person's size measured is 64×96. The justification for using a bounding box of size 96×96 pixels is that this size can accommodate all possible different orientation of the person. We therefore perform the experiment with these two different bounding box sizes. The results are shown in Figure 3.17. These results suggest that the using square box gives a better performance then using the rectangular box.

In this section we tuned different important parameters of the HOG algorithm. We have proved experimentally that unsmoothed simple 1-D centred mask, fine orientation

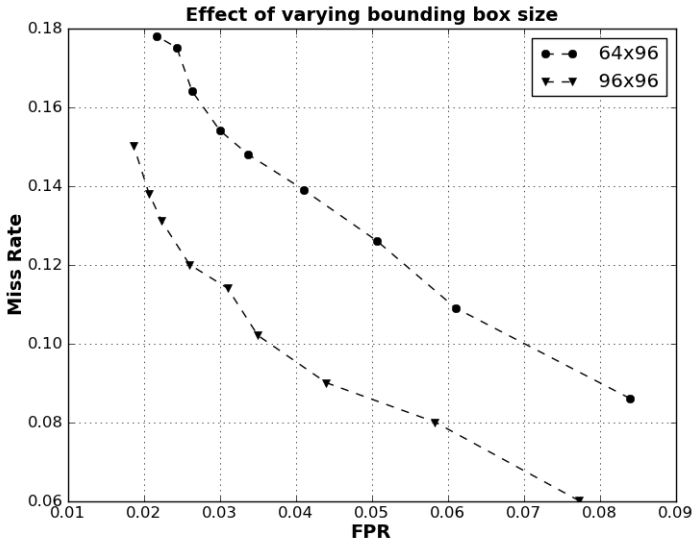


FIGURE 3.17: Effect of Bounding Box size: Increasing the bounding box size from rectangular to square increases the overall performance.

voting, coarse spatial binning and overlapping blocks with strong normalisation is necessary for good performance. We also have shown that except for window size, the rest of the parameters are consistent with the chosen parameters reported by the original author. In the next section, we then use these parameters to see the effect of classification performance by varying data samples in both training and test sets respectively.

3.5 Results of the Person Classification

We use the tuned parameters and found that all the parameters were consistent with the default parameters quoted in the paper by the original author except for the window size. Although the nature of our overhead view data is different from that for the normal view, even then the tuned parameters are the same as those used for person classification using normal view. This is not surprising because the original author in [28] used 10 different classes including Dog, Cat and horse and found that the HOG descriptor is equally suited for these object classes and key HOG parameters are mostly very similar to these classes as well.

In the implementation of the HOG algorithm we chose all these parameters as described in Table 3.5: i.e, a bounding box size of 96×96 pixel with a block size 16×16 pixel and using block overlapping, and 2×2 cells per block, local histogram of each cell with adding gradient magnitude in 9 orientation bins in the gradient range 0-180 degrees. The reason for using the 96×96 window is that a person is not always upright and is in same orientation contrast with the conventional view. For each position a person could have a variation of rotation upto 360 degrees; which is why a square shaped bounding

box gave better results.

TABLE 3.5: Chosen HOG parameters after tuning.

Gradient mask	Bins	Block size	Cell Size	Norm	Magnitude	Block overlapping
1-D centred	9	16×16	8×8	Yes	Yes	Yes

We conducted three different types of experiment. In these experiments we just varied the training/ test size by keeping all parameters constant. The intention is to check the classification performance by varying different training and test sizes.

3.5.1 Effect of extra background training

In this experiment we randomly chose 3000 positive detection windows (image patches with people) and the same number of randomly chosen negative detection windows (image patches without people) from person-free images. The same number of samples is used for training and test set. The initial training model was tested using the test set. The result was a TPR of 0.94 with a FPR of 0.078. To reduce the FPR we provide an additional 8000 background samples with a step size of 1000 to our initial learning model. As a result there are eight additional training models are formed. Each of these models was tested using the test set. The results of each of the training models are shown in Figure 3.18. The final training model has a TPR of 0.85 with a FPR of 0.018. This means that providing additional background training reduces the FPR by six percentage points but with a loss of nine percentage points in TPR.

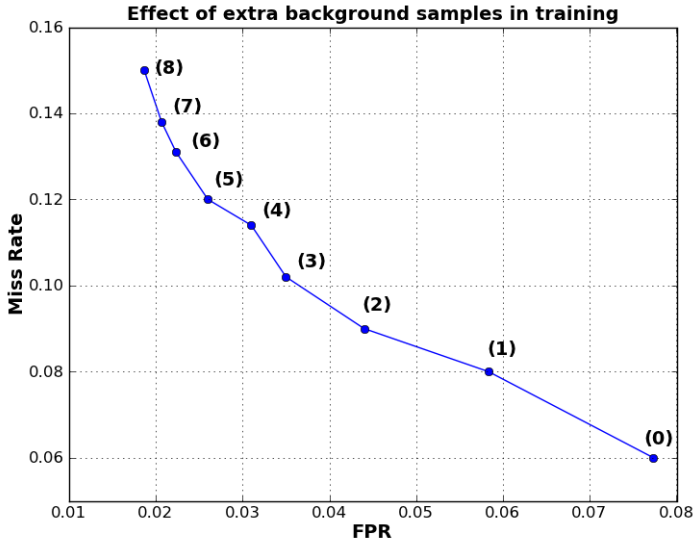


FIGURE 3.18: Effect of extra background samples in training. By providing additional background samples in the training, from right to left in this graph decreses FPR monotonically. The number beside the symbol is the number of additional training samples in unit of 1000.

3.5.2 Person Classification results using whole dataset

In this experiment we use the whole dataset and divide it into an equal number of positive and negative samples in both training and test sets. A 2-fold cross-validation process is then performed. The results are shown in Table 3.6. From this table we can see that there is no change in the classification results when the training and test sets are swapped. The overall result is a TPR of 0.94 with a FPR of 0.06.

TABLE 3.6: Results of 2-fold cross validation testing.

Fold1		Fold2	
TPR	FPR	TPR	FPR
0.94	0.06	0.94	0.06

3.5.3 Providing better training

In this experiment we divide the training and test set into 5:1 respectively, using the whole data set. The intention is to see the effect of providing more learning to the training model. The result was a TPR of 0.96 and FPR of 0.07. It can be seen that providing more samples to the training model resulted in a gain of two percentage points but with an increase of one percentage point in FPR as well. This means there is no significant change observed when increasing the number of samples in the training model.

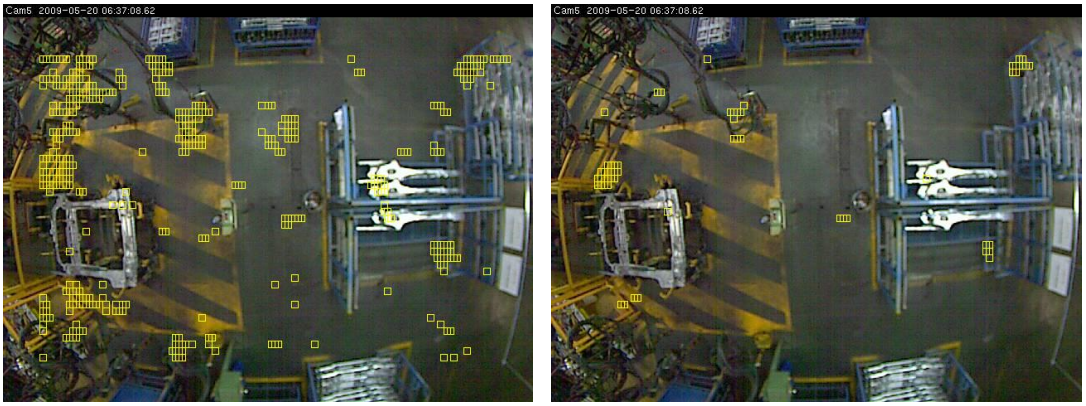
In this section we discussed implementation details of the HOG algorithm in conjunction with our overhead view data. We have experimentally shown that the HOG algorithm gives encouraging person classification results. The next step is to detect people in images by extending this classification task to the detection task.

3.6 False Classified Positions per Image

The main goal of this research is to detect people in images using the overhead view and the person classification is a pre-requisite for this. In the person classification, the role of the classifier is to decide whether the given detection window is a person or not. In the case of a full image, e.g a 640×480 pixels image, there can be multiple detection windows/windows per image. A person classifier is used to test each of these window for a person/no-person decision. In this section our main subject of discussion is to check the behaviour of a pre-trained person classifier (training model) while scanning the whole image.

During the Person Detection process an input image was scanned in linear order in both directions (horizontal and vertical) with a step size of four pixels horizontally and eight pixels vertically. This step size is consistent with the aspect ratio of a person shape,

where height of the person is about double that of width. Our 640 × 480 pixels images have about 6000 positions or centroids of detection windows or windows or bounding box per image. Each of these windows is tested by the pre-trained person classifier. The result for a typical classifier should be a low number of randomly spurious detections per image in the case of no person in image. In case of the person in the image there should be a compact cluster of detections around the position of the person in the image.



(a) Using Classifier-B (b) Using Classifier-A

FIGURE 3.19: False detections in person-free images using two different classifiers.

(a) Using Classifier-B (b) Using Classifier-A

FIGURE 3.20: False detections in image with a person using two different classifiers.

We chose two classifiers; the first, which described in Section 3.6.1, is a classifier with a TPR of 0.85 and FPR of 0.018 we refer to this from this point onwards as a Classifier-A. The second is described in Section 3.6.3 with a TPR of 0.96 and FPR of 0.07; we hereafter referred to as a Classifier-B. The reason for choosing these two classifiers is because the Classifier-A has a lower TPR but a better FPR, while the Classifier-B has a good TPR. We use 200 randomly chosen images with people and 200 randomly chosen person-free images. Person-free images are used to check FPR per image. We test these images using both classifiers. Using the Classifier-B, there were on average 450 falsely classified positions per image while on average there were 102 falsely classified positions per image when we used the Classifier-A. A sample person free image after plotting

these falsely classified positions are shown in Figure 3.19, while a sample image with a person after plotting all detections is shown in Figure 3.20(a). The falsely classified positions are shown in small yellow boxes. These spurious detections per image using both of these classifiers are not at random positions, these are mostly in the form of compact clusters. From these results we can say that using a classifier with a high TPR gives more detections around the position of the person in the image, as shown in Figure 3.20(a) along with a number of compact clusters of spurious detections. The noticeable thing is that even using a classifier with a low FPR, a few compact clusters can easily seen in Figure 3.19(b) and in the case of the person in the image as shown in Figure 3.20(b), the sizes of these spurious compact clusters are even greater than the actual detection of a true cluster around the person.

Discussion

In this chapter we use the HOG algorithm for Person Classification and People Detection in images. Although the classification results were promising, while scanning the whole image to detect people in the person-free test images, the performance was un-acceptable. Another finding is that according to the geometry and perspective of the imagery when using our data, this suggests that using a bounding box with different orientation and size with respect to perspective centre might be a better idea to avoid mismatch which could cause a high number of false detections per image.

Chapter 4

The Rotated HOG Algorithm

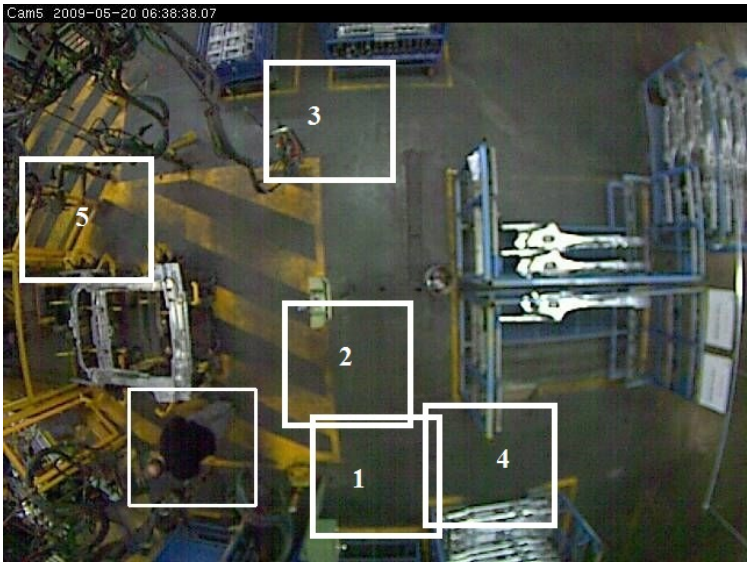
We used the Standard Histogram of Oriented Gradient (sHOG) algorithm and our overhead view dataset for person classification and detection. The classification results were good but the results of detecting people in person-free images produced numerous spurious detections per image. In this chapter we first demonstrate the effect of a bad bounding box match. We also explain how a person's size and orientation depending upon their position varies with respect to the centre of the image. To overcome this artifact we propose a novel technique which uses variable size bounding boxes with different orientations with respect to the optical centre. After discussing the algorithm and its performance we compare it with the sHOG algorithm as discussed in Chapter 3. The main conclusion is that our proposed algorithm achieves better results than sHOG.

In Section 4.1 we demonstrate the effect of a bad bounding box match. In Section 4.2 we explain, using examples, how we can align the person vertically using geometric transformation. Our proposed algorithm is discussed in Section 4.3 which uses oriented bounding boxes. We compare the average person image of the overhead view dataset before and after geometric transformation with the average person image of the conventional view dataset. In Section 4.4 we explain, with different examples, the bounding box size selection procedure. The person classification and false detections per image results are discussed in Sections 4.5 and 4.7 respectively. In Section 4.6 we discuss different levels of occlusion in our data and how this affects overall classification results. In Section 4.8 we discuss attempts made to reduce the complexity of the algorithm. The last section is about per detection window processing time using the sHOG and our proposed algorithm.

4.1 Mismatch of the Bounding Box containing a Person

In the previous chapter we used the Standard Histogram of Oriented Gradient algorithm to detect people in images. Although this algorithm gave good classification results, the

performance was not satisfactory while scanning the whole image. We hypothesise that this is because of bounding box mismatch. To demonstrate this we use an example by taking a sample image as shown in Figure 4.1(a). This is a test image where the person is shown in an unlabelled white square box. The same-sized bounding boxes are in the training set. Some positive training samples at different locations of the image are labelled as 1-5 and their associated samples of people are shown in Figure 4.1(b). While analysing even visually, one can easily note that the pose and orientation of the test image are significantly different from the training images. The test image has a very weak co-relation with sample training images in term of a person's orientation. Almost all of these images have completely different silhouettes. It is obvious that the feature vector generated using HOG would also be different, which causes the dissimilarities for learner and classifier.



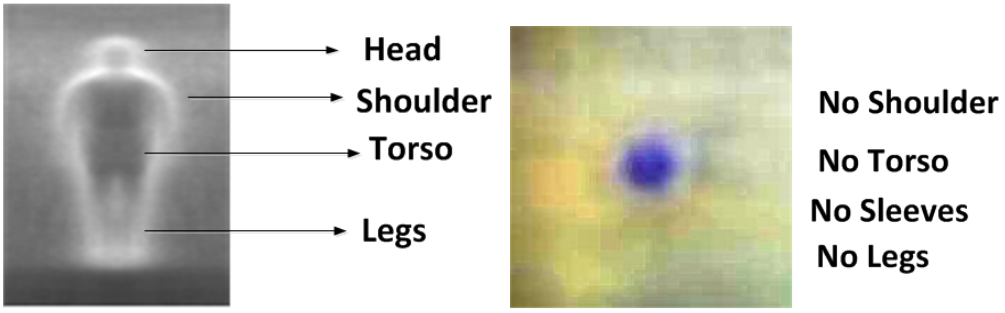
(a)

(b)

FIGURE 4.1: Bounding box mismatch: The bounding box of the test image is shown in the unlabelled white rectangle with a person in the first image. The bottom row images are corresponding training samples at different positions labelled as 1-5.

We compare the average image of a person using both the conventional and overhead views as shown in Figure 4.2. Here, by conventional view, we mean that the images of people are taken from a non-overhead view camera but the camera is usually parallel or horizontal to the person viewed. Furthermore an assumption of an upright visible person is also made in these images. The average person using the conventional view is shown in Figure 4.2(a) with a very clear structure of a person. All the major parts of

the human body like head, shoulder, torso and arms could be seen easily. Furthermore the background around the person is washed out. The main reason is that in all of these types of image the person is visible and at upright or vertical pose with head always on the top and feet always on the bottom.



(a) Conventional View (from [29]). (b) Overhead view.

FIGURE 4.2: Average Person: This figure shows the average person using the conventional view and the overhead view.

If we look at the average person using the overhead view as shown in Figure 4.2(b), it can be easily seen that that there is nothing visible that resembles a person. We can only see a circular-type shape with no other clues of major body parts which could represent a human. The background instances can also be seen around an average person. It looks more like background then a person and the shape of the person is also missing. We cannot see the major parts of the human body like the shoulder, torso, head or legs. We believe the reason for this could be because of the different orientation of the person. In contrast to the conventional view we are not using any assumption or restriction about the pose of the person. In our dataset a person can have a large variation of poses with different orientations from 0-360 degrees. Some examples can be seen in Figure 4.1(b).

FIGURE 4.3: Fixed sized bounding box: The person only occupies less than 50% of the area and the rest is background.

The reason for more background than structure of the person in the average person image could be because of using a fixed size bounding box. A sample example of a fixed sized bounding box can be seen in Figure 4.3. Here, for display purposes, we use a white rectangle to show the person. The person in this image has occupied less than 50% of the total area of fixed sized bounding box while the rest is the background. This

means that in this bounding box the classifier is learning more than 50% about the background. This redundant un-necessary background information might make it hard for the classifier to discriminate between a person and a background.

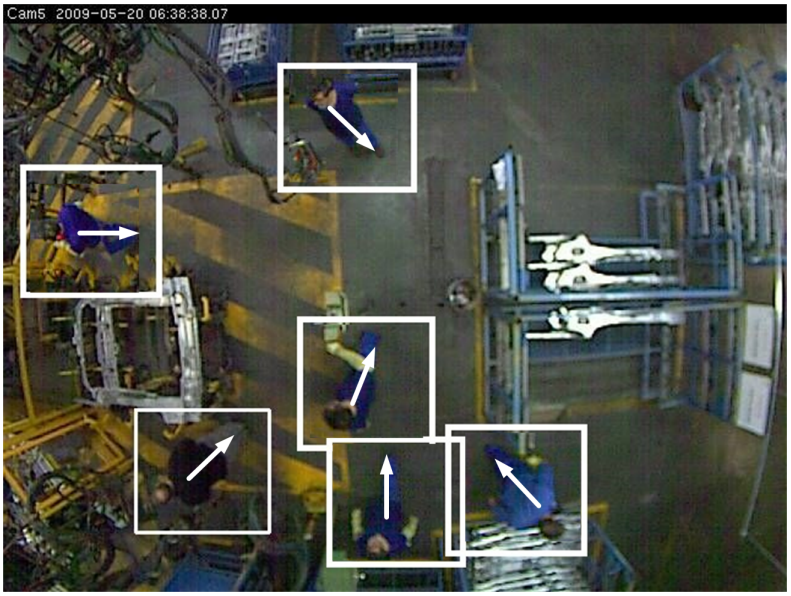


FIGURE 4.4: The orientation of the person shown as a white arrow drawn on the person. Each of this orientations is towards the centre of the image.

In the overhead view different orientations of a person at different positions are caused by the property of the overhead wide angle camera. There is a radial symmetry of orientation towards the optical centre (centre of the image), which is shown in Figure 4.4. In this figure we plot the orientation of the person as a white arrow. The arrow head is aligned with the feet of the person while its head is aligned to the head of the person in the image. If we see the overall effect of these arrows we can easily say that all of these arrows are pointing toward the centre of the image. In this example, each arrow is making a particular angle from the origin (image centre). This means the orientation can be measured with respect to the centre of the image. This clue suggests that we can use the oriented bounding box on the basis of the position of the person in the image with respect to the origin.

4.2 Alignment of the Orientation of the Person

In this section we discuss the use of geometric transformation to align the person at the up-right/vertical orientation. We demonstrate this using a sample image as shown in Figure 4.5 with a person having a fixed sized and fixed orientation bounding box shown in white colour. If we rotate the image along with the centre of projection (we assume this to be the centre of the image) and then translate it towards the centre of the image, the resultant image is shown in Figure 4.6. The fixed sized bounding box after rotation is shown in this figure where the person is at an upright orientation. If we rotate all

other people at different positions in the images they would always be at upright or vertical orientation.

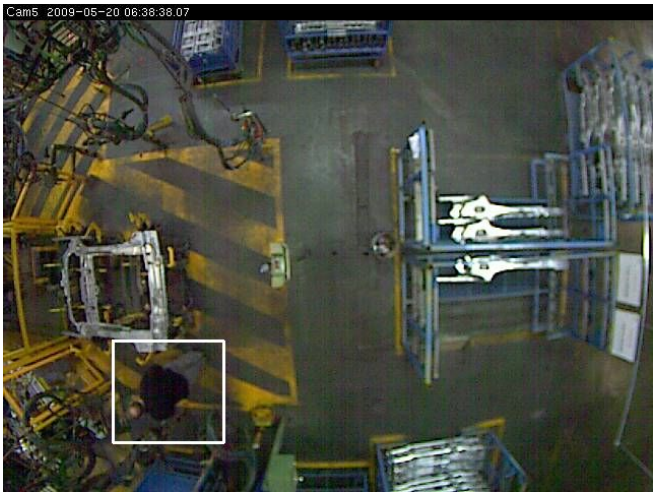


FIGURE 4.5: A sample image with a person. A fixed sized and orientation bounding box is shown in white colour.

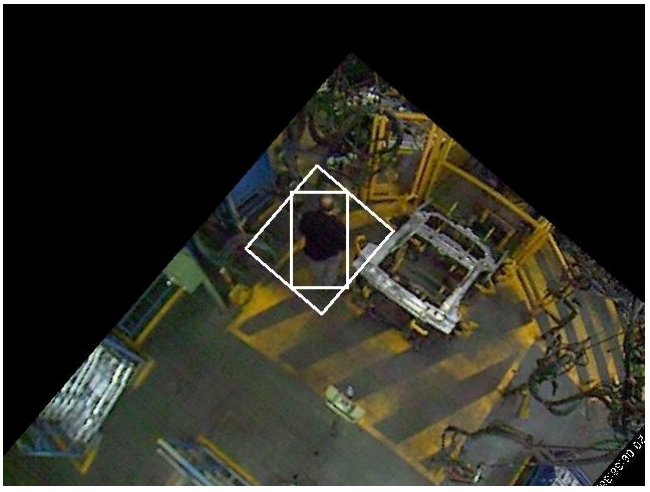


FIGURE 4.6: Image after rotating the centroid of the person along with the centre of rotation of the image. This rotation causes the upright or vertical orientation of the person.

This suggests that we can remove the artifact of different orientations if we rotate the person along with the centre of the image. An appropriate sized bounding box can then be cropped out using the known centroid of the person. We can then apply inverse transformation to measure the four co-ordinates of the bounding box in the image before rotation. The resultant image after applying inverse transformation is shown in Figure 4.7. Here we can see the original static-sized bounding box in the white square and our proposed bounding box which is exactly oriented along with the person's orientation. Another noticeable aspect is that our proposed rectangular bounding box occupies only the region around the person and does not contain un-necessary background information. Applying this technique to the whole dataset, the resultant average person's image is shown in Figure 4.8.

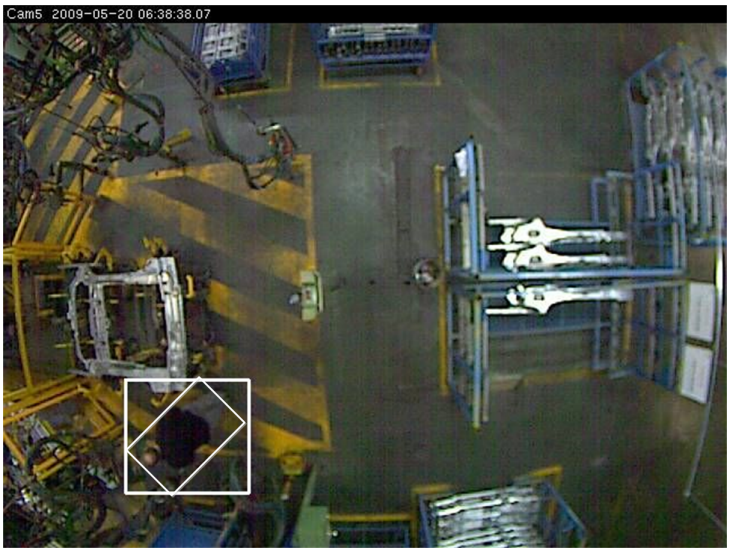


FIGURE 4.7: The resultant image after applying transformation and inverse transformation to the original image. The rectangular box is exactly oriented along with the person's orientation and not taking un-necessary background information.

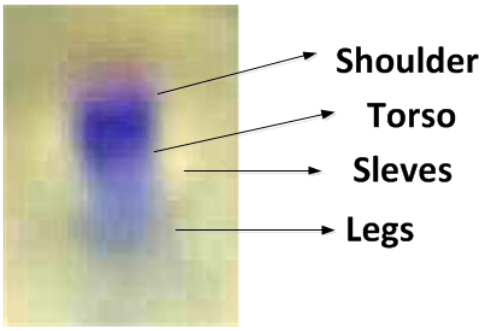


FIGURE 4.8: Average Person after applying rotation.

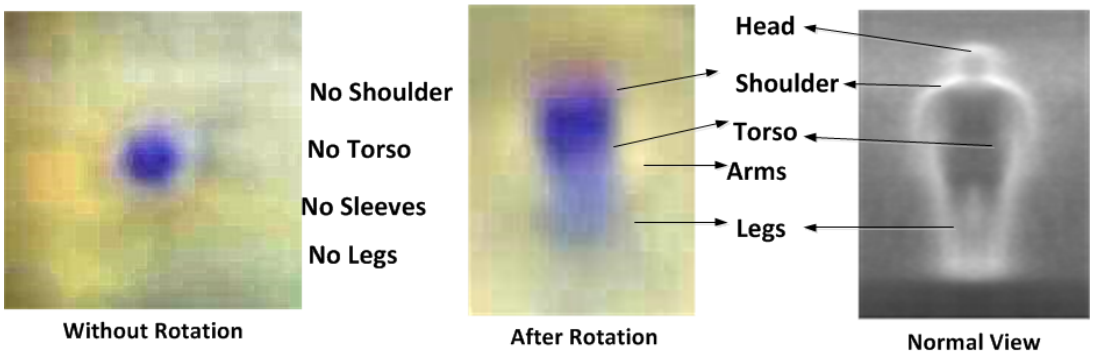


FIGURE 4.9: Comparison of average images: This figure shows the average person image using overhead view (first two) and the normal view.

In Figure 4.9 we compare three average images of the person. In this figure the first two images (before and after rotation) are based on overhead view data while the third is based on normal view data. It can be seen that compared with the first image the second has more structure, like a person, and resembles to some extent the normal view average image. We can also see some evidence of legs and arms. The torso and upper

body can also be seen more clearly than shown in Figure4.2(b). The head of the person in this average image is not prominent because of too much variability. In our overhead view images people have worn uniform with short-sleeved shirts, which is why we can see arms as a skin colour in this average image.

4.3 Working of Rotated Histogram of Oriented Gradient Algorithm

Our proposed Rotated Histogram of Oriented Gradient (rHOG) algorithm for person detection from the overhead view is explained step by step, as follows:

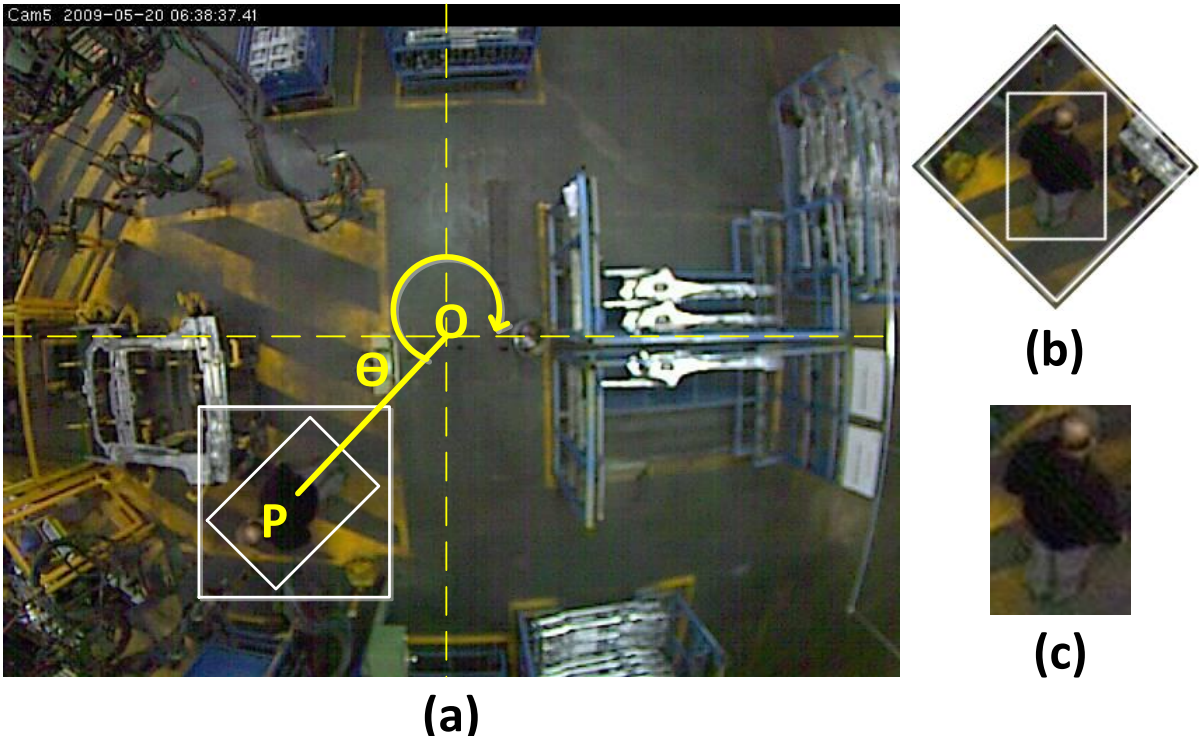


FIGURE 4.10: The rHOG algorithm in detail.

In Figure 4.10(a) the point O is the centre of rotation and P is the position of the person.

1. The first step is to calculate the geometric transformation, required to standardise the bounding box. This is calculated as follows:

   (a) Translate the position P to the origin O using matrix $S(x, y)$.

   $$S(x,y) = \begin{bmatrix} 1 & 0 & -|P_xO_x| \\ 0 & 1 & -|P_yO_y| \\ 0 & 0 & 1 \end{bmatrix}$$

   (b) Rotate the coordinate system by the angle θ, using rotation matrix $R(\theta)$.

$$R(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

This is the angle subtended by the line OP and the x-axis. It can be calculated as $\theta = \arctan\left(\frac{P_y - O_y}{P_x - O_x}\right) + \pi/2$. The orientation of the angle is clockwise.

(c) The composite transformation T is:

$$\mathrm{T} = S(x, y) * R(\theta)$$

Thus the new position of the point P is now TP and this is at the origin O. This is shown in Figure 4.10 (b).

2. The next step is to check that the projected standard bounding box lies totally within the image.

   (a) Determine appropriate size bounding box vertices $(v_0, ..v_3)$.

   (b) Perform the inverse transformation on all of these four vertices i.e $\bar{v}_i = \mathrm{T}^{-1}v_i$ for $i = 0..3$

   (c) Check that all the vertices lie within the original image. If any of these vertices lies outside of the image then that position can not be processed. This ensures that the bounding box always contains valid image data.

3. (a) Take a square sub-region centred at P such that the bounding box is exactly contained. An example is shown in Figure 4.10 where the square size region is shown in the white box.

   (b) This sub-image is rotated by angle θ using the warp-Affine() function of OpenCV with inter-linear interpolation.

   (c) The sub-region is now oriented parallel to the original image and the bounding box can be cropped from this. An example of the chosen bounding box is shown in Figure 4.10 (b). The bounding box is shown as a white rectangle in this image. The final cropped bounding box is shown in Figure 4.10 (c). Thus, before and after rotating the image region or bounding box containing the person, there is a constant shift of gradient. HOG features can be generated from this bounding box.

Applying rotation always results in an upright orientation i.e the person always appears aligned with the new vertical axis if we rotate and translate the image patch to the centre of the image. Some sample images after applying rotation and translating towards the centre of the image are shown in Figure 4.11.

FIGURE 4.11: Orientation of people in images before applying rotation (row1) and after applying rotation (row 2)

4.4 Bounding Box Size Selection

Using the overhead view with a camera having a wide range lens the size and shape of an object varies with respect to the centre of the camera. Here we demonstrate this effect with the help of two simple examples. In the first example we demonstrate this behaviour using a general object i.e a box in this case while we use SCOVIS data and observe the size change of a person in the second example.

4.4.1 Observing a general object (Box) size

In this experiment we chose a box of a 11.4 cm width, 9 cm depth and 23 cm height. The red top face and green+ white at side face of the box as shown in Figure 4.12, are used to facilitate the measuring of sizes in the images. The intention of this experiment is to see how this rectangular box appears when viewed from an overhead camera with wide angle lens and when placing the box at different positions from the position just below the camera to the edge of the scene. We use a Point Gray camera with a Fujinon CCTV zoom lens with a focal length of 1.4-3.1 mm. The overhead view camera was set at a height of 93 cm with a horizontal field of view of approximately 153 degree. In total, 40 images are used with different box positions at the same orientation. The size of the image is 1024 x 768 pixels. We move this box with a starting position under the camera until it is at the edge of the scene, and then record its height and width. This setup is shown in Figure 4.12. This is a scene from the computer laboratory of the University of Southampton, the United Kingdom. The camera is mounted on a tripod in the middle of the room. The computer desks around the camera setup can also be seen. The centre of the image is shown in Figure 4.12 as a small white circle plotted on the box beneath the overhead camera.

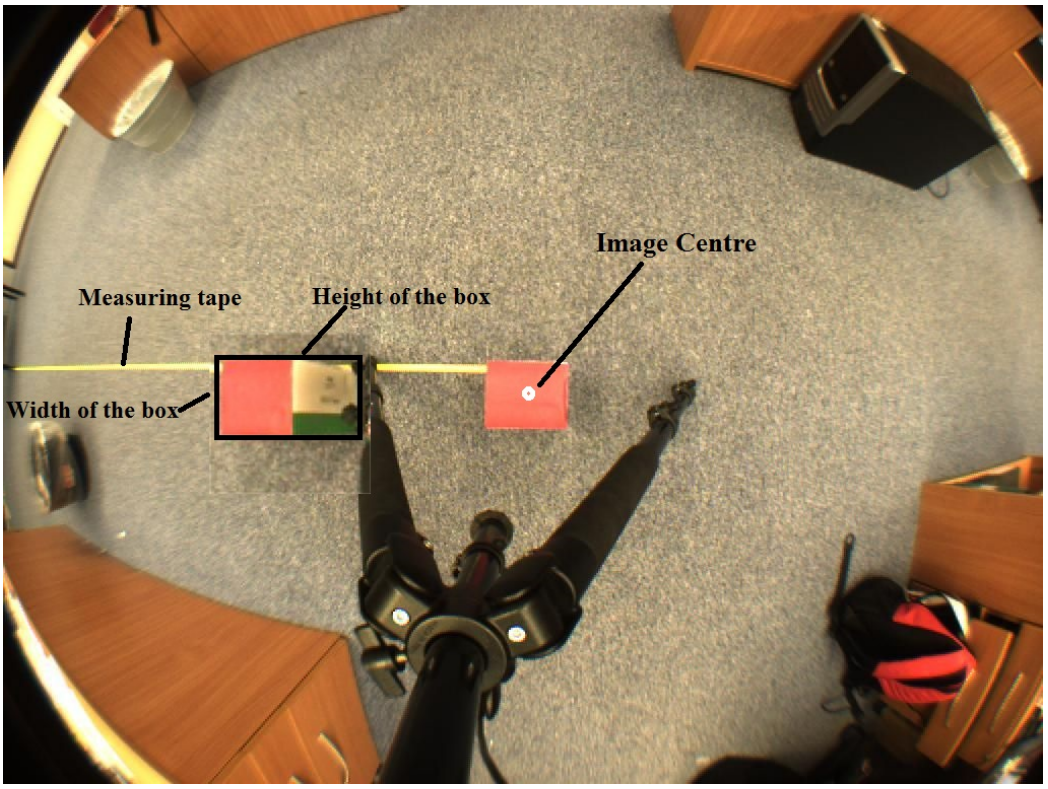


FIGURE 4.12: Camera setup for measuring the size of the rectangular box at different positions using an overhead camera.

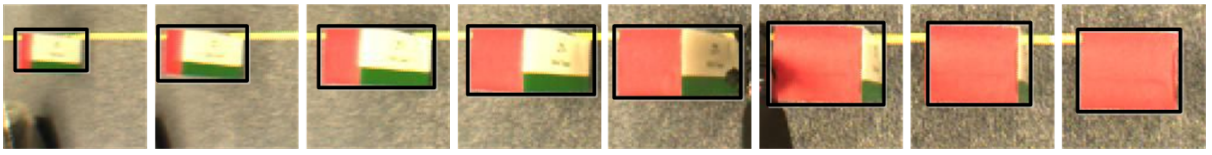


FIGURE 4.13: Effect of variation in width and height of the box while away from the centre of the camera. The bounding box can also be seen as a black rectangle around the box.

We use a measuring tape from the centre of the camera to the edge of the scene in a horizontal direction. This measuring tape can be seen as a yellow strip in Figure 4.12 and is used to record the measurement of the box distance from the centre of the camera. It can be seen that just below the camera the lower part of the box is obscured and its visibility only starts to emerge when we move it away gradually from the camera centre. The changes in the width and height of the box can also be seen while away from the centre. The height gradually increases while the width decreases gradually, until mid-scene. After that, due to radial distortion we could see an overall decrease in the box size. Starting from the position underneath the camera up to the edge of the secne, we measure the width and height of the box while moving it with a fixed interval of 12 cm. Here we consider the height of the box as a bounding box and it is treated in such a way as to cover the whole box within it. In Figure 4.13 we can see some sample images of the box. The bounding box is shown as a black rectangle. We mapped these readings from real units (centimetres) to image units (in pixels). The overall results of these readings

are plotted as a graph and shown in Figure 4.14. In this graph the height and width of the box in pixels is plotted along the y-axis. In this graph, the x-axis represents the radial distance in pixels starting from the centre to the edge of the image. In the graph after the third reading there is a big gap of missing data. This interval is because of the interaction of the box with the tripod at these positions. The overall conclusion is that the height of the box increases gradually upto the middle of the radial distance from the image centre and after that it reduces gradually due to the radial distortion of the image.

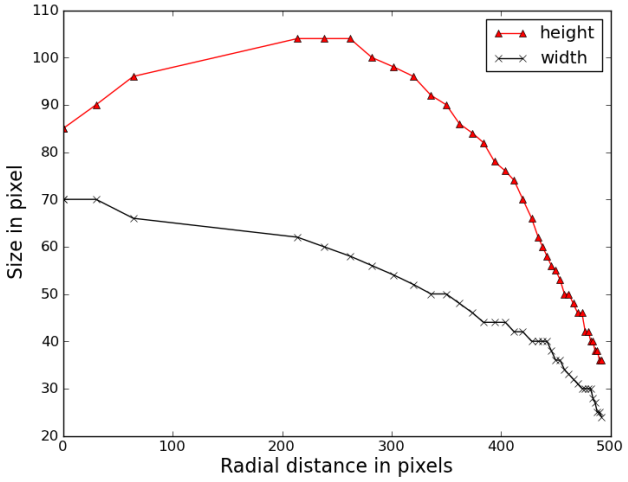


FIGURE 4.14: The width and height of the box as a function of its radial distance from the centre of the image.

4.4.2 Observing a person's size from an overhead camera

In this experiment we use the SCOVIS dataset for the purpose of measuring the size of the person. We first took a few readings about the width and height of a person in different consecutive images when he moves towards a particular direction from the centre and out to the edge of the scene.. For the demonstration purpose we plot some of these positions with a person in a synthetic image as shown in Figure 4.15. Even with visual analysis it is very clear that the size of the person increases gradually while away from the centre. The reason for this is that, in the middle of the image just below the overhead camera, the lower body of the person is obscured by the upper body and it only starts to become visible as he moves away from the centre. For example by looking at the person in Figure 4.15 the torso and legs are clearly visible while the person is near the edge; these were previously obscured in the middle of the image.

We further explain these visual effects using some quantitative measurements. To do this we use about 900 sample images where the person is at different radial positions from the centre. We manually record the width and height of the person starting from the radial distance of 60 pixels and a step size of 10 pixels. The height of the person was

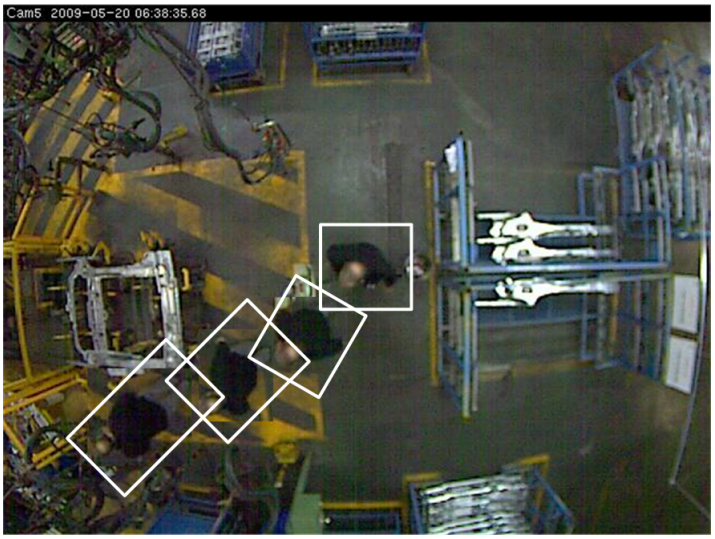


FIGURE 4.15: Synthetic image: People at different positions at a particular orientation. The size of the person increases while away from the centre.

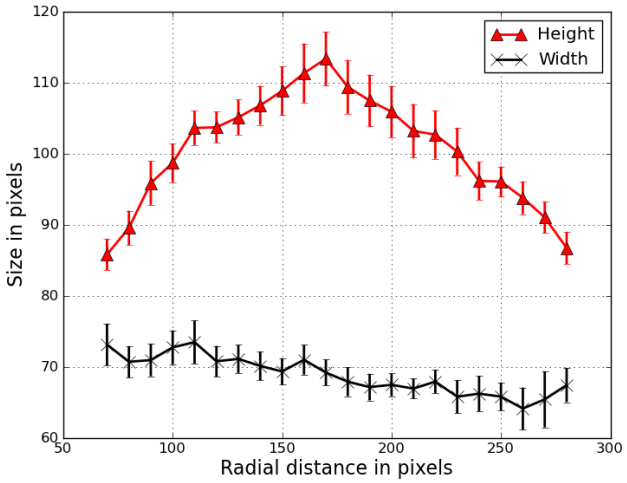


FIGURE 4.16: The width and height of the person as a function of his radial distance from the centre of the image.

measured from the top of the head to the feet. The mean height and width of the person at each of these radial positions are measured with a standard deviation from the mean. These results are depicted in Figure 4.16. It can be seen in this graph that although the width of the person is not varying so much, the pattern of width in this graph is not consistent. The reason is that in contrast to a static object like a box the person is a deformable object and could have a great deal of variation in any pose or activity he performs. For example, the width of the person at the same position can vary when the person is standing or walking normally, and when a person extents his hands to carry or handle a metal rod, for example. The height of the person in this graph increases upto a radial distance of 180 pixels. The reason is the same as was explained above using Figure 4.15. After this, due to the radial distortion of the image and the greater

distance from the camera, the size of the person reduces gradually.

In this section we explain visually and quantitatively how the size of an object changes. The conclusion is that in the overhead view beneath the camera the upper body occludes the lower, and while away from the camera it starts to become visible and elongated in size. As the person moves further away due to an increasing distance and distortion effects occur, the size reduces gradually.

4.5 Results of the Person Classification

In this section we first describe the bounding box size selection procedure and then compare the person classification results of our proposed algorithm with sHOG. We use already tuned parameters of the HOG algorithm discussed in chapter 3.

FIGURE 4.17: Annular bands of 20 pixels. In total 14 sub-regions of the whole image.

Using visual inspection, the image was split into 14 sub-regions. Starting from the image centre each region is divided on the basis of radial distance of about 20 pixels. These radial bands per 20 pixels are shown in Figure 4.17. An average image is then formed for each of these 20 pixels circular bands. Figure 4.18 shows the average image per band. The number of people per band is shown in parenthesis. For example, in radial region 151-170 there are a total number of 2112 people. We measure the height and width of each of these average images. These measurements are shown in Table 4.1. The overlapping regions with the same-sized measurement were merged. For example in Table 4.1 the region with a radial distance of 31 to 90 pixels has the same width and height so these regions were merged. After merging, seven regions were chosen; these are shown in Figure 4.19. The average person images of these seven annular bands are shown in Figure 4.20.

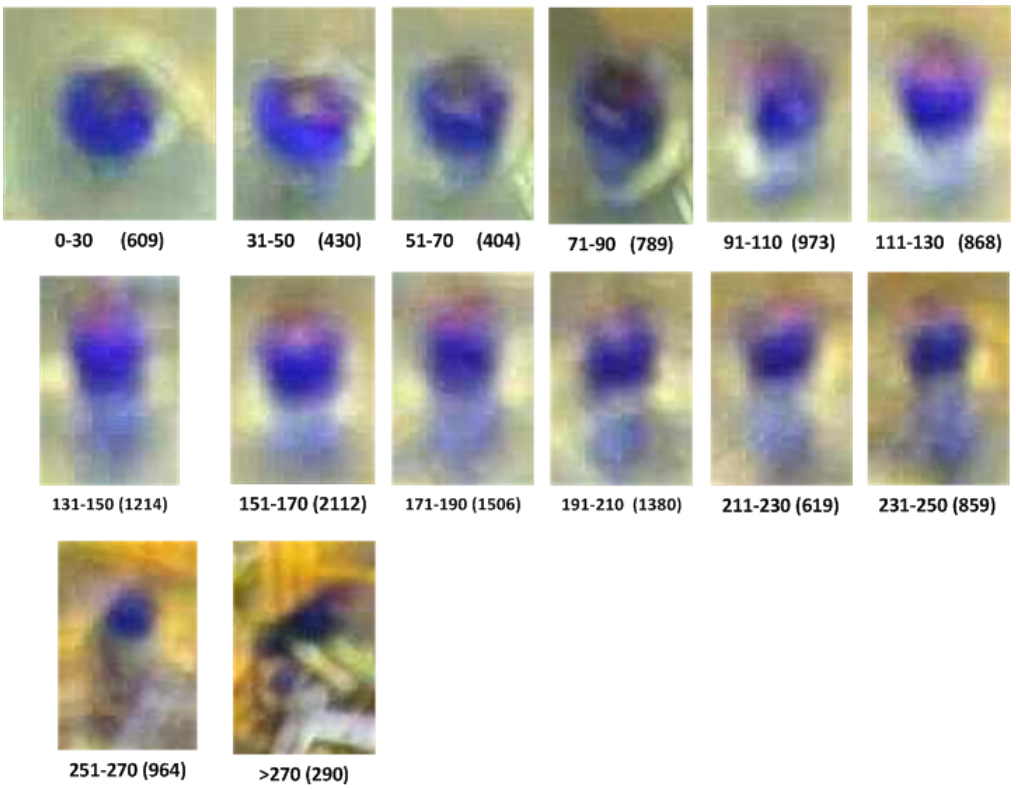


FIGURE 4.18: Average image per 20 pixels radial band.

TABLE 4.1: Average person size per 20 pixels radial distance.

Radial distance	Width	Height
0-30	64	64
31-50	64	72
51-70	64	72
71-90	64	72
91-110	64	82
111-130	64	82
131-150	64	104
151-170	64	96
171-190	64	96
191-210	64	96
211-230	64	96
231-250	72	86
251-270	72	86
270 onward	72	82

We use the sizes of the chosen bounding boxes as mentioned in Table 4.1 and perform 2-fold cross-validation testing for each of these bands by varying the width and height of the bounding box. These results are shown in Appendix-A. The final selection of bounding box size for each region is made on the basis of best classification result. These results are shown in Table 4.2. For the sHOG algorithm, we used a square box whose size was 96x96 pixels. This size was also determined by maximising the performance.

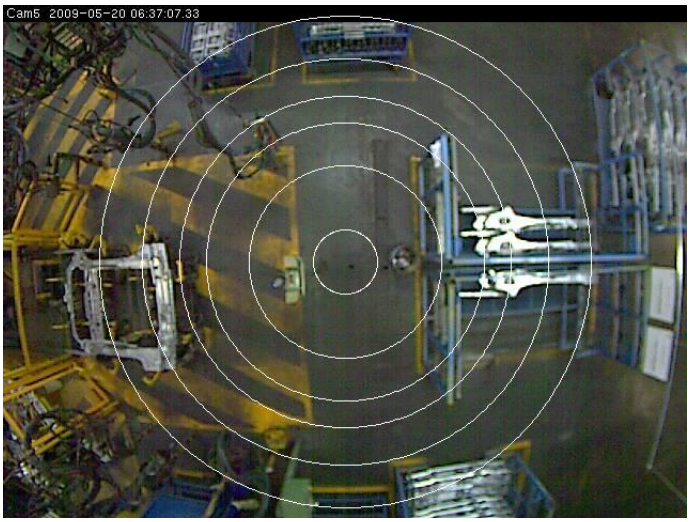
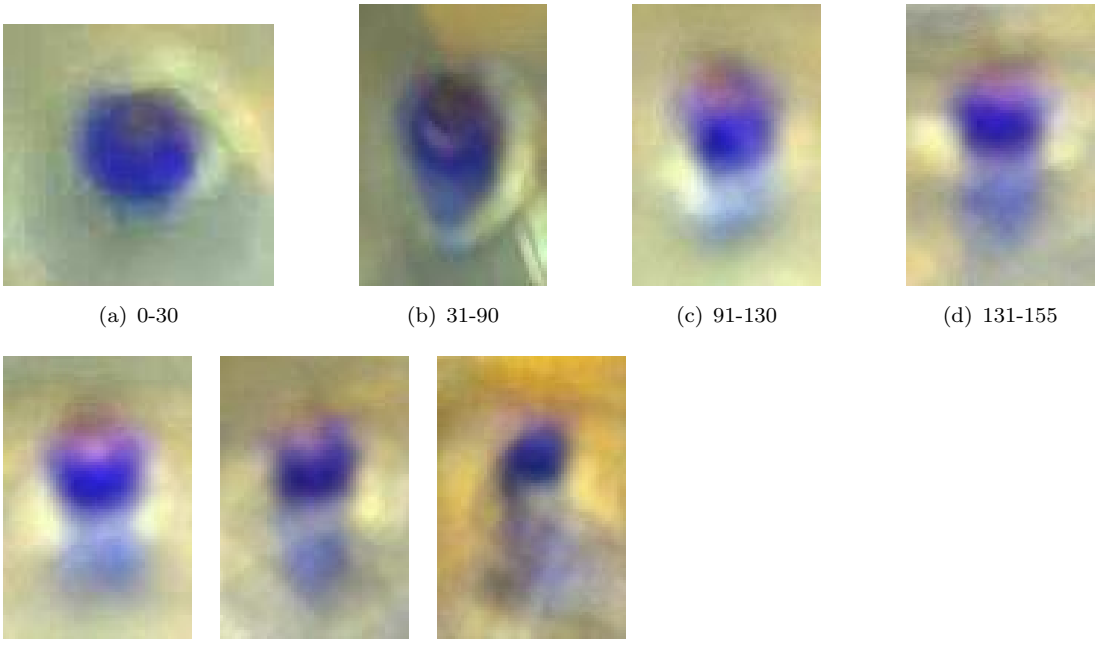


FIGURE 4.19: The chosen seven circular regions.



(a) 0-30 (b) 31-90 (c) 91-130 (d) 131-155

(e) 155-190 (f) 190-230 (g) 230 onward

FIGURE 4.20: Average person per circular region. Each of this image is represents the average person per radial distance.

We achieved average TPRs of 97% and 94% for rHOG and sHOG respectively, together with average FPRs of 2% and 6% using 2-fold cross-validation. No significant variation was found when the test and training data were swapped. These results show that our new algorithm improves TPR by three percentage points and reduces FPR by four percentage points.

TABLE 4.2: Person Classification results of each circular band.

Radius	Images	Box size	TPR	FPR
0-30	609	80 x 80	98 %	0 %
31-90	1623	64 x 104	97 %	0.2 %
91-130	1841	64 x 120	98.4 %	0.9 %
131-155	1546	64 x 128	97 %	1 %
155-190	3286	64 x 96	95 %	3 %
190-230	1999	64 x 96	94.5 %	3 %
230-	2106	72 x 96	97.5 %	4 %

4.6 Effects of Occlusion

In the previous section we discussed the classification results per circular region of image. The results are shown in Table 4.2. From this table it can be seen that the FPR increases gradually from 0% to 4% as we move from the inner-most to the outer most circular region. In this section we address the reason for high FPR in results.

As discussed in the data section of Chapter 3, more than 50% of our dataset contains occluded people. Table 4.3 shows the number of people per chosen circular band. The occlusion percentage at each band is also shown. From this table and the results shown in Table 4.2, it can be seen that the FPR appears to increase with the increase of occlusion level. For example, in the inner band there are not occluded people and a FPR of 0% but at the outer-most band 80% of the people are occluded in some way and the FPR is 4%.

TABLE 4.3: Occluded people samples per band in the range 0%-75% occlusion.

Radius in pixels	0%	25%	50%	75%	Total Occlusion %age
0-30	609				0
31-90	1473	150			9.24
91-130	1333	340	168		27.59
131-155	811	488	240	7	47.67
155-190	1012	1697	537	40	69.20
190-230	488	850	601	60	75.58
230	169	873	911	153	80.24

The performance of the algorithm at different circular regions of the image are shown in Table 4.4. The true positive rate decreases gradually as the occlusion level increases. It can be seen that using a test set with no occluded samples a TPR of 98% can be achieved while a test set contains only 75% occluded samples results a TPR of 94%. It is likely that the low drop in performance with occlusion is due to the SVM having been trained with occluded data.

TABLE 4.4: Performance per band: This table shows the true positive rate per band using people samples in the range 0%-75% occlusion.

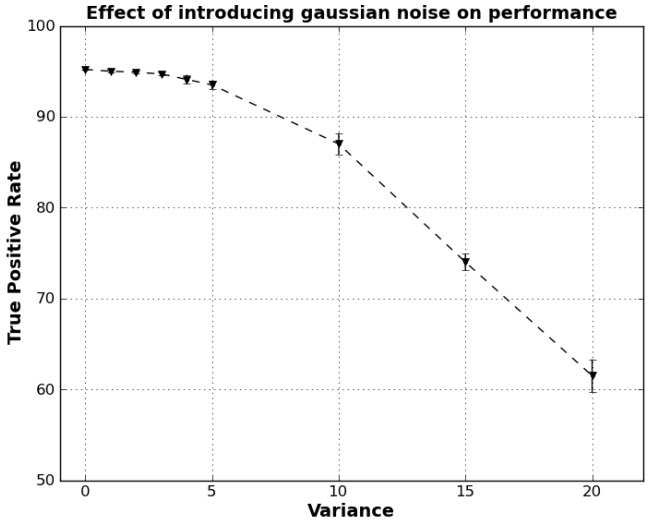
Radius in pixels	0%	25%	50%	75%
0-30	98			
31-90	97	97		
91-130	99	98	97	
131-155	98	97	97	95
155-190	98	95	93	92
190-230	96	95	94	93
230 onward	99	98	97	96
All	98	96.6	95.6	94

4.7 Effect of Noise on Performance

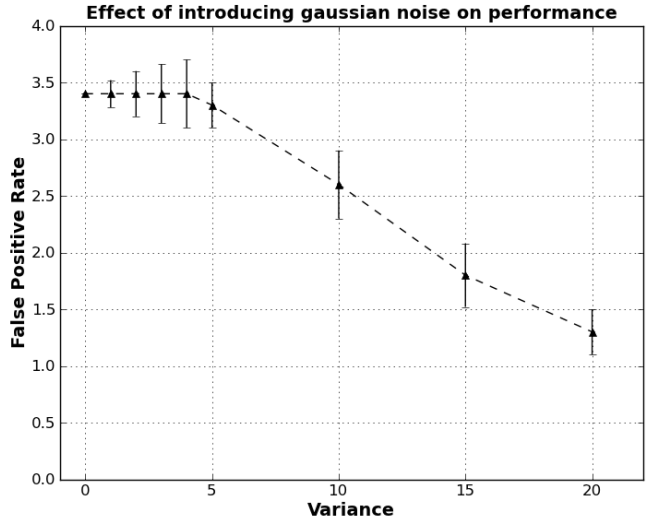
In this section we discuss the effect of added noise at the performance of the system. This experiment is performed using all the samples from band-4 (155-190 pixels radius). This region was chosen because this circular band contains the larger number of people samples. We randomly split the data into equal number of training and test samples. A previously trained model was used and Gaussian noise was introduced into the test images. This had zero mean and σ was varied from 0 to 20. This process is repeated four times and the results are shown in Figure 4.21. TPR against different level of Gaussian noise is shown in Figure 4.21(a). Starting from no noise at σ=0 to 5 there is only a small decrease in the TPR. The performance then decreases steadily with increasing of noise until it reaches approximately 60% at σ=20. The results of FPR against different level of Gaussian noise are shown in Figure 4.21(b). The noticeable thing is that with the increase of noise in the images false positive rate also decreases. This can be understood if one considers a region of a picture which is falsely classified as a person. This misclassification is due in part to the structure in the image. Adding noise should make the region less like a person, reducing FPR, which is the effect observed.

4.8 False Classified Positions per Image

To give the classifier more information we partitioned the training and test data into 5:1 and for the purpose of comparison we use exactly the same data as were used in the previous chapter for sHOG. The person classification result using rHOG was a TPR of 0.98 with FPR of 0.02 while the sHOG person classification result was TPR of 0.96 and FPR of 0.07. This is an overall improvement of seven percentage points in performance using the rHOG algorithm compared to the performance of sHOG algorithm. To evaluate the performances of both algorithms by scanning the whole image using pre-trained classifiers We use 200 images with people and the same number of person-free images. When person-free images were scanned using the classifier trained with



(a) Gaussian noise Vs. True Positive Rate.



(b) Gaussian noise Vs.False Positive Rate.

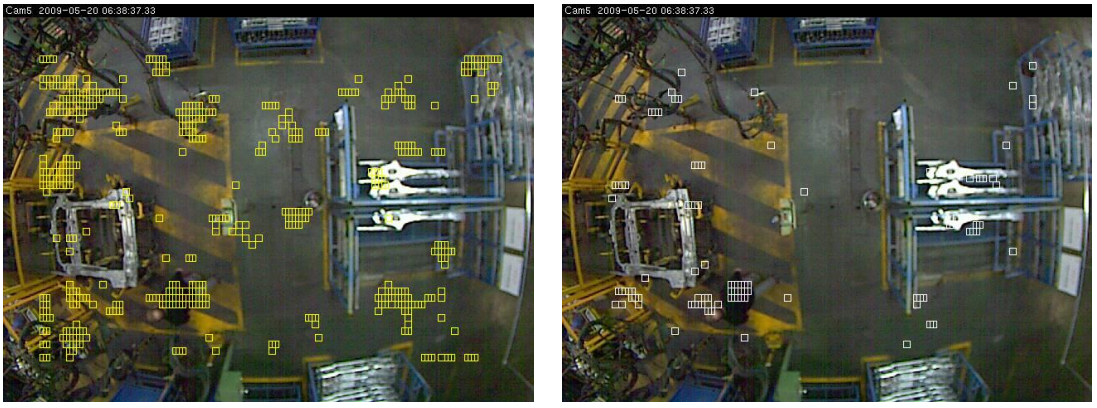
FIGURE 4.21: Effect of adding gaussian noise in test images. The horizontal points in these graphs show the variation of σ values in a range 0 to 20. The error bars represent the variations in the result using test sequence when adding gaussian in four different times.

the rHOG algorithm, on average we get 96 false detections per image; this means that an average of 96 positions of image while scanning gave instances of person. A sample image is shown in Figure 5.5(a) where the small boxes are the positions of the image that gave false classification. The detection results using a sample image with a person in it, can be seen in Figure 4.23. If we compare these average false classification results with sHOG when using the same training and test data, we get a high number of falsely detected positions; i.e, 453 with a classifier having TPR of 0.96 and FPR of 0.07. This means that by using the rHOG algorithm, we get more than four times fewer numbers

of spurious detections per image. Another factor worthy of note is that compared to the sHOG image scanning results where we got compact clusters here using rHOG, the false detections are randomly distributed and not compact. This effect can be seen in Figure 4.23.

(a) Using sHOG (b) Using rHOG

FIGURE 4.22: Falsely detected positions (shown in small box) using both algorithms while scanning person-free image.



(a) Using sHOG (b) Using rHOG

FIGURE 4.23: Falsely detected positions (shown in small box) using both algorithms while scanning the image with one person in it.

4.9 The Effect of Reducing Number of Classifiers

In the previous sections we discuss dividing the image into different regions or circular bands with respect to radial distance from the centre of the image. We use a separate classifier for each of these regions and the final result per image is produced by concatenating the classification results of all these regions. In this section we discuss our attempt to reduce the number of classifiers per image by merging different adjacent circular regions. The main conclusion is that merging these circular regions degrades the overall performance of the detection system.

We first use the information about bounding box size as demonstrated in the graph in Figure 4.16. To decide bounding box height, we use the fitting of two straight lines as shown in Figure 4.24.

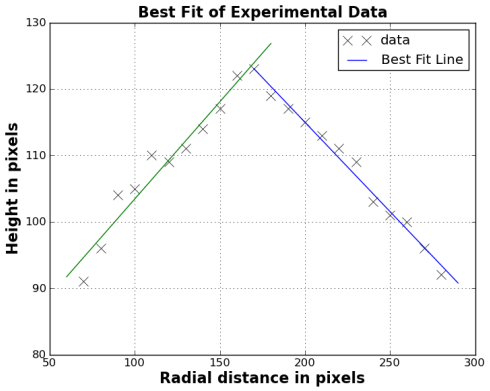


FIGURE 4.24: Curve fitting to estimate height of the bounding box.

These two straight lines are represented by Equations 4.1 and 4.2. The radial distances in these equations are represented by r. These two equations correspond to the height of the bounding box within radial distance 60-170, and 170 onward respectively.

$$H1 = 0.3245r + 77.42 \tag{4.1}$$

$$H2 = -0.3127r + 186.3 \tag{4.2}$$

The straight green line presents height when the person moves from the centre to the outer bounds of the image and his height in the image gradually increases. After the second half of the radial distance, due to radial distortion effect, the height reduces gradually until the end of the scene. This effect is shown using a blue line. The width of the person is not varying as much therefore we use a constant width of size 72 pixels. Here our intention is to reduce the number of classifiers by merging the outer circular regions of the image. Through-out the following experiments the inner or central region (0-60 pixels) is treated as a special case and a separate classifier is used in this region. We devise the following experiments using this curve-fitting information about measuring the bounding box size at each position of the image.

4.9.1 Merging all outer regions

In the first set of experiments we use one single trained classifier for the outer regions. The height of the bounding box at different scanning positions of the image is calculated using Equations 4.1 and 4.2 respectively. The width of the bounding box was 72 pixels.

The initial trained classifier was then tested using 200 person-free images. The result was on average 324 spurious detections per image. We build different trained classifiers by varying width of bounding box from 64 to 72 pixels. We also use different variations of bounding box height. The conclusion was that using one classifier degrades the performance of the person detection system.

4.9.2 Merging adjacent circular regions gradually

We perform a set of experiments by gradually merging seven different regions of image. We analysed that the spurious detections per image increase gradually as the number of classifiers is reduced. We concluded that the best performance was when we use our default split regions discussed in Sections 1.5 and 1.6 respectively. We assume that the following could be the reasons for this behaviour:

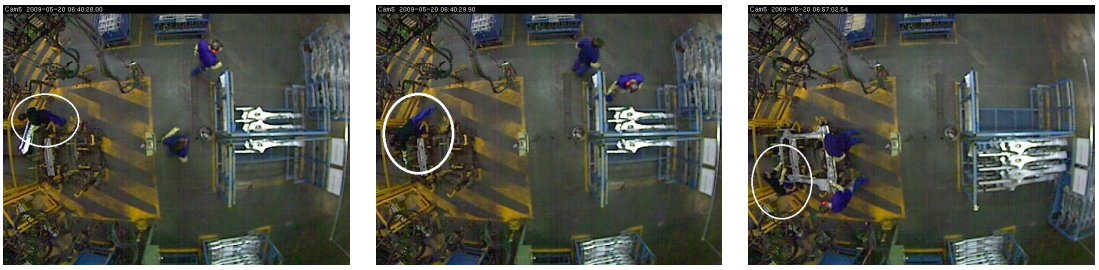
In the images used in our overhead view data, the people are performing different activities. Most of the time the people are carrying steel rods and handling and moving these rods onto and from the racks. These racks are at the outer region of the image. When undertaking these activities the people are occluded with rods, wires and heavy machinery and this occlusion level increases as the person moves aways from the central area which is comparatively simpler then the outer region. This evidence can be seen in Table 4.3. When moving from the central region to the outward edge in the image, the occlusion level for the person increases from 0% to more than 80%. In these occluded data the visibility of person reduces by as much as 75%. Some discussion about occlusion has already been undertaken in Sections 1.3 and 3.3, respectively, with sample examples.

(a) (b)

FIGURE 4.25: The person is at the same region and same position. The person is not only occluded with wires and rod here but also looks completely different.

Another reason could be that, as shown in Figure 4.20, the average person looks very different mostly in the outer regions. This is because the person in this environment performs different activities and their interaction with different machinery. Due to performing these activities it looks completely different within the same region or even at

the same place. Here we demonstrate this effect by some sample examples by highlighting the person within a circle just for display purposes. In Figure 4.25 the person in these two images is not only within the same radial distance but also at the same position. It can be seen that the person is not only obscured at the wire and rod but also looks very different when trying to strike steel rod with the rack while standing at the same position. Another example can be seen in Figure 4.26 where the person is at the same radial distance but looks different in all of these three images. The significant variation in poses even at the same radial distance can clearly be seen in these images.



(a) (b) (c)

FIGURE 4.26: The person is at the same radial distance with different poses. Due to performing different activities,the person looks significantly different even at the same position.

Finally we are using very noisy data with high compression rate in JPEG format.
Due to all the above-mentioned reasons, we can say that it becomes very hard for SVM to learn the majority of these variations and to distinguish the class of person with no-person.

4.10 Execution Time per Detection Window

The computer we use has an Intel Xeon 2.67 GHz dual core processor with 12 GB RAM and Windows-7 operating system. The processing time per detection window or bounding box using sHOG is 0.5 milli-seconds while our algorithm takes 0.7 milli-seconds to process a detection window. The reason why more processing time is taken using our algorithm is because of geometric transformation it performs before feature extraction.

Discussion

In this chapter we demonstrated how the size and orientation of the person varies in the overhead view. We explain our proposed algorithm which takes advantage of the radial symmetry property of the overhead camera. The geometric transformation used in our proposed algorithm always results in an upright orientation of the person in the image. We also explain the criteria for choosing the bounding box size. The conclusion is that

rHOG gives an overall improvement of seven percentage points in person classification results over the sHOG results. Similarly while scanning person-free images using both algorithms we found that our rHOG algorithm is not only four times better than sHOG but also gave randomly distributed false detections rather than compact cluster in the sHOG case.

Chapter 5

Finding people in images

In this chapter we discuss in detail the process of detecting people by scanning the whole image. For the purpose of comparison we use both the standard histogram of oriented algorithm and our proposed algorithm concurrently to illustrate different results associated with the person detection system. The main conclusion is that our proposed algorithm has a very strong discriminative power of detecting people in images.

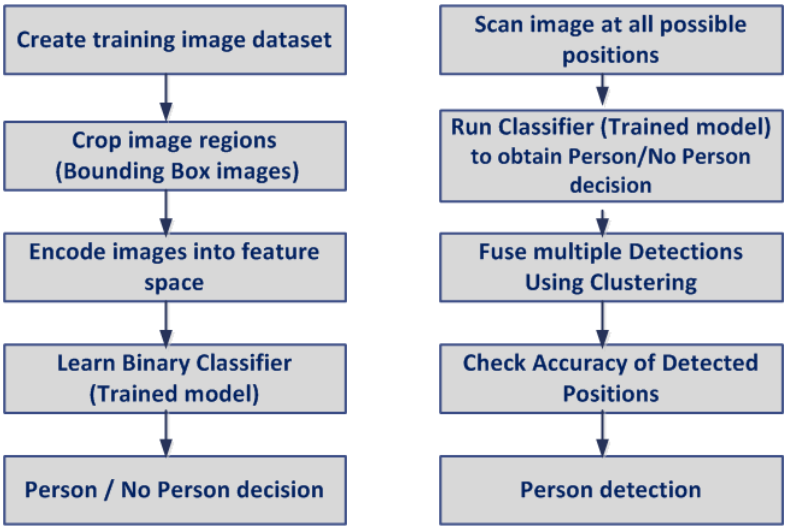
Starting from Section 5.1, we explain the overall architecture of a person detection system which involves a learning phase to build a binary classifier which is used in subsequent detection phase testing for the existence of people in images. Section 5.2 demonstrates the image-scanning process using sHOG with fixed and rHOG with variable orientation bounding boxes, respectively. In Section 5.3 we discuss the building process of a trained binary classifier for person/no-person classification which is a pre-requisite to the image detection system discussed in detail in Section 5.4. In Section 5.4 we also discuss how to fuse multiple detections per image using our proposed simple clustering algorithm. Section 5.5 presents the criteria for deciding the validity of a detected position using ground truth. The last section is a brief discussion of processing time per image.

5.1 Overall Architecture of Person Detection

The overall architecture of the person detection system is built around the person classification method. We divide this into two phases the Learning Phase and the Detection Phase. The steps associated with each phase are shown in Fig: 5.1.

In the Learning Phase we first create a training data set using images with and without people. We extract positive and negative detection windows from these training images. The positive detection window is the region of image containing a person and is labelled manually. The negative detection windows are chosen randomly from person-free images. We then extract rHOG features from these detection windows or bounding box images

and feed these features to a linear SVM to build a learning or training model. This learning process can be seen in Fig: 5.1(a). The result of the learning phase is a binary classifier that provides a decision for a person/no-person for a given test bounding box image or detection window.

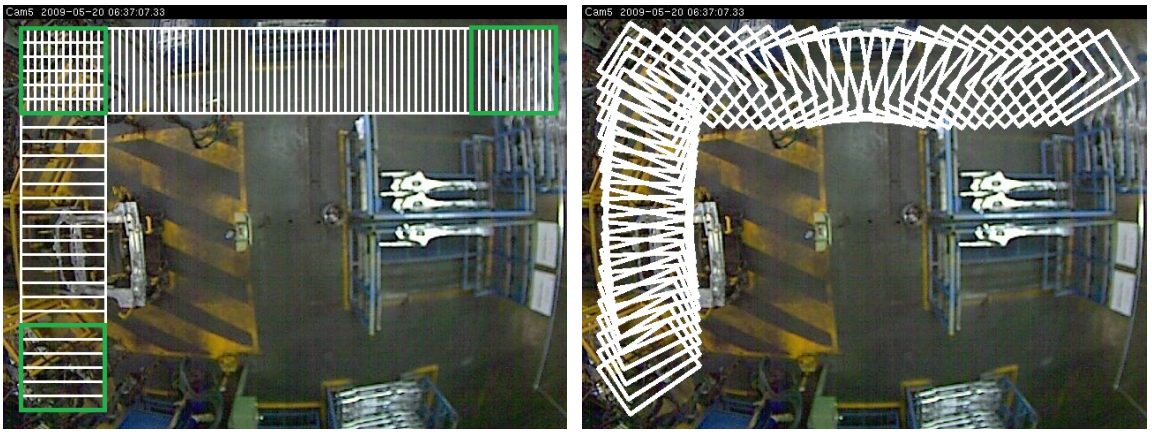


(a) Learning Phase. (b) Detection Phase.

FIGURE 5.1: Overall Person Detection Architecture: The learning phase is used to build a Binary Classifier for a Person/No Person decision. In the Detection Phase we test the given input image for the possible candidate(s) of a Person.

In the Detection Phase we use a pre-trained classifier of the Learning Phase to scan around all the possible positions (centroids of the detection window) of the given test image for a person/no-person decision. These classifications are then combined based on some fusing or clustering technique to produce compact clusters of detections. The accuracy of these detections is tested using ground truth information and a threshold is then decided for the final decision about the detection of a person. This threshold can be an overlapping region or simply a distance between ground truth and a detected position. These steps for detecting a person in the given test image are shown in Fig: 5.1(b).

In this section we briefly discuss the overall framework of Person Detection in the image. There are two main modules: a Learning Phase and a Detection Phase. The main role of the Learning phase is to build a trained model or Binary Classifier for a Person/No Person decision by providing a detection window or bounding box image. In the Detection Phase we exhaustively search the whole image at all possible positions using a pre-trained Binary Classifier for the presence of a person.



(a) sHOG with fixed orientation bounding box. For clear visibility the size of the bounding box is shown as a green square.

(b) rHOG with variable orientation bounding box.

FIGURE 5.2: Scanning positions (centroids of the detection windows) in image: This figure shows the image scanning process using two different algorithms. A sample of bounding box overlapping in horizontal and vertical directions can also be seen.

5.2 The Image scanning Process

Person classification is a pre-requisite to detecting people in images. Therefore to detect people in images we use a pre-trained binary classifier with an overlapping detection window to test each possible location of the image for a person/no-person classification. For a 640×480 pixels sized image, with 4 pixels horizontal and 8 pixels vertical step size, there will be a total of about 6000 possible locations that need to be classified as a person/no-person. This classification result per image is then used for onward processing.

The image-scanning process of sHOG and rHOG as shown in Fig: 5.2 is the same i.e from left to right in the horizontal direction and top to bottom in the vertical direction, in linear order. However the orientations of the bounding boxes are radically different. Here we just show one sample of row and column scanning using an overlapping bounding box or detection window, shown in white colour. For visual clarity, we use a green square as shown in Fig: 5.2 to emphasise the size of bounding box used in the sHOG algorithm. In the case of the sHOG algorithm we can see a fixed orientation of bounding boxes at all positions of the image as shown in Fig: 5.2(b) but in the case of rHOG, the orientation of the box varies depending upon its position and radial distance from the centre of the image. For example, near the corners of the image, these boxes looks diagonal and change orientation as they move from left to right or top to bottom of the image. At the middle row and column it seems vertical and horizontal respectively. The radial symmetry can also be observed. For example the boxes at the two ends of the row or column are in different orientations.

The conclusion of this section is that in the image-scanning process of detecting people a fixed orientation bounding box is used in sHOG algorithm but a variable orientation bounding box is used in the case of the rHOG algorithm.

5.3 Creating a Binary Classifier for Learning

To build a binary classifier, the data set was randomly split into 3050 test images and 9277 training images. From these images 10803 and 2213 examples of people were used for training and test respectively, while identical numbers of negative samples were randomly chosen from person free images. The ratios of training and test samples in all annular bands were similar. We built our initial classifier using rHOG and sHOG algorithms. The classification result was a TPR of 0.98 and FPR of 0.02 by the initial classifier using rHOG algorithm. For the sHOG algorithm, with exactly the same data, the result was a TPR of 0.96 and FPR of 0.07.

These initial classifiers are then used to scan 200 randomly chosen person free-images from the test set and as a result we got on average 453 and 96 spurious detections per image using sHOG and rHOG algorithms respectively.

FIGURE 5.3: Average False positions per person free image. The additional training samples at horizontal axis are shown in percentage.

To reduce spurious detections or false positives, we analysed the training images to determine common failure modes. We selected a number of these false positive examples to use as extra negative samples. We retrained the classifier, adding these extra negative samples in batches. Thus the final training used a total of 12203 negative samples by adding 1400 background samples. In this way we built eight different binary classifiers. Each of these classifiers is used to test false positives using person-free images and true

positive using images with people. The same process was used for both algorithms. We tested 2000 person-free images and 1050 images with people using the sHOG and rHOG algorithms. The effect of extra background training is shown in Fig: 5.3. The horizontal axis shows that the additional background samples in training start from 0% (No extra background training using initial classifier) to 13 % additional person-free samples in training. It can be seen that the false positive rate reduces gradually with the increase of additional background samples using both algorithms. From this graph we can infer that, providing additional training reduces FPR from 0.07 to 0.04 using sHOG and 0.02 to 0.006 using rHOG respectively. The True Positive Rate was reduced by 2% and 1% as shown in Fig: 5.4 using sHOG and rHOG respectively.

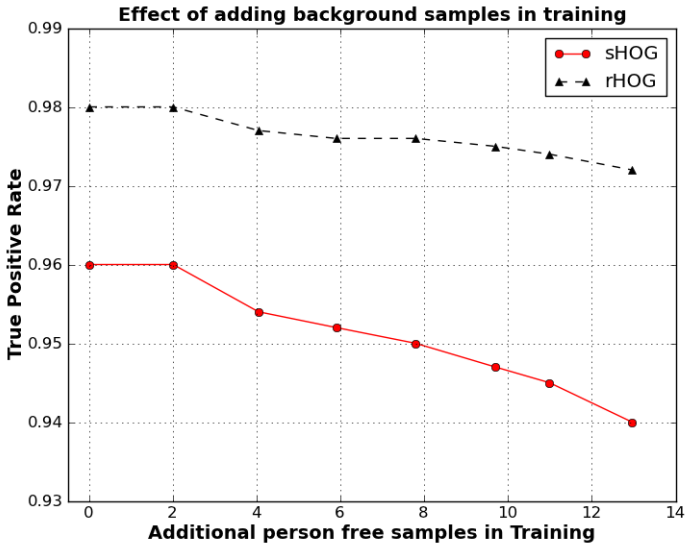


FIGURE 5.4: Effect of additional background training on TPR. The additional training samples at horizontal axis are shown in percentage.

To reduce FPR for sHOG further training is required with many more hard negative examples, but this also reduces TPR unacceptably. We select the final classifiers which have a low false positive rate for both algorithms.

In this section we discuss how to build a binary classifier used for person classification and person detection in images. We also explain that extra background training is used to reduce false positive rate. We also compare different detection window based classification results using the sHOG and rHOG algorithms. The Person Classification is a pre-requisite to Person Detection in the whole image. Therefore, in the next section, we use the same data (2000 person-free images and 1050 images with people) and extend this discussion to detecting people in images.

5.4 Fusing Multiple Detections

In the image-scanning process, we use overlapping detection windows to search for the existence of a person in all possible positions of an image using a pre-trained binary classifier. We start with our initial pre-trained binary classifier using the rHOG algorithm as discussed in the previous section and test a few sample test images with and without people.

(a) Person free image: Randomly distributed spurious detections

(b) Image with a Person: Multiple compact detections around the person in image

FIGURE 5.5: Result of image scanning using a pre-trained binary classifier using rHOG algorithm.

In the case when there is no person in the image, randomly distributed spurious or false positive detections are observed. When a person is present in the image, a concentration of positive detections is found around the position of the person along with some randomly spurious detections. This situation can be seen in Fig: 5.5. Here, a post-processing technique is required, which can combine these compact detections for the decision about the possible candidate of a person.

[28] proposed a general solution based on kernel density estimation. It has three main steps. The first step maps the SVM scores to positive values or detections. Due to window overlapping in the image-scanning process at different scales there can be many detections associated to a single person. In the second step each of these detections is mapped to a 3D-space (x,y,scale). In the last step mean shift mode detection algorithm [24] is applied. This process is shown in Fig: 5.6. This non-maximum suppression has some parameters which are a) scale ratio, b) window-stride, and c) smoothing parameters for x and y. It is reported by the author that these parameters can have a very significant impact on the overall performance and therefore proper evaluation is necessary. We refer readers to [28] for an in-depth explanation of this.

As the data in our study differ in nature, we do not need scaling because, in our case, scales are fixed. We here propose a simple one-pass clustering algorithm which makes

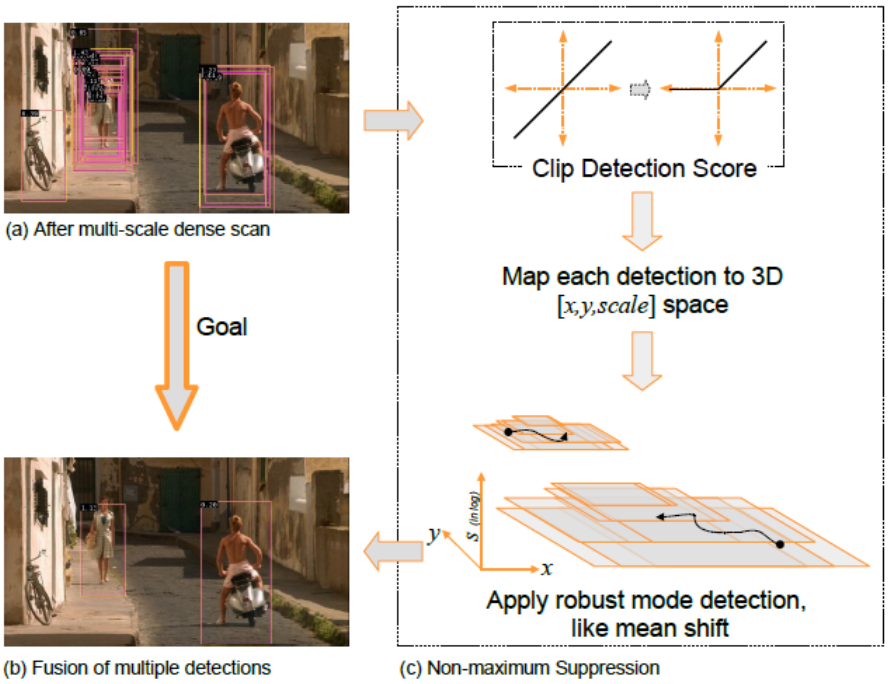


FIGURE 5.6: Fusing multiple detection. ([28])

clusters of detected positions based on the intra-distance threshold. The clustering algorithm is discussed briefly as following:-

1. The first detected position is taken as a cluster and all the remaining positions are compared with this using euclidean distance. The euclidean distance of a detection position from the centre of a cluster is denoted by (ED). A cluster radius threshold D is then decided, which is a fixed radial distance of a cluster centroid.

2. If $ED < D$, the point is considered as the member of this cluster. The centroid of this cluster is then updated using mean positions of cluster centre and given point. Thus the growth of the cluster is controlled only by this D.

3. If $ED > D$, the given point will be considered as another cluster and steps 3-5 will be repeated.

4. At the end of this process we get a list of clusters. Each of these clusters has its mean position and its size (number of cluster members).

5. A cluster size threshold is then chosen. Any cluster above the cluster size threshold is then considered as a person and the mean position of the cluster is assumed to be the position of the person in the image.

For our clustering algorithm two parameters need to be selected. The first is the radial distance of the cluster centroid and the second is the maximum number of cluster members. To evaluate the clustering parameters we use 1050 and 2000 images with and

without people respectively. These images are tested by a pre-trained Binary Classifier with least false positive rate using sHOG and rHOG algorithms as discussed in the previous section. The results of clustering parameters are shown in Fig: 5.7.

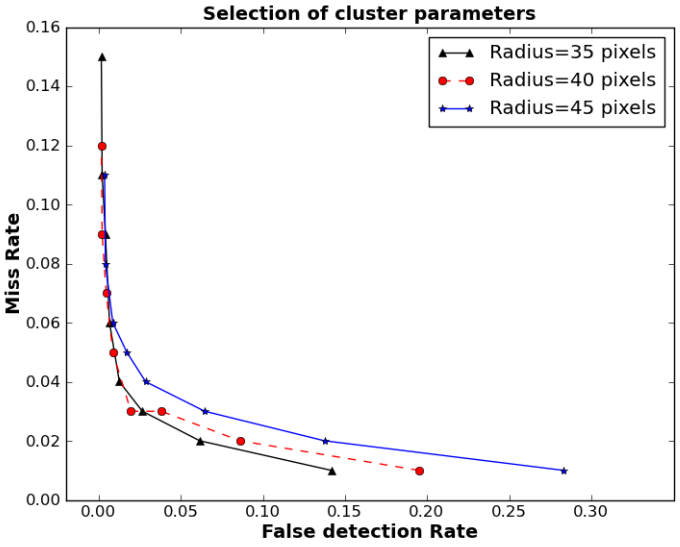


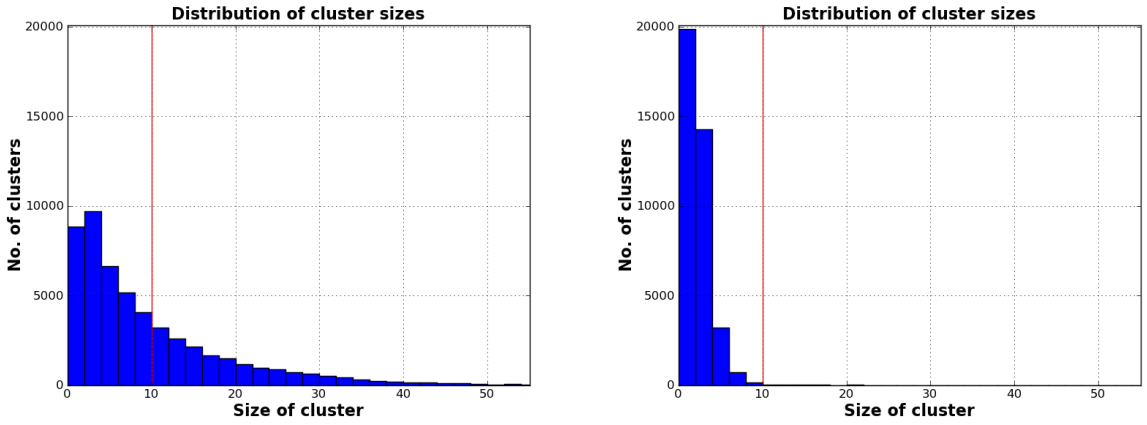
FIGURE 5.7: Selection of clustering parameters: Cluster members (7-14) are shown here from right to left.

We vary the radius of clusters from 35 to 45 pixels respectively. In the graph the size of the cluster (from 7-14 members) are shown from right to left. From this graph we found that a cluster joining distance of 40 pixels and 10 members was appropriate. This 40 pixels size is also consistent with the sizes of bounding boxes used.

In Fig: 5.8, the distribution of cluster sizes using 2000 person-free test images is shown using both algorithms. The horizontal axis represents the size of cluster i.e total number of cluster members and vertical axis shows total number of clusters associated with cluster size. The chosen cluster size threshold is shown by drawing a red line. It can be seen that there are very few falsely detected clusters above the chosen threshold using the rHOG algorithm, whereas in the case of the sHOG algorithm, many false clusters can be seen.

To see the effect of extra background training after using the clustering algorithm, we test 2000 person-free images using eight classifiers as discussed in previous section. In these person-free images the clusters with more than 10 members are considered as a False Cluster or False Detection. The results are shown in Fig: 5.9 using both sHOG and rHOG algorithms. According to these results we detected approximately 2 and 12 false clusters per image for rHOG and sHOG respectively.

An important observation, as shown in Fig: 5.9, is that the number of False Clusters



(a) Using sHOG (b) Using rHOG

FIGURE 5.8: Distributions of cluster sizes using 2000 person-free images.

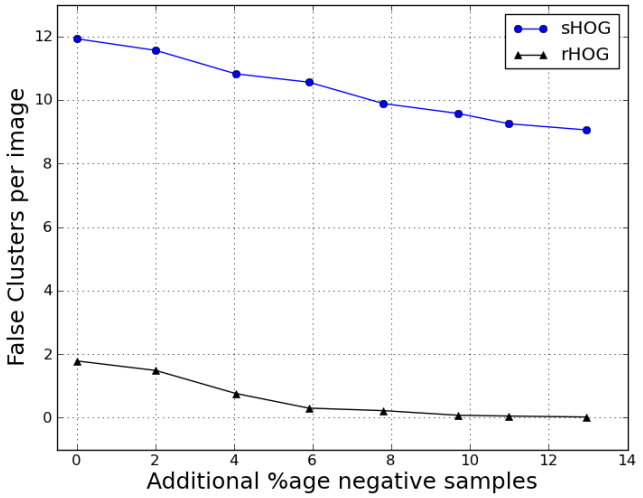
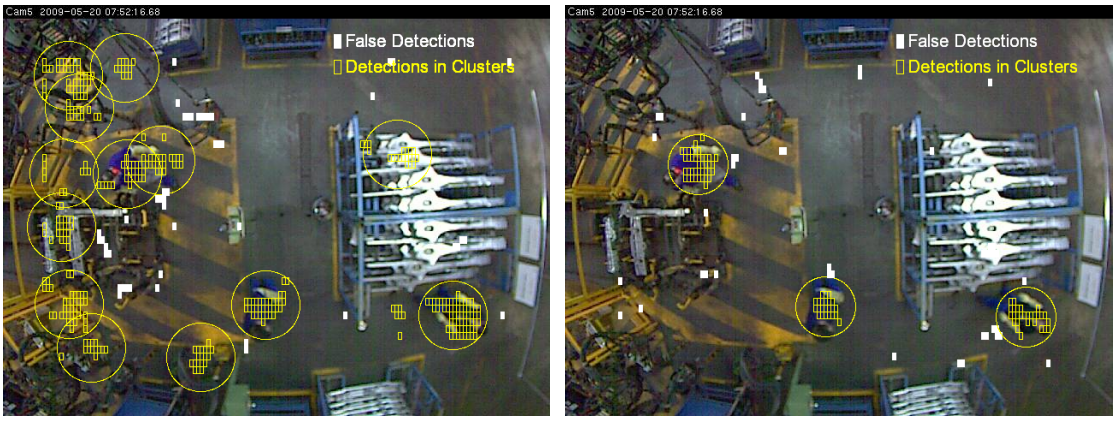


FIGURE 5.9: Effect of extra background training after clustering.

(falsely detected people) using the rHOG algorithm dropped to almost zero (1 false cluster in 50 images) while that for sHOG dropped to 9. This means that each image on average contain 9 spurious clusters regardless of the presence of a person or not. A sample example can be seen in Fig: 5.10(a) and Fig: 5.10(b) where the automatic detection of three people is made using sHOG and rHOG respectively. In these two examples we present the clusters as a yellow circle just for display purposes. The small yellow box within the circles corresponds to cluster members whereas the white small boxes are detections less than the cluster size threshold. Both of these algorithms have detected clusters around the position of the people in the images. Using visual analysis of the image we can say that the rHOG algorithm detects three clusters which are associated with people, but in case of the sHOG algorithm 9 additional clusters are shown which do not correspond to people.



(a) Detection result using sHOG (b) Detection result using rHOG

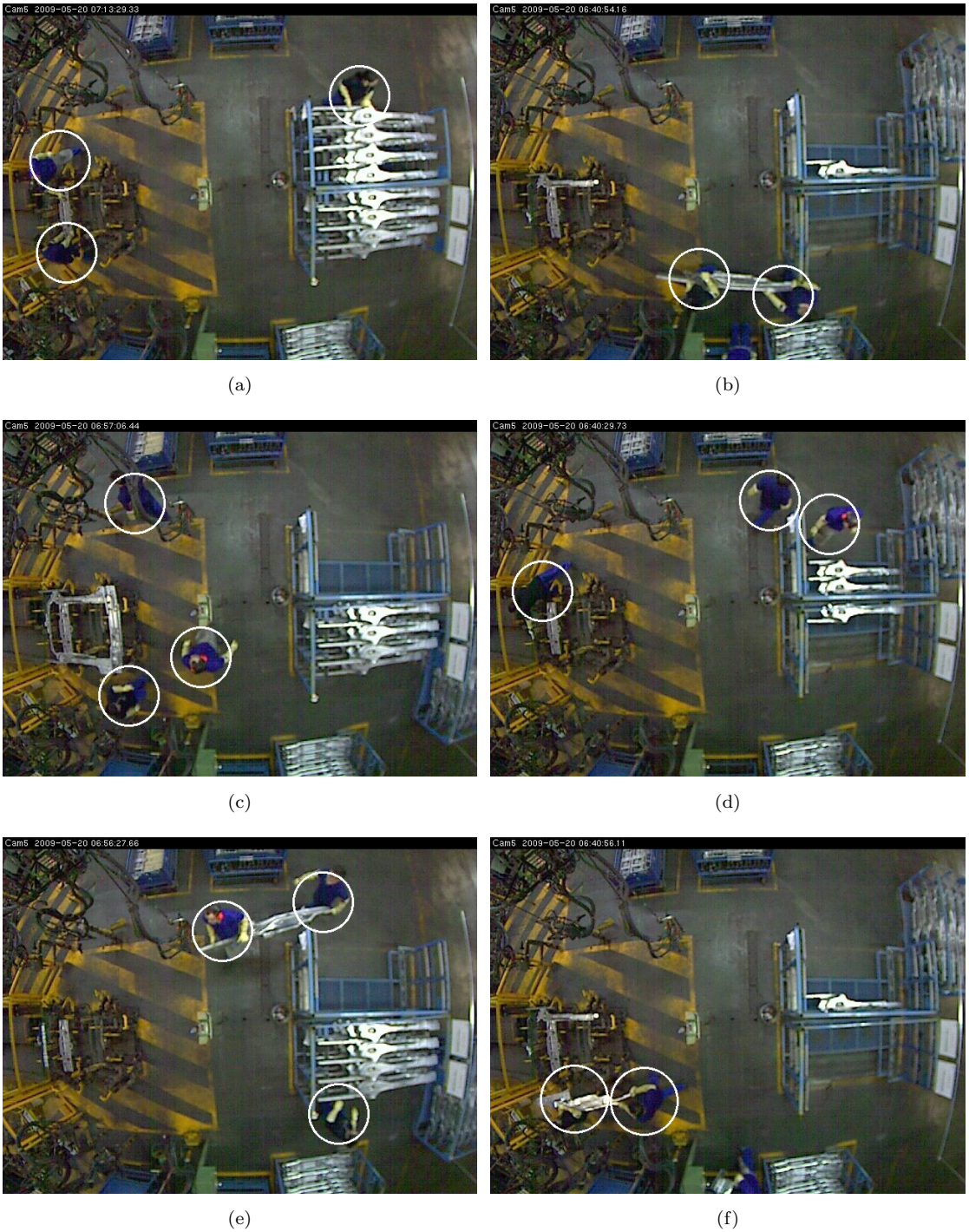
FIGURE 5.10: Automatic detection of people using both algorithms.

Some more sample examples of detecting people at various places of images using rHOG algorithm are shown in Fig: 5.11. In these sample images, people are detected without any false detection. Different variations in the poses of people can also be seen. Our algorithm can even detect occluded people. For example in Fig: 5.11(a), Fig: 5.11(e) and Fig: 5.11(f) the lower body of the person is occluded with assembling part of the car. Similarly in Fig: 5.11(c) an occluded person with wires is shown.

In this section we explain in detail the process of fusing multiple detections. We introduce our simple but effective clustering algorithm which is controlled by cluster size and cluster radius. The selection of cluster parameters has been explained using test images with people. We also demonstrated the effect of extra background training on false detections per image after applying clustering algorithms. The main conclusion is that our rHOG algorithm detects one false person in 50 images whereas the sHOG algorithm detects nine falsely detected people per image.

5.5 Validity of Detected Positions

Once the cluster parameters are decided, the next step is to test whether the detected cluster corresponds to the actual person or not. We use 1050 test images containing people. To demonstrate this process in an easier way we use those images which contain one person per image. For this analysis we assumed that the largest cluster corresponds to the person in the image. We then measure the distance between actual and measured position of the person in the image. Fig: 5.12 shows the disparity between actual and detected position using the two different algorithms. We chose a disparity of 36 pixels shown by the red line in Fig: 5.12. It can be seen that most detected clusters are within this disparity using rHOG, whereas a wider distribution of clusters can be seen using sHOG. Thus, for the rHOG algorithm 95% of all detections are centred within a radius of



(a) (b)

(c) (d)

(e) (f)

FIGURE 5.11: Automatic detection of people using the rHOG algorithm: The automatic detection of people in different positions with variable poses can be seen. The detection of occluded people can also be seen in (a),(b),(e) and (f).

36 pixels from the true position of the person. For the same radius, the sHOG algorithm contains only 59% detections.

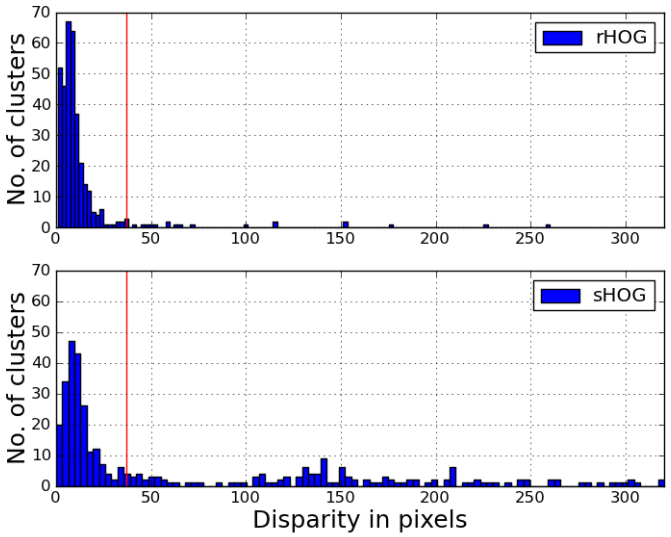
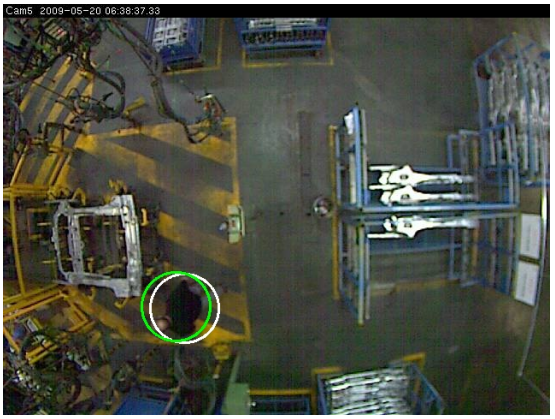


FIGURE 5.12: Disparity between actual and detected positions.

In Fig: 5.13 some examples are shown using the rHOG algorithm. In these examples ground truths are shown in the white circle, detected positions are shown in the green circle and false detections are shown in the red circle. In the first three examples (a)-(c), a disparity value represented by d is shown in pixels which varies from 10 to 35 pixels. In the last three examples (d)-(f), a disparity range 42 to 45 pixels are shown and their corresponding detected positions are highlighted as red circles. As these detections are beyond our chosen disparity threshold, they are therefore considered as false detections. The noticeable thing is that these false detections are not distributed randomly. We can see a co-relation of false detections to the person's position, because these occur close to the position of people.

5.6 Execution Time

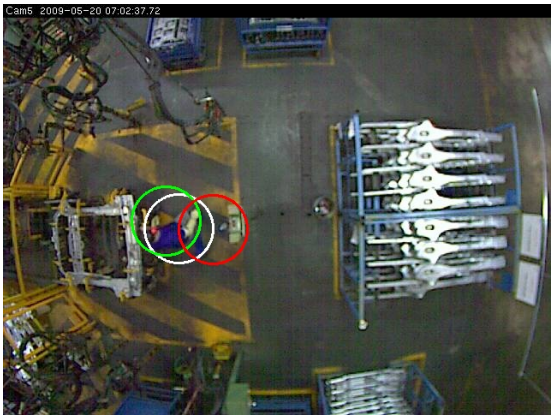
The computer we use has an Intel Xeon 2.67 GHz dual core processor with 12 GB RAM. The processing time per 640×480 pixels image using the sHOG algorithm is 2.9 seconds while our algorithm processes the same image in 4.25 seconds. Our algorithm is slower than sHOG because of using geometric transformation at each position of the image. The second reason is that our code is not optimised. Furthermore, we are not using any advantage of parallel processing or GPU computing, and we believe that doing this could reduce the processing time significantly.



(a) disparity=10 pixels

(b) disparity=20 pixels

(c) disparity=35 pixels



(d) disparity=42 pixels

(e) disparity=45 pixels

(f) disparity=45 pixels

FIGURE 5.13: Some examples of disparity measurement. Actual positions of people are shown in the white circle whereas the green and red circles correspond to detected positions and false detections respectively. The disparity measurements are shown from actual ground truths to detected positions in images (a-c) and false detections (d-f) respectively.

Discussion

In this chapter we discuss the overall frame work of a person detection system which consists of a learning phase to build a binary classifier for person classification. This binary classifier is a pre-requisite for the next phase where we detect people in images. We explain our simple but effective clustering algorithm. We have shown experimentally that our rHOG algorithm detects one false cluster in 50 images compared to the sHOG algorithm which detects 9 false clusters per image. We also explain the criteria of testing the validity of a detected positions. The results show that our algorithm detects people correctly in images 95% of the time whereas the sHOG algorithm detects people correctly in images only 59% of the time.

Chapter 6

People Tracking

An overhead view using a camera with a wide range lens reduces the overhead tracking problem to a more manageable level, while providing full visual access to the environment. An example of this application area has been recently used by Arbab-Zavar et al.[9] in the context of a work-flow analysis within an industrial environment to recognise different activities (standing, walking, carrying and handling) carried out by the people. They use a hierarchical approach based on an ad-hoc blob tracker. In this chapter we are motivated to show that our person detection algorithm can do better than that of Arbab-Zavar et al.[9]. Our main aim is to demonstrate the performance of our proposed person detector by incorporating it in our people-tracking-by-detection algorithm. We compare the results of our tracking algorithm with the five built-in OpenCV algorithms. The main conclusion of this chapter is that our simple people-tracking-by-detection algorithm outperforms the OpenCV algorithms. The rest of this chapter is organised as below.

Section 6.1 describes the detailed implementation of our proposed tracking algorithm. We explain briefly the OpenCV algorithms in Section 6.2. The experimental results from a comparison of our algorithm with the OpenCv algorithms are discussed in Section 6.3. In Section 6.4 we compare the execution time of our algorithm against the OpenCV algorithms.

6.1 Tracking Methodology

Each image is allocated a title, and the recorded time of each image was extracted as shown in Figure 6.1 using a simple digit reader software developed in python. The details of this tool are discussed in Appendix-B.

FIGURE 6.1: Image region containing time information.

1) Scan images using rHOG algorithm until it detects a person(s).
This results a single cluster or multiple clusters.

2) a. For each cluster, a cluster object is created with attributes.
cluster_object = (name, time, centroid of cluster, size, members)

b. Create a list and append all cluster objects into it.
cluster_object_list = [cluster_object1, cluster_object2,..... cluster_objectn]

3) a. For each cluster in the cluster_object_list , a tracker object is created with following attributes.
tr_object = (names[], positions[],times[], cluster_object[])

b. Create a tracker object list and append each tracker object into it.
tr_object_list = [tr_object1, tr_object2, ... tr_objectn]

4) For each object in the tr_object_list
Find velocity (V), its magnitude (|V|) and time difference Δt.

5) Estimate expected position of a person in the next frame as following:
exp_position = current position + (V * Δt)

6) Calculate the size of search grid in the next image.
Search region = 32 + ΔS| , 32 is the initial constant size for search reion.

7) Take a square search region from the exp_position and run rHOG algorithm for the detection of a person in this search region.
   a. If it detects a person then measure the distance between this and previously detection position.
      i. If distance < 40 pixels, then update the tracker object and repeat 4)-6)
      ii. Else, create a new tracker object, append it into tr_object_list and repeat 4)-6)
   b. Else
      i. Remove the current tr_object from the tr_object_list
      ii. Check tr_object_list, if it is not empty then repeat 4)-6)
      Otherwise Exit Tracking.

FIGURE 6.2: Basic steps of our tracking Algorithm.

6.1.1 Main Tracking Module

The basic steps of the tracking process are shown in Figure 6.2 and discussed below.

1. The first step is to start image scanning from the first image of the test sequence using the rHOG algorithm to check the existence of a person in the image. This process continue until a person has been detected. In the case of a detected person, the rHOG algorithm returns cluster(s) of detected position(s).

2. For each of these detected clusters, a cluster object is created with some attributes. These attributes are the name of the image where the cluster is detected, the recorded time of the image, members of the clusters, total number of cluster members, and the centroid of the cluster which is the mean position of cluster members. These cluster objects are then appended into a list.

3. In the next step, for each of the cluster objects, a tracker object is initiated with some attributes. These attributes include the names, detected positions, times and cluster objects associated with this tracker object. All of these attributes take the form of a list This tracker object is then updated, in case of any future detection associated with it. Each of these tracker objects is appended into a list.

4. In this step we iteratively choose each object from the tracker object list to estimate its associated position in the next frame. To do this we first find the velocity v and its magnitude $|v|$. One of the following three possible cases arise:

   - If tracker object = 1 detected position, then v=0. This happens only for the first frame.
   - If tracker object = 2 detected positions, then $v = \frac{\Delta S}{\Delta t}$. ΔS and Δt are the difference between the two detected positions and times, respectively.
   - If tracker object > 2 detected positions, then we use curve fitting upto maximum of 10 previously detected positions to find velocity.

5. The next step is to estimate the next expected position of the tracker object.
$$\text{next expected position} = \text{current position} + v * \Delta t$$

6. In the next step we measure the appropriate square sized search region. In our data the average width of the person is 64 pixels. We assume that the detected blob size should be greater than or equal to half of the width of the person. Therefore we start search for that blob whose size is 32×32 pixels. For the rest of the frames this square sized search region is updated by adding distance history of the already detected blob.
$$\text{Search region} = 32 + \Delta S$$

7. In the next step, this squared-size search region is used to search for the person in the next frame using the rHOG algorithm.

- In case of a detected position, this new position is associated with the existing tracker object if it is within a distance of 40 pixels from the last detected position. The tracker object is then updated by appending these new positions, and steps 4-7 are repeated.

  In case of a distance greater than 40 pixels from present and last detected position, a new tracker object is created, and steps 4-7 are repeated.

- In case of no detection found by the rHOG algorithm in the frame, the current tracker object is removed from the tracker object list. The remaining list is then checked. We repeat steps 4-7 in case of a non-empty list; otherwise the tracking process terminates.

In the tracking process using our dataset, the following situations can occur:

- Person-free frames or images at the start of the sequence
- The entering and leaving of people into and from the scene respectively.
- Due to abrupt change in time or missing frames, a person can have radically different positions before and after time gaps.

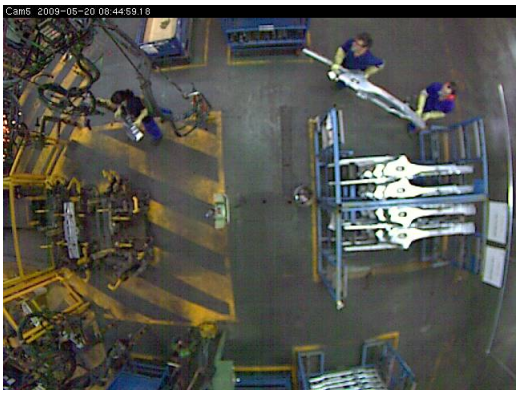
To make our algorithm faster, we incorporate a blob detection module in our basic tracking algorithm. The advantage of using a motion blob is that, instead of testing all un-neceaasry positions of the image, it enables us to search for potential regions of image where movements are found. We then search only those motion blobs for the existence of people in them. Doing this reduces processing time per image significantly. In the following section we discuss the blob detection module.

6.1.2 Blob Detection Module

The main intention of introducing blob detection module in our tracking algorithm is to make it faster by searching only motion regions of the image for the existence of a person. Worthy of note is that, our tracking algorithm is used to track people unlikely the OpenCV algorithms which use motion regions as moving blobs and then use those blobs for tracking. To extract motion blobs we use a process similar to that proposed in [8]. We explain this process step by step as below.

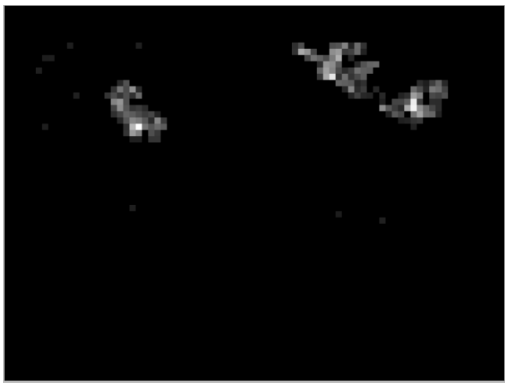
- The first step is to perform consecutive frame difference to get a motion map image. For two consecutive frames I_t and I_{t+1}, motion map image M_t at frame t is computed by taking the absolute difference between two frames using the following equation.

$$M_t = abs(I_{t+1} - I_t)$$



(a) Sample RGB image

(b) Motion map Image



(c) Data accumulated image

(d) highlighted small blobs of data accumulated image

(e) Segmented image

FIGURE 6.3: Blob Detection Process: During the blob segmentation process, small blobs less than 32×32 pixels were filtered out.

Figure 6.3(a) and Figure 6.3(b) shows an example of how a motion image is produced by absolute frame differencing.

- The second step is to perform an accumulation process on the motion map image. We divide the motion map image into sub-regions of 8×8 pixels. The reason for this value is that in our image scanning we use a maximum step size of 8 pixels in the vertical direction. The intensity values of pixels at each of these sub-region are then summed up and a threshold is applied. The resultant image is shown in

Figure 6.3(c). Different sub-regions in the form of small boxes can also be seen in this figure.

- The final step is to get a segmented image. We apply connected component analysis with four nearest neighbours; this produces different blobs and a size threshold is then applied. The blob above this threshold 32×32 pixels is chosen as a possible candidate for a person. The reason for this chosen value of 32 pixels is that we assume the minimum detected blob should not be less than half the width of the person which is 32 pixels. The small blobs below this threshold are highlighted in Figure 6.3(d). The segmented image after thresholding can be seen in Figure 6.3(e), where small blobs have been filtered out.

In this section we discuss briefly the implementation details of our proposed tracking algorithm. Our initial algorithm was using whole image-scanning to detect people which took much longer to track people. To reduce time complexity we append a simple blob-detection module which enables our algorithm to search for only potential movement area of the image. This reduced the processing time considerably. To compare the overall performance of our algorithm we use five standard algorithms of OpenCV. In the next section we briefly discuss each of these algorithms.

6.2 OpenCV built in Blob Detection Algorithms

We use five built-in algorithms of the OpenCV as a black box with default parameters. The documentation of the OpenCV blob tracking module can be found at [4] and [2]. An article providing an overview of the complete architecture of blob detection and tracking can be found in [16] and some useful, related material can be found at [3].

The blob tracking system used in OpenCV includes five modules as depicted in Figure 6.4. These modules are briefly discussed below.

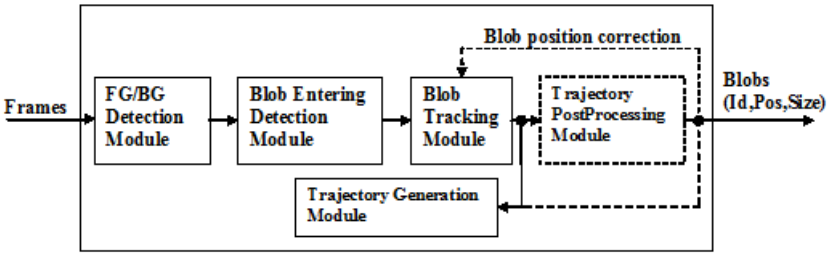


FIGURE 6.4: The block tracking modules of OpenCV (from [2]).

The FG/BG Detection module performs the segmentation of foreground and background for each pixel. In the Blob Entering Detection module the result of the FG/BG detection module is used to detect a new blob entry on each frame and this result is then used in

the Blob tracking module to track each newly detected blob. The Trajectory Generation module combines the result of all blobs positions and stores these into secondary memory at the end of the tracking process. Finally, the Trajectory Post Processing module which is optional is used to perform smoothing function on blob trajectory using a Kalman Filter [116]. The detailed explanations of each of these modules can be found in [2].

In the following sections we briefly discuss the Mean-Shift algorithm of OpenCV. The details about the remaining four algorithms are discussed in brief in Appendix-C.

6.2.1 Mean-Shift Method

The Mean-Shift method is based on using colour intensity histograms. Mean-Shift is a non-parametric feature-space analysis technique, a so-called mode seeking algorithm [17]. It has been used widely in computer vision for a range of purposes like cluster analysis and image processing [24]. It was introduced by [46]. Mean-Shift considers feature space as a probability density function. It is a procedure which locates the maxima of a density function with given sample data. It is an iterative method and useful for detecting modes of density function. It usually works in three specific things: (1) A distance function is used for measuring distances between pixels. This distance function could be Euclidean distance or some other distance functions like Manhattan Distance; (2) a fixed radius h or window: All pixels within this radius or window will be used for mean calculation, and (3) it then shifts the centre of h to the mean and repeats this process until converges. The end of each iteration causes window shift to a more dense region of the dataset. Figure 6.5 shows the process of the Mean-Shift algorithm.

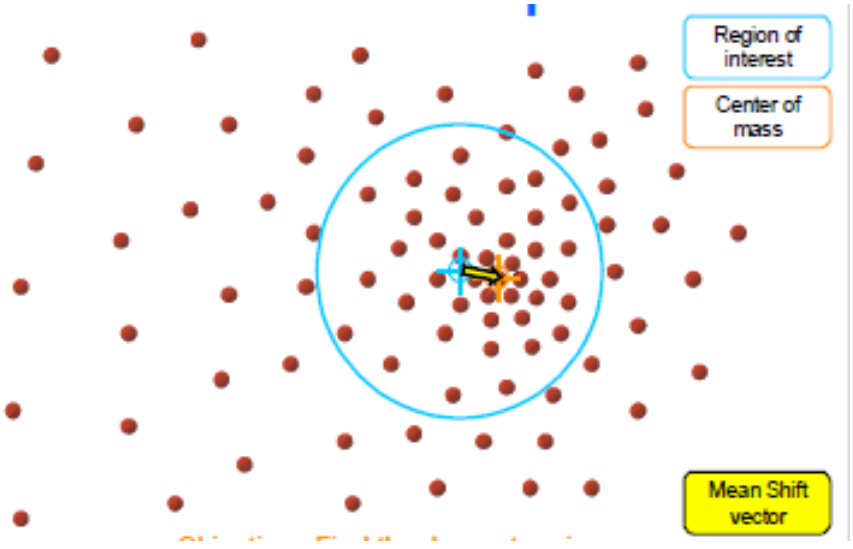


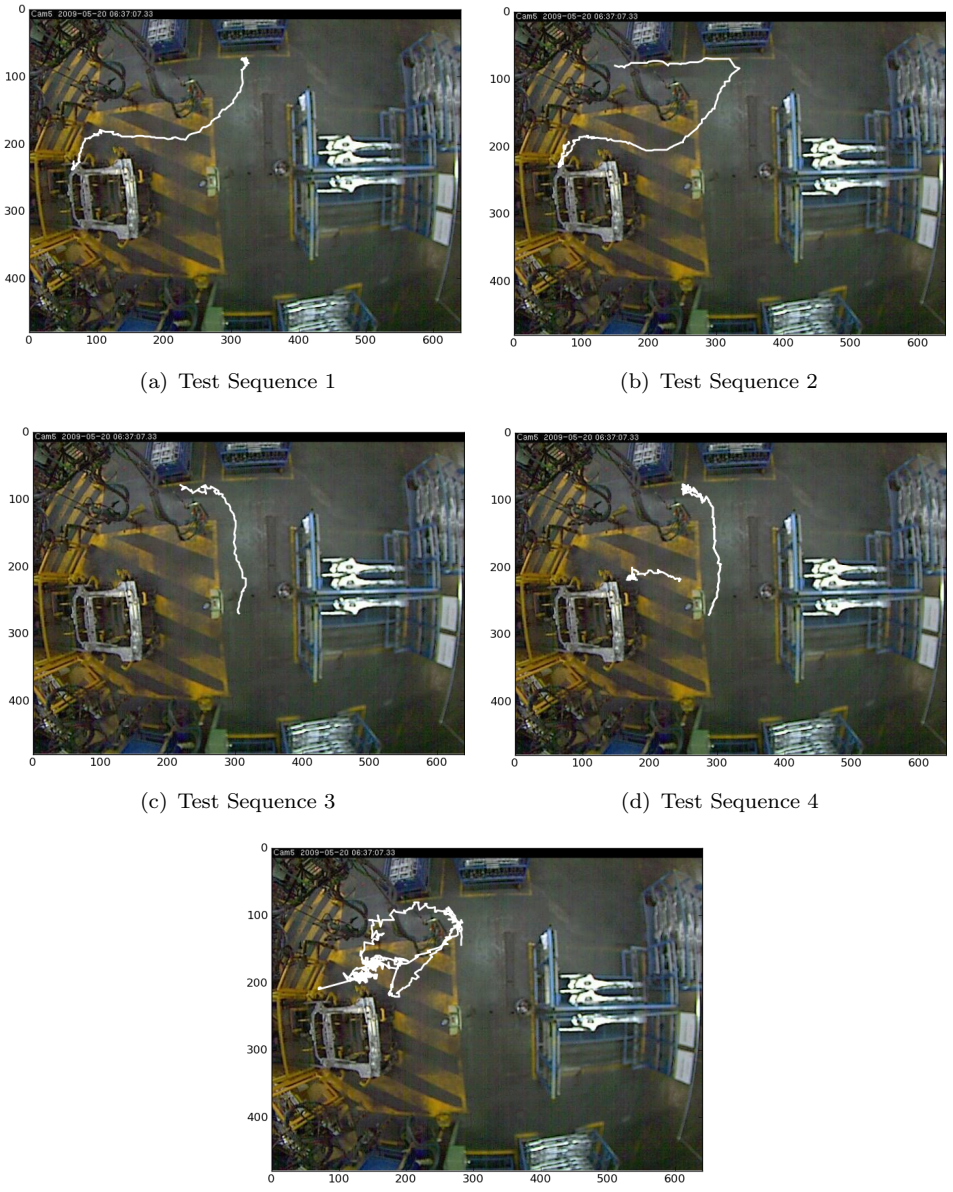
FIGURE 6.5: The Process of the Mean-Shift algorithm (from [23]).

Although MS is a non-parametric algorithm, it still requires the bandwidth parameter or radius or window h to be tuned, (KNN could be used to find out this parameter). The choice of h influences the number of clusters and convergence rate. A large h

might result in incorrect clustering and might merge distinct clusters while a very small h might result in too many clusters. Mean-Shift is used in an object's colour based tracking because it is simple and robust.

In this section we discussed in brief the Mean-Shift algorithm of the OpenCV. In the next section, we discuss the results of the OpenCV algorithms when tested with our five test sequences.

6.3 Experiments and Results



(a) Test Sequence 1

(b) Test Sequence 2

(c) Test Sequence 3

(d) Test Sequence 4

(e) Test Sequence 5

FIGURE 6.6: Trajectories of five test sequences

To compare the results of our algorithm with OpenCV algorithms we use five different test sequences. Each of these sequences is labelled manually; their trajectories can be seen in Figure 6.6. In the first test sequences the person picks out a steel rod from the top middle rack, carries it along the path shown in Figure 6.6(a) and places the rod at the assembling plant of the car. The second sequence is similar to the first except that the person enters into the scene from the top left position and walks toward the top middle rack. At the start there are four small jumps of 0.8, 0.65, 0.55 and 0.56 seconds, respectively, but before and after each of these time gaps the person remains nearly in the same position. In the third test sequence the person simply walks from the middle to the top of the scene. In this sequence the person is neither carrying nor handling anything. In the fourth test sequence, the person walks from the middle of the scene to the top and then walks towards the assembling area. During this, there is one big data gap of about 2.7 seconds and because of this time jump the person's position in the scene shifts from the top middle to the middle near the welding stand, as shown in Figure 6.6(d). He then walks towards the car assembling plant and interacts with the plant. During this time the person makes no significant movement from his original position. The last sequence is the most difficult. In this case, the person picks up the welding machine and moves toward the car assembling plant; he then starts welding steel rods and during this whole sequence sparks arise from the welding machine. There are also some time gaps in this sequence. Most of the time, the person stays near the rack and welds different parts of the rod by moving around the upper side of the car assembling area. This situation is depicted in Figure 6.7. The sparks that are generated during this welding process can also be seen in these images.

(a) Frame 50 (b) Frame 171 (c) Frame 1018

FIGURE 6.7: Some samples from sequence 5.

We first compare our results with OpenCV algorithms using the first four sequences. The fifth sequence is a very difficult one and also has big time gaps, which is why we discuss it separately. We use the default parameters of OpenCV algorithms because this would make it easy for anyone to draw comparisons. Here, two factors are noticeable; the first is that in these OpenCV algorithms we are not filtering small blobs, but in our tracking algorithm we are filtering out all those blobs which are less than 32×32 pixels. The reason for this chosen size is to keep it consistent with the half width of the person, which is 32 pixels. The other factor is that we observe the normal blobs in green colour

and abnormal blobs in red colours using OpenCV algorithms. We are not eliminating or filtering out any abnormal blobs which are detected near the positions of people in images.

Among the test sequences, the first one is the easiest, because this sequence has no data gaps. When we plot all the detected positions produced by the Connected Component Mean Shift Particle Filter (CCMSPF) algorithm using this test sequence, the output (as shown in Figure 6.8) is not continuously detected positions in contrast to other algorithms which produce continuous detected positions for the whole test sequence. We want to draw a uniform comparison without manipulating the source code or default setting of these OpenCV algorithms. Therefore in the following discussion of the results, we do not consider this algorithm.

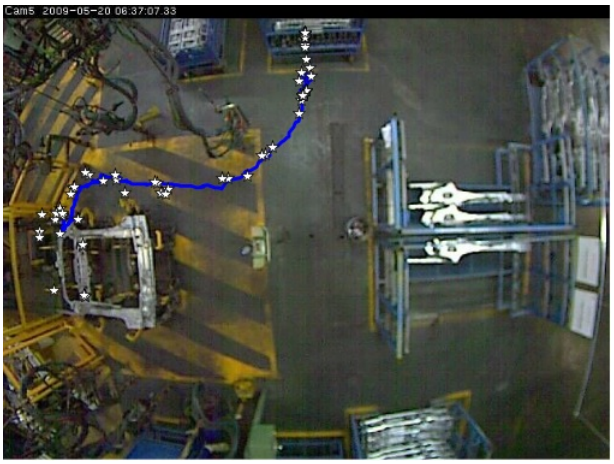


FIGURE 6.8: The detected positions produced by the CCMSPF algorithm are shown in white. The blue line shows the trajectory of actual positions of the person.

When we test the first sequence for each of these algorithms, the result is a list of detected positions: each of these positions could be a true position of the person in the image or a wrong detection. To check the validity of a detected position, we use a distance threshold, which is the distance in pixels between a detected position and a ground truth or actual position. We measure different detection results by varying the detected positions in a range of 15-50 pixels, from the ground truth. These results are shown in Figure 6.9. The horizontal axis shows the distance of all detected positions from the ground truth. The associated hit rate or detection rate is shown in the vertical axis. For our algorithm the results are shown by taking a mean and standard deviation using all four test sequences.

It can be seen from this graph that our algorithm has a much higher hit rate than others. We chose a distance threshold of 20 pixels because at this threshold (TH) our algorithm has a hit rate close to 100%. Among the OpenCV algorithms, the Mean-Shift (MS) algorithm performs relatively better with a hit rate of 64%. Therefore in the rest of comparison we use the MS algorithm against our algorithm.

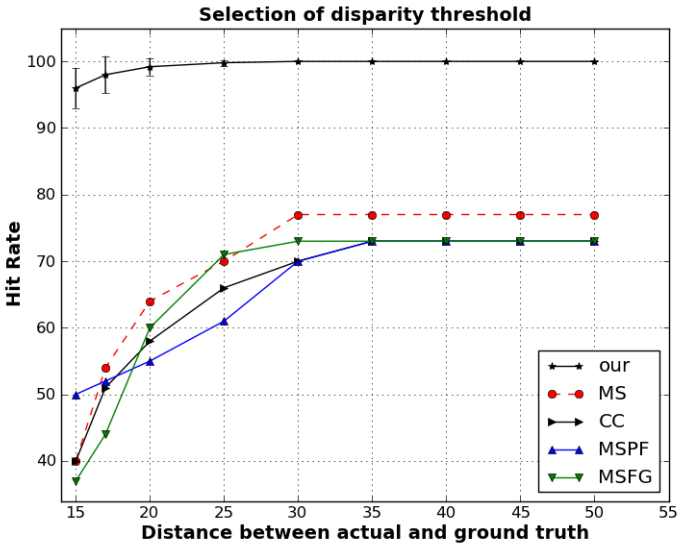


FIGURE 6.9: Selection of distance threshold to decide validity of detected positions.

The results of each test sequence are described using six graphs. The first three graphs correspond to the Mean-Shift algorithm while the last three correspond to our algorithm. Each of these six graphs is labelled in a range, from (a)-(f). The first graph (a) represents all ground truths shown as a gray line, the detected positions within the threshold are shown as a blue plus symbol and detections above the chosen threshold shown as a red diamond symbol. The next two graphs are used to plot time against detected y positions and x positions to understand what is happening with respect to time and position. For the next three graphs we depict the results information in the same sequence as discussed above but using our tracking algorithm. At the end of each sequence we present the results of the Mean-Shift and our algorithm, respectively, using a table. We also present a mean distance d which is the average distance between all detected positions and the ground truths. At the end, we summarise the overall results of using all the OpenCV algorithms and our tracking algorithm. In the following sections we describe the results of each test sequence.

6.3.1 Test Sequence 1

When we use the Mean-Shift algorithm, there are no detections at the start of the sequence and the blob detections start after the 30th frame. When the person walks by holding a steel rod, the MS algorithm estimates a blob by considering the person and rod as one single object. In that case the blob centroid becomes further away from the actual centroid of the person; this causes the detected blob position to move beyond the threshold, and as a result such a position is classified as not detected. This kind of detection is shown in red in the graph, as shown in Figure 6.11(a). The detected x and y positions against time can be seen in Figures 6.11(b) and 6.11(c), respectively. The

positions detected in the last segment are away from the actual positions; the reason for this is that, in these frames, the person is placing the rod on the assembly plant. At this stage the person is almost stationary while moving the rod. The MS algorithm detects this rod as a moving blob and keeps detecting the rod instead of a person as long as the person remains stationary. One such example is shown in Figure 6.10 where the steel rod is considered as a moving blob by the MS algorithm and is shown in a green ellipse.

FIGURE 6.10: Test sequence 1. In this image, instead of a person, a steel rod is considered as a moving blob (shown in a green ellipse) by the Mean-Shift algorithm.

On the other hand, if we compare these results with the results produced using our algorithm as shown in Figure 6.11(d)-(f), we can see that the detected positions are quite close to the ground truths and that, compared to the MS algorithm, there are no Mis-Detections at the start, middle or end of the trajectory. The reason for this is that we use a tracking-by-detection-algorithm, which first searches for a motion-region or motion-blob and then scans that blob for the existence of a person. Therefore, in the case when a detected blob includes a person and rods, our algorithm filters out all instances other than a person. Thus, our algorithm not only detects the blob, but also has a strong discriminative power of distinguishing a person with any other object in the background. The results of detected positions are shown in Table 6.1. The mean distance between the actual and detected positions within the chosen threshold is also presented in this table. Our algorithm has a Hit rate of 99.5% with a mean distance of 6 pixels while the Mean-Shift algorithm has a Hit rate of 64% with a mean distance of 11 pixels. This confirms that our algorithm not only performs best but also that the detected positions are close enough to ground truths with a deviation of about 6 pixels compared with the performance of the Mean-Shift algorithm, which has a deviation of 12 pixels.

TABLE 6.1: Detected positions within threshold using Test Sequence 1.

Algorithm	Mean distance in pixels	Hit Rate
Our	6	99.5 %
Mean-Shift	12	64.0 %

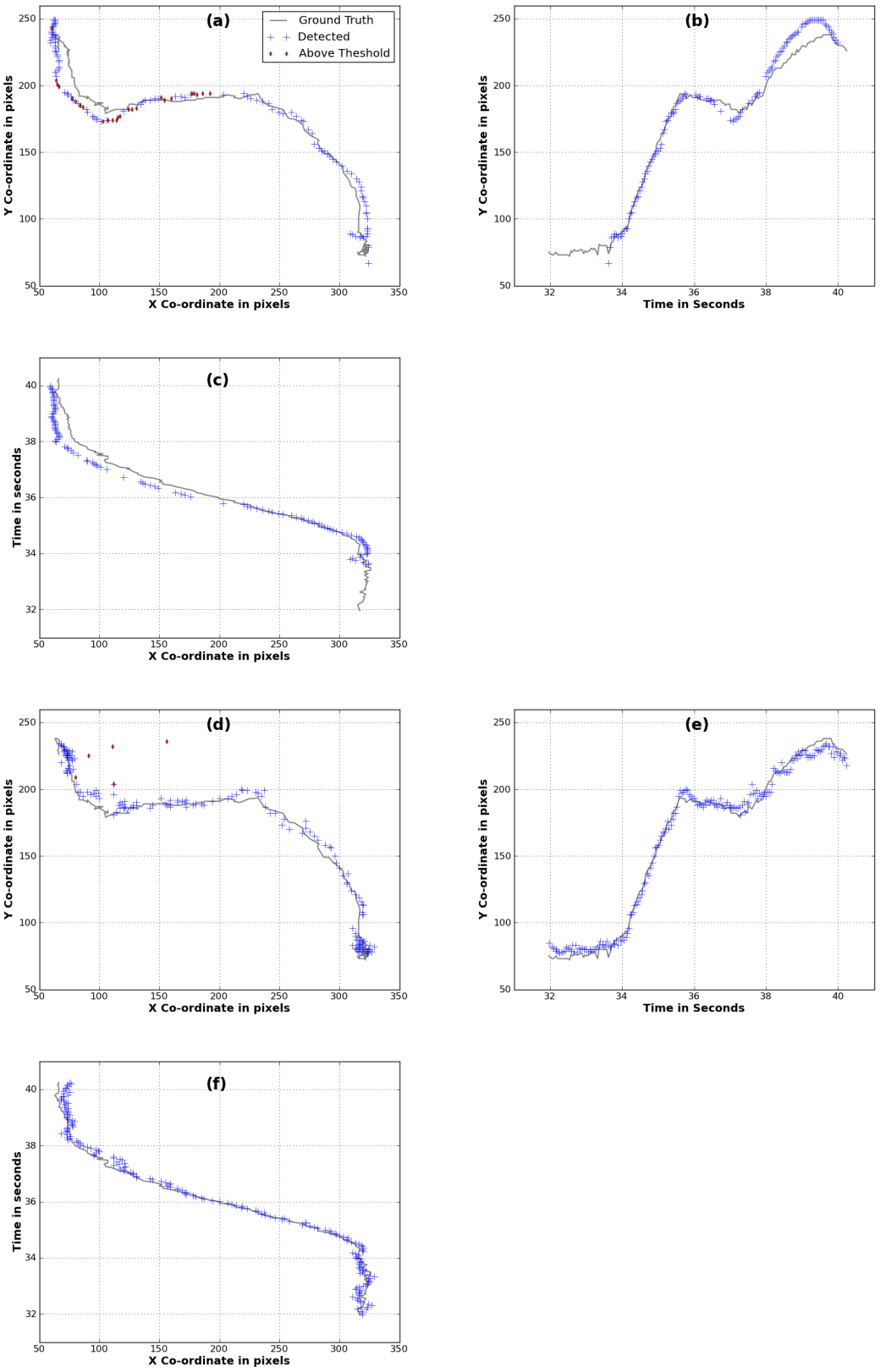


FIGURE 6.11: Results of test sequence 1 using Mean-Shift (a,b,c) and our algorithm (d,e,f).

6.3.2 Test Sequence 2

At the start of this sequence, there are four small jumps of 0.8, 0.65, 0.55 and 0.56 seconds respectively, but during these time gaps the person remains nearly in the same position and Mean-Shift enables us to track the person during these jumps. These time gaps can be seen in Figure 6.13(b)-(c). Another outcome is that when the person is moving his arm to a large extent at the start or at the end of this test sequence, then during this, most of the time, the Mean-Shift algorithm detects a blob near the elbow of the person and this effect can be seen in Figure 6.12. In such a case the centre of the detected blobs (arms in this case) moves away from the centre of the person and results as a not detected position. These positions can be seen in Figure 6.13(a) in red. In the middle of the sequence the person was not detected for a couple of frames which could be due to a weakness in the algorithm. From the graph it can be seen that even the detected positions are away from the ground truths, mostly at the start and end of the sequence. In the start between x coordinates 250-300 the detected positions are away and the reasons is because, most of the time the Mean-Shift algorithm detects the arm of the person and the same effect can be seen at the end of the sequence where this algorithm detects the rod most of the time. Two such examples can be seen in Figure 6.10 and Figure 6.12. The detected blobs by Mean-Shift algorithm are shown in green ellipses in these examples.

FIGURE 6.12: Blob detection by Mean-Shift shown in green ellipse, when using our algorithm, it is shown by a red circle. The white circle is the ground truth position. The horizontal white line shows edge of the area of the image where the algorithm can be applied.

The results of this sequence using our algorithm can be seen in Figure 6.13(d)-(f). Our algorithm has a smooth trajectory, showing that it detects the person in almost every frame. The detections near y-axis around 100 pixels are a small distance away from the ground truths. For example, in Figure 6.12, the ground truth is shown in a white circle and detection by our tracking algorithm is shown in a red circle. The horizontal white line shows edge of the area of the image where the algorithm can be applied.

The results are shown in Table 6.2 using both algorithms. Our algorithm achieves a

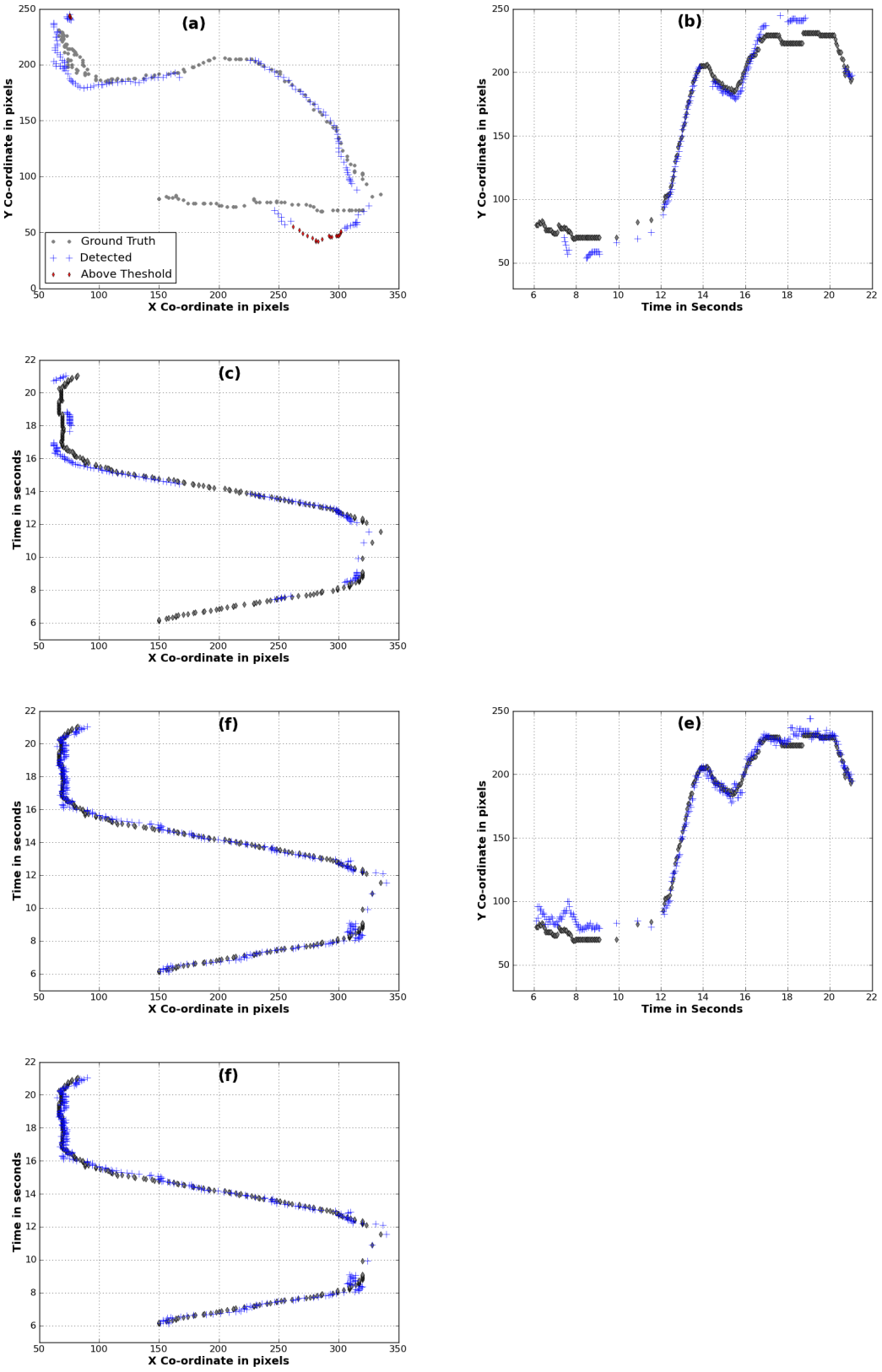


FIGURE 6.13: Results of test sequence 2 using MS (a,b,c) and our algorithm (d,e,f).

TABLE 6.2: Detected positions within threshold using Test Sequence 2.

Algorithm	Mean distance	Hit Rate
Our	7	97.0 %
Mean-Shift	13	55.0 %

hit rate of 97% while for the Mean-Shift algorithm the hit rate was 55%. The mean distance of detected positions using our algorithm and the Mean-Shift algorithm are 7 and 13 pixels, respectively.

6.3.3 Test Sequence 3

FIGURE 6.14: Test sequence 3. A very small detected blob near the shoulder of the person. The detected positions using Mean-Shift and the ground truth are shown here with green and white circles respectively.

The Mean-Shift algorithm misses a lot of positions in this sequence. The detected blobs are found to be very small and positions of those blobs are further away from the ground truths. One such example is shown in Figure 6.14. The green and white ellipses correspond to the detected blob and the actual centroid, respectively. It can be seen that the detected blob using Mean-Shift is very small and only occupies a very small area of the person's body. This effect was observed most of the time in this sequence. The sizes of these kind of blobs were 8×20 pixels. As discussed above, we did not manipulate any of the OpenCV tracking algorithm parameters. In our algorithm, however, we are not considering such small detected blobs even if they appear on the exact centroid of the person. The reason for not doing so in our algorithm is that, we assume that the minimum blob size should be atleast half the width of the person, which is 32×32 pixels. This means that we filter out all detected blobs less than a 32×32 pixels size in our algorithm. The blob detection results are shown in Figure 6.15(a)-(c). It can be seen that the Mean-Shift algorithm fails completely at the start and end of this sequence. In the middle of the sequence it detects blobs for a couple of frames, but even these detected blobs are a distance from the ground truth.

If we analyse the results produced by our algorithm as shown in Figure 6.15(d)-(f), it

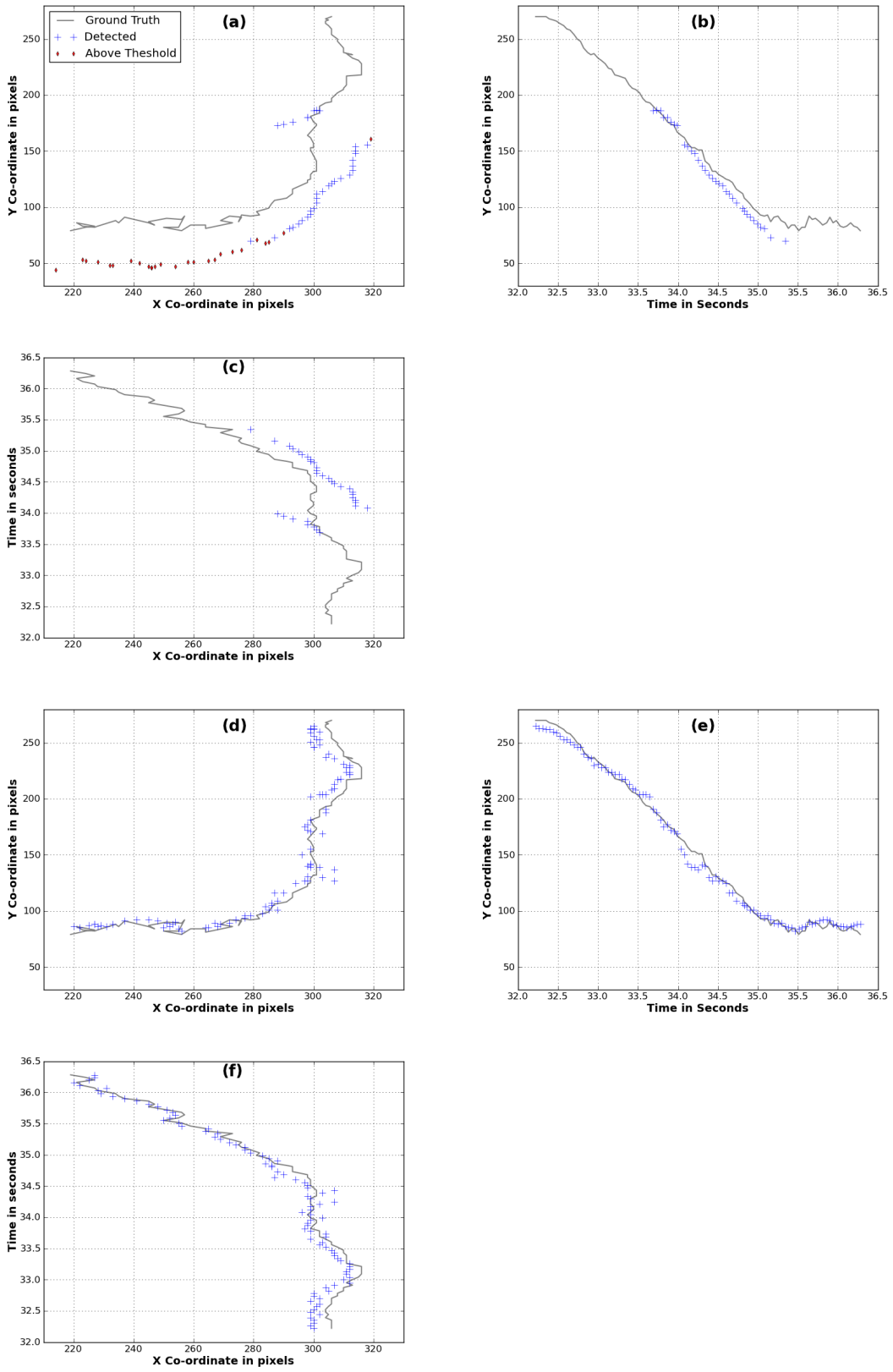


FIGURE 6.15: Results of test sequence 3 using MS (a,b,c) and our algorithm (d,e,f).

can be seen that our algorithm not only detects all the positions but also that these detected positions are very close to the ground truths. Our algorithm detects 100% with a mean distance of about 6 pixels while MS correctly detects only 35.8% with a mean distance of about 13 pixels. These results are shown in Table 6.3.

TABLE 6.3: Detected positions within threshold using Test Sequence 3.

Algorithm	Mean distance	Hit Rate
Our	6	100.0 %
Mean-Shift	13	35.8 %

6.3.4 Test Sequence 4

In this sequence the Mean-Shift tracking algorithm starts tracking from the 12th frame and can be seen in Figure 6.17(a)-(c). The detected blob sizes are very small and are not located consistently around the persons body. In most cases, these small blobs are detected away from the centre of the person, i.e on the corner of shoulder and moving towards the arm of the person. Some sample images with detected positions (shown in green) using the Mean-Shift algorithm are shown in Figure 6.16. As these images show, most of the time we observe a very small blob, relative to the size of the person, in the image. Furthermore the positional inconsistency of detected blobs can also be seen in these images. Because of this positional inconsistency and distance from the ground truth, the majority of these positions are categorised as not detected and are shown in red in Figure 6.17(a). There is a time gap of about 20 seconds as shown in Figure 6.17(b)-(c) after 191 frames. The Mean-Shift algorithm starts detection from the 6th frame after this big time gap. These detections were mostly exactly central on the person and the size of the detected blobs were consistent to the size of the person. One such example can be seen in Figure 6.16(d), where the person after some time becomes almost stationary, and the Mean-Shift algorithm fails to locate the blob here. These effects can be see in the graphs shown in Figure 6.17(b)-(c).

(a) Frame 22 (b) Frame 40 (c) Frame 131 (d) Frame 201

FIGURE 6.16: Some samples from sequence 4. Detected blobs using Mean-Shift algorithm are shown in green colour.

Compared to Mean-Shift, our algorithm starts detection from the very start, and the detected positions are very close to the actual positions of people. Our algorithm is capable of handling large time gaps and continues to track during the long time jump of

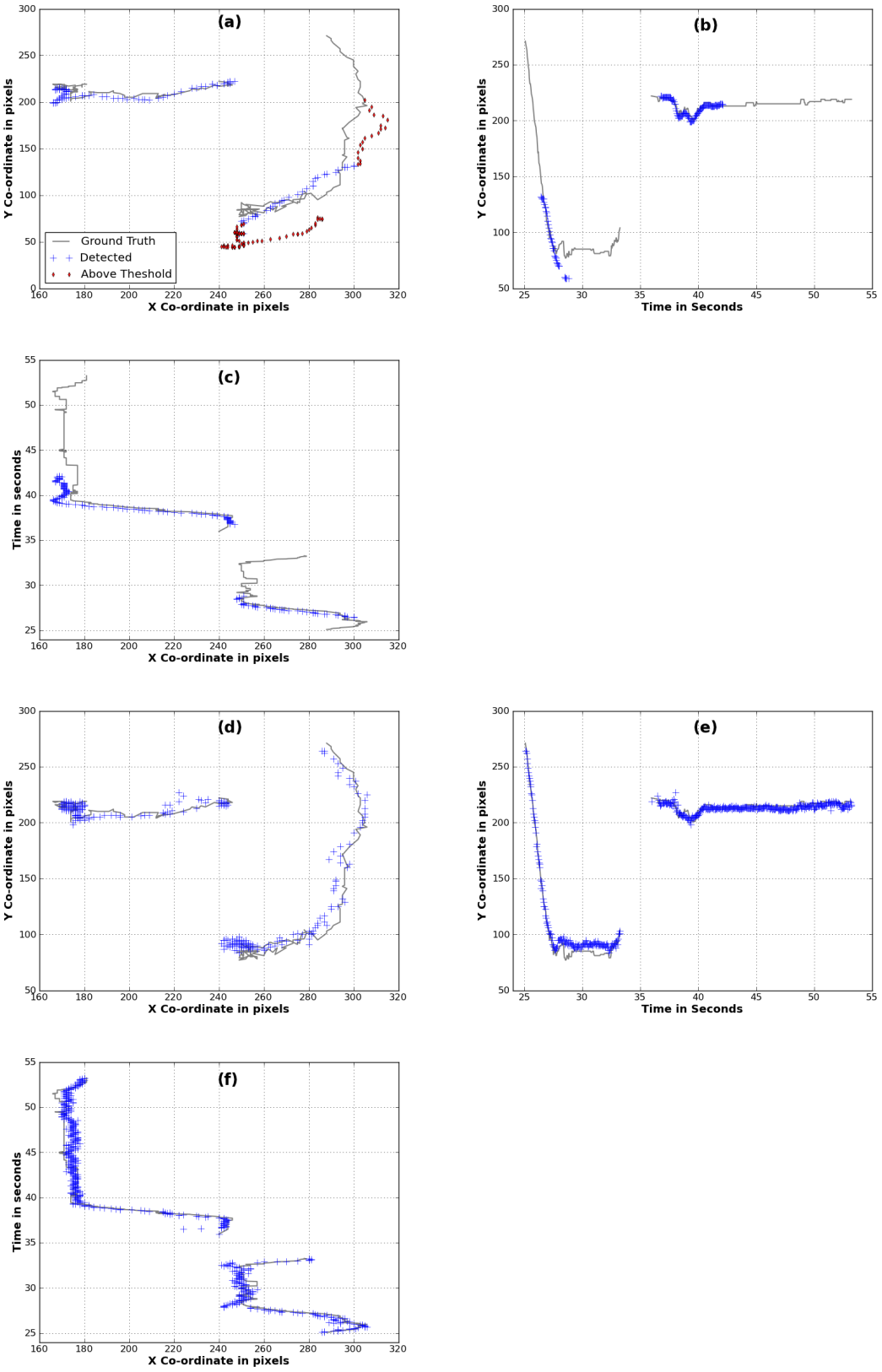


FIGURE 6.17: Results of test sequence 4 using MS (a,b,c) and our algorithm (d,e,f).

20 seconds. Another point to note is that even when the person is not moving much at the end of the sequence, our algorithm detects the person in all of those frames where the Mean-Shift algorithm failed to do so. The trajectory of detected positions using our algorithm is quite close to the ground truth with almost all correctly detected positions. The detection behaviour using our algorithm can be seen in graphs shown in Figure 6.17(d)-(f). The results of the correctly detected positions using both algorithms are shown in Table 6.4. Our algorithm has a detection rate of 99.8% with a mean distance of 5 pixels from ground truth; while the result achieved using the Mean-Shift algorithm is only 38.70% with a mean distance of 9 pixels.

TABLE 6.4: Detected positions within threshold using Test Sequence 4.

Algorithm	Mean distance	Hit Rate
Our	5	99.8 %
Mean-Shift	9	38.70 %

Overall results of four test sequences using OpenCV Tracking Algorithms

Here we summarise the results of each of the four test sequences using all OpenCV algorithms. Table 6.5 presents a summary of these results. The first column of the table shows different tracking algorithms. In this table the detection rate or Hit Rate is denoted by HR, whereas the average mean distance of detected positions is represented by md. From the table we can see that our algorithm outperforms all other OpenCV algorithms with an average detection rate of 99.08%. Among others, the best one is the Mean-Shift algorithm with an average detection rate of 48.38 %. These average results show that there is no significant variance of results among OpenCV algorithms. The overall behaviour of all the OpenCV algorithm is almost the same except for sequence 4 where the Connected Component algorithm produced the best detection results i.e 67%. The reason is that in this sequence the said algorithm detects stationary person most of the time when rest of OpenCV algorithm fails. Similarly, if we see the other parameter i.e average mean distance of detected positions, our algorithm produces a result of about 6 pixels, whereas the second closest average distance is 10 pixels using a MSPF algorithm. The overall conclusion is that our algorithm achieved significant detection results compared to all other OpenCV algorithms.

6.3.5 Test Sequence 5

We discuss test sequence 5 separately because it is a very difficult sequence with different data gaps and dynamic changes in the environment in the images. In this sequence there are 1021 frames with some data gaps. Most of the time during these time gaps the person remains at a distance of about 40 pixels in the same direction except one long jump of

TABLE 6.5: Overall results of all 4 sequences.

Algo	Test-1		Test-2		Test-3		Test-4		Average	
	md	HR	md	HR	md	HR	md	HR	md	HR
Our	6	99.5	7	97	6	100	5	99.8	6	99.08
MS	12	64	13	55	13	35.8	9	38.7	11	48.38
CC	12	57.6	12	42.1	10	25.3	13	67	12	48.0
MSFG	13	59.7	12	50	12	30	10	43.6	12	45.83
MSPF	9	55.5	13	32.1	9	42.1	7	32	10	40.43

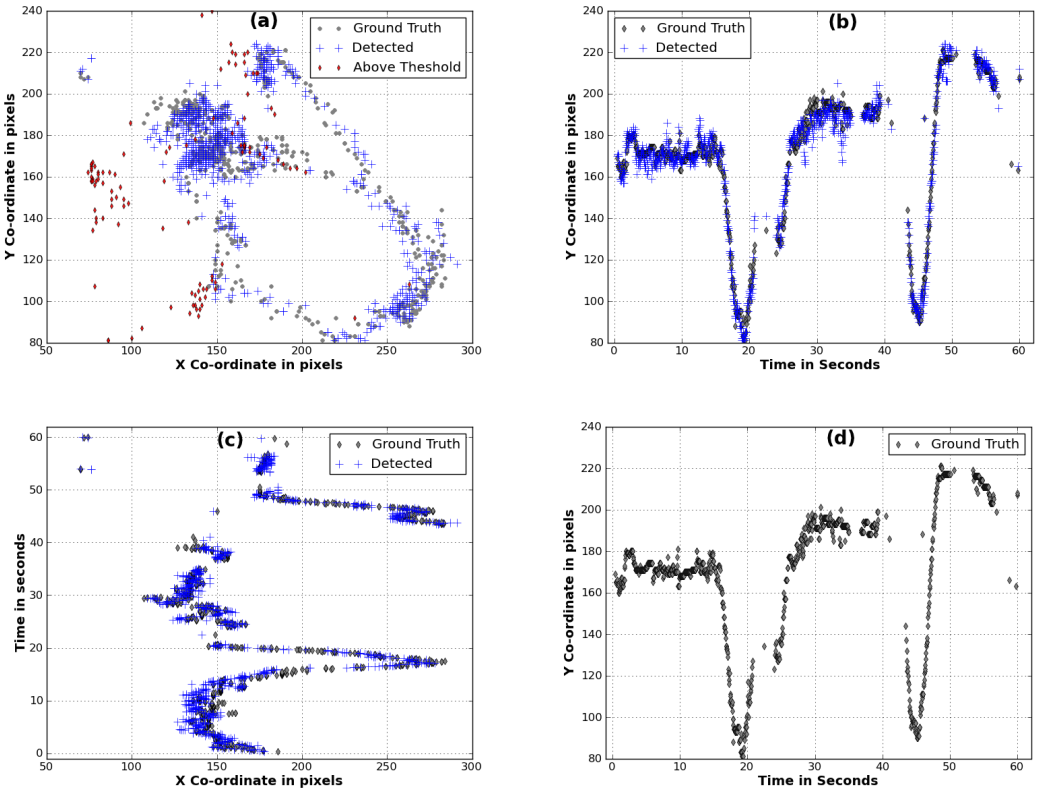


FIGURE 6.18: Results of test sequence 5 using our algorithm.

about 8 seconds when the person moves 100 pixels away from his previous position in the same horizontal direction. The total recording time in this sequence is about 76 seconds. On average there are missing data of about 30 seconds with a total number of about 730 missing frames. Most of the time, the person is occluded with wires, sparks and machines. In this sequence the person has a welding machine in his hand and he undertakes some welding activity, which generates sparks in the scene. Some sample images are shown in Figure 6.7. We compare the performance of our algorithm with all four OpenCV algorithms. The trajectories of detected positions are shown in Figure 6.18. In the graph shown in Figure 6.18(a) it can be seen that mostly detected positions are close to the ground truth. In the graph there are some above- threshold positions which are away from the ground truth. These positions are actually false detected positions caused by the flame of the welding machine. The ground truths (y

coordinates) against time are shown in Figure 6.18(d) to clearly demonstrate the time gaps in this sequence. The trajectories of detected positions against time along x and y positions are shown in Figure 6.18(c) and Figure 6.18(b), respectively. From these time graphs we can easily analysed how well our algorithm detects a person even with large time gaps, and the results for this are shown in Table 6.6. Our algorithm retains its excellent detection behaviour even with this most difficult sequence. The best result produced by the MSFG algorithm is only 31.2 %, whereas our algorithm correctly detects a person 97.8 % of the time.

TABLE 6.6: Results using Test sequence 5.

Algorithm	md	HR
Our	9	97.8 %
MS	11	21.7 %
CC	14	22.1%
MSPF	11	23.7 %
MSFG	11	31.2%

6.4 Execution Time

In this section we discuss the total processing time per sequence using OpenCV algorithms and our algorithm respectively. The performances of the five test sequences are presented in Table 6.8. As our algorithm is not optimised and uses two separate programs written in C++ and python respectively to produce overall tracking results. To identify a detected blob as a person/ no-person we use the rHOG algorithm written in C++, and for the rest of the processing we use the code written in python. The processing time calculation is undertaken in the same way for both algorithms to assure consistency. For our algorithm we calculate the processing time by concatenating the segments as shown in Table 6.8. In this table the first column describes five different test sequences. The total number of new blobs detected per sequence are presented in column 2. Column 3 of this table describes the total number of search regions per sequence while the total number of detection windows (DWs) per search region are shown in the next column. The initial constant processing time per sequence is presented in the fifth column. The values in column 6 of the table is the processing time for classifying these detections window as a person/no-person using the rHOG algorithm. Column 7 presents the processing time used by our tracker, which includes all sub-processes of blob detection, formation of tracker object and so on. The eighth column is the overall total processing time of our algorithm.

We first compare the execution time using OpenCV algorithms run on test sequence 1 only. The result is shown in Table 6.7 including the times taken by different algorithms to process test sequence 1. Throughout, we have compared our results with the MS algorithm; therefore, for the uniform comparison here we also compare results using just

the MS algorithm. Table 6.9 presents the processing time of five test sequences using our and Mean-shift algorithm, respectively.

The overall conclusion is that, compared to all built-in algorithms of OpenCV, our algorithm outperforms in accuracy of detections and performance.

TABLE 6.7: The processing time using all OpenCV algorithms tested on sequence 1.

Algorithm	MS	CC	MSPF	MSFG
Time (in seconds)	29	24	63	29

TABLE 6.8: Processing time of five test sequences using our algorithm.

Seq	Blobs	Search regions	DWs	Initial	rHOG	tracking	Total
1	93	271	32	4.25 s	5.4 s	6.7	16.35 s
2	91	358	32	4.25 s	8.8 s	8.1	21.15 s
3	22	174	28	4.25 s	3.8 s	2.6	10.65 s
4	33	620	30	4.25 s	14.4 s	16.8	35.45 s
5	394	1663	40	4.25 s	30.3 s	64.0	94.3 s

TABLE 6.9: Comparison of processing time of five test sequences using our and Mean-Shift algorithm, respectively.

Sequence No.	Our algorithm	Mean-Shift algorithm
1	16.35 s	29 s
2	21.15 s	42 s
3	10.65 s	15.6 s
4	35.45 s	88 s
5	94.3 s	166 s

Discussion

In this chapter we discuss our simple tracking by detection algorithm. Our main intention is to demonstrate the performance of our person detection algorithm. We incorporate a simple motion detection frame work to make this faster. The detected motion-blobs are then scanned with our rHOG algorithm for the existence of a person. The expected position of a person in the next frame is estimated using the history of the currently tracked person. We use five different test sequences of varying complexity and compare the results of our algorithm with five standard OpenCV algorithms. Our algorithm achieved far more significant results than the OpenCV algorithms did. Furthermore, our algorithm can detect a stationary person, handles occlusion and keeps tracking even when there are large time gaps.

Chapter 7

Generalising the Person Detector

In Chapter 5 we discussed the detection of people in images from an overhead view using a wide angle overhead camera. We have shown experimentally that our proposed algorithm produced attractive results; however, the main self-criticism was that we used only one specific domain for the person detection system. We did not test the detection of people in a changed environment with different camera settings; e.g, field of view or height. Changing these parameters could have a significant impact on the size of the person being viewed. In this chapter we address all of these issues with the experimental results. The main conclusion is that our algorithm has an excellent generalisation ability and, applying a pretrained model using one dataset, can detect people in a dataset relating to a completely different domain.

In Section 7.1 we discuss our new overhead dataset which differs completely from the SCOVIS dataset we used previously. Section 7.2 demonstrates the criteria of bounding box size selection. In Section 7.3 we discuss the results of the Person Classification and Person Detection in images, respectively, using two different overhead datasets. In these experiments we use different combinations of these datasets for both training and testing to demonstrate the generalisation performance of the person detector. We have shown experimentally the excellent capability of automatic labelling of people by our proposed person detector. The last experiment demonstrates that our algorithm is to a certain extent invariant against the change in a person's size and the position of camera.

7.1 Southampton Dataset

We recorded a new dataset at the University of Southampton. It comprises about 40 minutes recording with 15 frames per second. The size of the image is 1024 x 768 pixels. The camera used was a Point Grey camera with a Fujinon zoom lens having a focal length of 1.4-3.1 mm. The height of the camera was approximately 4 metre from

the floor. The calculation of the field of view of the camera was difficult because the regions at the edges are surrounded by walls. We measure it based on the maximum diagonal distance from the centre to edge of the scene. This distance was measured as approximately 9 metre and is shown as a green line in Figure 7.1. In this way the manually measured field of view of the camera was approximately 132^{o}. To find the centre of the image we use a mirror and the image's centre was then chosen when the camera was viewed in the mirror. This setup can be seen in Figure 7.1. The camera was mounted at the gallery of the second flour with the help of a wooden plank. The small mirror is placed in, about the middle of the image and the overhead camera can be seen in the mirror.

Our original intention was to record the real data for multiple people without their knowledge, when they moved around the area; however, due to certain issues, such as ethical approval of all stake-holders and the data protection act it was not possible to arrange in the time available. Therefore the author decided to video himself; this is the reason, for the limitation of this dataset in that there is only one person per image.

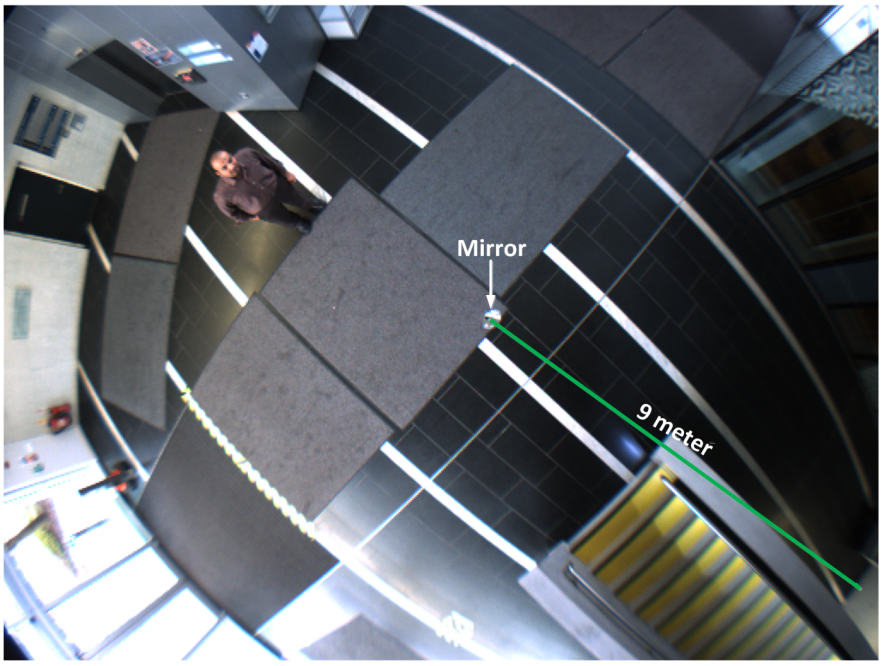


FIGURE 7.1: Locating optical centre of the camera in the image using a mirror. The green line represents the diagonal distance from the center to the edge of the scene and is used to measure the field of view of the camera.

In total, we use 10,000 images for different training and testing. Among these images, 7100 contain one person in and the rest are person-free images. These images are labelled manually and are in a combination of 14 different sequences. The total number of images in each sequence is shown in Table 7.1. The reason for the variable number of images in each sequence is because, during recording, the person enters and leaves the scene, and from these sequences we used only those images which contain a person.

TABLE 7.1: Data sequences.

Sequence	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Images	123	588	885	511	234	646	637	186	207	606	783	238	732	695

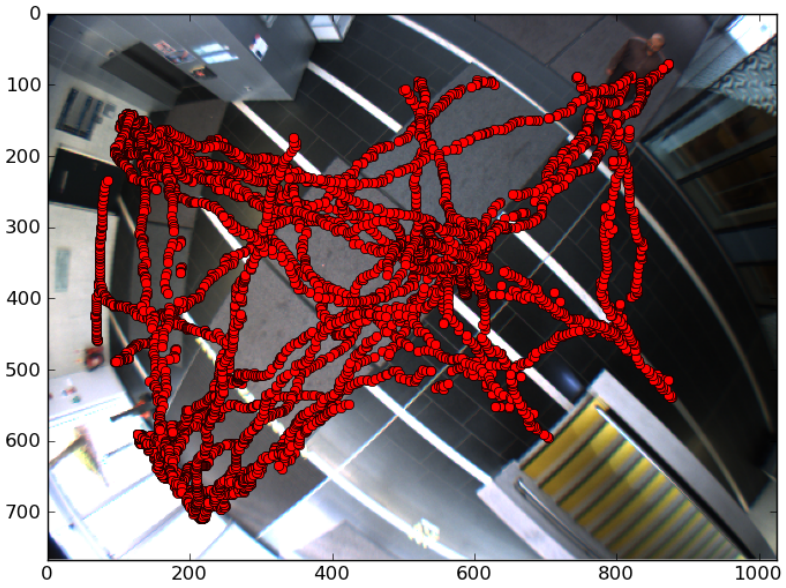


FIGURE 7.2: Different positions of a person in the scene using all 14 data sequences of 7100 images.

The overall trajectory of the positions of a person in all of these sequences can be seen in Figure 7.2. While recording the data the author wore different clothing and adopted different poses to reduce the limitations of specific clothing and pose. Some sample examples at different positions of the scene are shown in Figure 7.3. The same person can be seen wearing different clothing, appearing with and without a cap, and using different body articulations. In the rest of our discussion, we refer to this dataset as the SOTON dataset.

7.2 Bounding Box Size Estimation

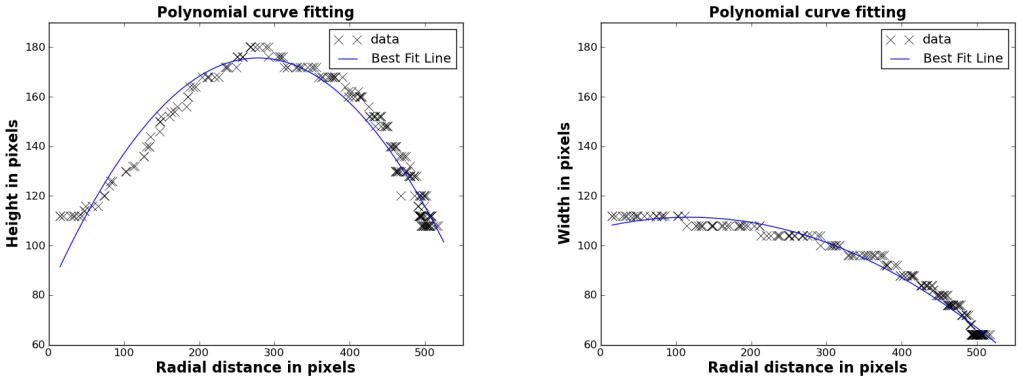
To estimate the size of the bounding box, we use about 200 images with the person at different positions, starting from the centre and extending to the edge of the scene. We then manually measure the height and width of the person at each of these locations and the radial distance from the centre of the image. The minimum width and height of the person were measured as 64 and 108 pixels, respectively, whereas 120 and 180 pixels are the maximum width and height of the person. On average, the width and height of the person are 90 and 140 pixels respectively. The real height of the person in

FIGURE 7.3: Some sample examples of a person at different positions, wearing different clothing. Variable poses and different body articulations can also be seen.

these images is about 1.7 meter. We use a curve fit for both height and width, and the results are shown in Figure 7.4. In this figure we have shown the height and width of the person against his radial distance from the centre of the image. As shown in Figure 7.4(b) the width of the person decreases as he moves aways from the centre whereas the height of the person increases upto a radial distance of about 300 pixels which is roughly the first half radius of the image: this is shown in Figure 7.4(a). The reason is that just below the camera the lower body of the person is obscured due to looking at it from an overhead view. The person is more visible as he moves further away from the centre of the camera. After the centre point of the radial distance, the height of the person reduces gradually due to radial distortion. Overall these effects can also be seen in Figure 7.3. The central region is the special case when the person looks more circular; therefore for that region upto a radius of 80 pixels we use a square box of 120 x 120 pixels. This size can accommodate all possible rotational movements of the person. The equations for selecting bounding box height and width are shown in Equations 7.1 and 7.2. In these equations r is the radial distance from the centre of the image to the centre of the person's position in the image. Using these equations our algorithm decides the width and height of the person at different positions of the image. When scanning an image using these equation it returns a total of 16386 positions or centroids of detection windows per image.

$$BB_H = 1.22 \times 10^{-3} r^2 + 0.68r + 81.6 \tag{7.1}$$

$$BB_W = 3.05 \times 10^{-4} r^2 + 0.072r + 107.2 \tag{7.2}$$



(a) Height of the Person

(b) Width of the Person

FIGURE 7.4: Height and width of the person at different positions in the scene using curve fit. The horizontal axis shows the radial distance from the centre of the image.

7.3 Experiments and Results

We conducted the following different types of experiments. Each of the positive image used in this experiment contains only one person. For clustering we use the default parameters as discussed in the previous chapter. The cluster size with greater than 10 members is considered as a person in the person-free images. In the images containing people, we assume that biggest cluster with a cluster members greater than 10 corresponds to a person. In our all experiments, we scale all the variable size bounding boxes calculated by the Equations 7.1 and 7.2 respectively, to a constant size of 64×96 pixels to generate a fixed sized features vector for the SVM.

7.3.1 Selection of a Pretrained SCOVIS Model

The main purpose of this experiment is to select an appropriate pretrained SCOVIS model for detecting people using test images from the SOTON dataset. We use 70 test images from the SOTON dataset where the person is at different positions and poses in the scene. We use five previously trained different SCOVIS models, which were used at five outer radial regions of the image. Each model is used to test these 70 images containing people. We then manually count the detected positions around the person in each of these images and calculate the average detected positions per person for each of

these SCOVIS models. These results are shown in Table: 7.2. In this table the second column represents the radial region of the SCOVIS image. The average detections per person in 70 images are shown in the third column. These numeric results show that the model trained with region 155-190 pixels gives maximum detections per person.

TABLE 7.2: Selection of a pre-trained SCOVIS model. These results are produced by measuring the average detections of 70 test images containing one person at different positions using the SOTON dataset.

Sr. No	Radius range	Detections
1	91-130	39
2	131-155	47
3	155-190	57
4	190-230	36
5	230 onward	25

We also observe the number of detections and false detections per image using visual inspection. In our visual analysis we found that the model trained with the radial region 155-190 pixels produced a small number of false detections per image along with compact clusters of detections near the positions of persons compared to all others. The overall performance of all models was also good except for the last one. We have already discussed the reason in Chapter 5; that the outer models have a lot of variation in poses and occlusion of people with rods and racks which makes it less discriminative than other models. So we decided to use the model trained with the radial region 155-190 pixels of the SCOVIS image in the rest of our experiments.

The conclusion of this experiment is that we use five different pre-trained SCOVIS models to test the existence of a person using the SOTON dataset. We manually inspected the detections results per image using each of these models and chose the one which gave the best results.

7.3.2 Reducing False Positives per Image

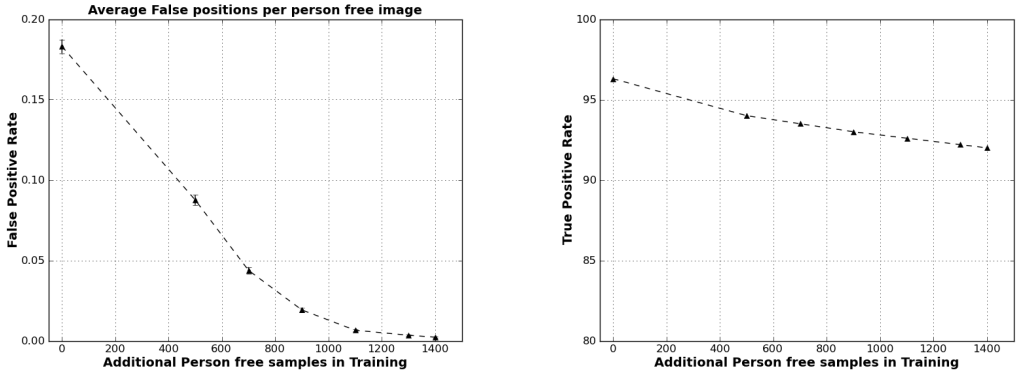
In our previous experiment we chose a pre-trained SCOVIS model. The selection of this model was made by inspecting and counting the number of detections near the person in the image. To do that we just use the images with people and found that the said model produce a cluster of detections around the person in the image along with randomly distributed spurious detections or false positives. To analyse the exact behaviour of these spurious detections, we chose 1000 person-free test images from the SOTON dataset. The chosen model was used to scan those positions of the test image which are 80 pixels away from the image centre and for the central region we use pre-trained SCOVIS central model. The result was about 3000 false detected positions out of 16386 scanned positions per image. We then provide some additional, randomly chosen 500 background samples from 100 person-free training images from the SOTON

dataset. A second training model is then built by appending these extra background samples to the initial training model. We scan 100 person-free training images using this second training model and the result was, on average, about 1400 false detections per image. From these false detections we randomly chose 900 samples and retrained the remaining models by providing some additional background training with a step size of 200 samples. This resulted in seven training models (including the initial one). Each of these models was then used to test 1000 person-free images. For the central model we use only 130 background samples and retrain the model. The number of samples used in these two models are shown in Table: 7.3.

TABLE 7.3: Number of positive and negative samples in the training models.

Radius	SCOVIS +ve	SCOVIS -ve	SOTON +ve	SOTON -ve
0-80	973	973	0	130
> 80	2712	2712	0	1400

Figure 7.5 shows the effect of extra background training. It can be seen from Figure 7.5(a) that additional training reduces FPR to almost 0%. To evaluate the true positive rate we use 7071 manually labelled test samples containing people from the Southampton dataset and tested using SCOVIS trained model with extra background training as discussed earlier. Doing this true positive rate reduces from 95% to 92% as shown in Figure 7.5(b) As we are using all the positive samples from only SCOVIS dataset, We, therefore, in the rest of the experiments, refer this model as the SCOVIS trained model.



(a) Effect on false positive rate.

(b) Effect on true positive rate.

FIGURE 7.5: Effect of extra background training on performance. Additional background training reduced false positive rate to almost 0%. The true positive rate reduces from 96% to 92%.

The conclusion of this experiment is that when we test 100 person-free images of the SOTON dataset using the pre-trained SCOVIS model, we achieved an average of 3000 false detections per image or 18% FPR per image. We then train the model by appending a total of 1400 background samples from the SOTON person-free images. Providing this additional training reduces the false positive rate close to 0%.

7.3.3 SCOVIS trained model with SOTON testset

The main purpose of this experiment is to check how well the SCOVIS-trained model performs when testing with the images from the SOTON dataset. We use the SCOVIS-trained model and test 7071 positive test images from the SOTON dataset. The result of detected positions are matched with ground truth positions; to do this we simply use disparity (distance between actual and detected positions in pixels). The result is shown in Figure 7.6. In the SOTON dataset the average width and height of the person are 90 and 140 pixels respectively. So we decided to set two thresholds; i.e, 40 pixels and 65 pixels shown in red and blue vertical lines, respectively, in Figure 7.6 Overall, 96% of detected positions are within a distance of 40 pixels from the actual positions while 98% of them are within a distance of 60 pixels.

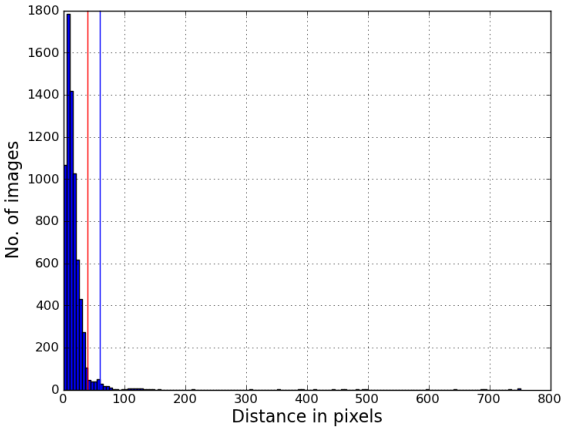


FIGURE 7.6: Disparity in pixels using all 7071 test images.

The result of each individual sequence within 40 pixels disparity is shown in Table: 7.4. The True Detection Rate (TDR) and False Detections Rate (FDR) are shown as percentages. Overall average TDR= 96% and FDR= 4% are achieved. In the rest of experiments we use the disparity of 40 pixels.

The conclusion of the experiment is that, using our algorithm to train a model with one set of data is robust enough to detect people from images taken from a completely different environment and camera settings with an accuracy of 96%.

TABLE 7.4: Result of each individual sequence. TDR and FDR are shown as percentages.

Seq	1	2	3	4	5	6	7	8	9	10	11	12	13	14	Avg
TDR	97.6	98.3	94.6	95	93.2	94	93.8	96.8	96.1	93	98.8	98	95	95	96
FDR	2.4	1.7	4.4	5	6.8	6	6.2	3.2	3.9	7	1.2	2	5	5	4

7.3.4 Person Detection using SOTON Training and Testset

In this experiment we use only the SOTON data set for both training and testing. We reserve sequence No 7 and 11 respectively, of Table 7.4 as a test set and and all the remaining ones as a training set. These two sequences are the worst and best long sequences as tested in the previous section. From the training set we randomly chose 2712 manually labeled samples of people and the same number of randomly chosen person-free samples from background images. The reason for choosing this number is to make it consistent with the SCOVIS model which used the same number of samples. A classifier is then built using this training set. We test the images of test sequences using this classifier. The result was a 99% detection rate for sequence 7 and a 100% detection rate for sequence 12.

7.3.5 Person Classification and Automatic Labelling

We devise three different experiments of person classification. In the first experiment we use all manually labelled 2712 samples of people and the same number of randomly chosen samples of the background using the SCOVIS dataset. 2-fold cross-validation testing is then performed and the results are shown in Table: 7.5. A TPRs of 95% was achieved, while the FPRs was 3%.

In the second experiment we use the SCOVIS trained model to detect people from positive images from the SOTON dataset (except for sequences 7 and 11). We use a clustering algorithm to select the biggest cluster per image and assume that the detected position of the cluster as a centroid of the person. This is referred to as Automatic Labelling from this point on in the thesis. From these automatic labelled positions, we chose 2712 as positive samples of people. The same number of negative samples are chosen from person-free images and a 2-fold cross-validation testing is performed. The results were 97.6% TPR with a 1.5% FPR as shown in the second row of Table: 7.5.

In the third experiment, we chose all positive samples (except samples of sequences 7 and 11 respectively) and randomly chose 2712 positive samples and the same number of negative samples from the person-free images from SOTON dataset. A 2-fold cross-validation testing is then performed, and the results are shown in Table: 7.5. We achieve a TPRs of 99% with a FPR of 0.6%.

If we compare the results of SOTON trained model with automatic labelling with the results produced by the SOTON manually labelled model, we can see a decrease of about 1% in the TPR with a slight increase in FPR. This small degradation of performance is because in automatic labelling we also used some false detections, which were treated as samples of people. Overall, the SOTON dataset produced better results than the SCOVIS dataset. The reason could be because of good quality, uncompressed images.

In this section our main subject of discussion was automatic labelling. We have shown experimentally that automatic labeling gives quite satisfactory results when used for person classification.

TABLE 7.5: Person Classification results using three different trained models.

	Fold-1		Fold-2		Total	
Trained Model	TPR	FPR	TPR	FPR	TPR	FPR
SCOVIS trained	95%	3%	95%	3%	95%	3%
SOTON trained with automatic labelling	97.6%	1.4%	97.4%	1.5%	97.5%	1.45%
SOTON trained	99.3%	0.6%	99.2%	0.6%	99.2%	0.6%

7.3.6 Person Detection and Automatic Labelling

In the previous experiment we discuss person classification using automatic labelling. In this experiment we extend the work of Person Classification to Person Detection using the SOTON dataset. We chose the trained model with automatic labelling as shown in the second row of Table: 7.5. This initially trained SOTON model with automatic labelling was tested using 10 person-free training images. The false detections from these person-free training images are used to provide extra training to the initial model. We append 180 additional background samples to retrain the model. This retrained model is then used to test two sequences; i.e, sequence 7 and sequence 11 with a total of 637+783= 1420 images. The result was a TDR of 97.17% for sequence 7. For sequence 11 a detection rate of 99.36% was achieved. These results shows that our model trained with automatic labelling can detect people with an accuracy of 97.5%.

7.3.7 Overall Comparison of Detection Results

We discuss here the results of two test sequences (7 and 11) of the SOTON dataset using our three training models as discussed in Section 7.3.5. The results are shown in Table 7.6.

TABLE 7.6: Person Detection results: Results of detecting people in images using three different trained models.

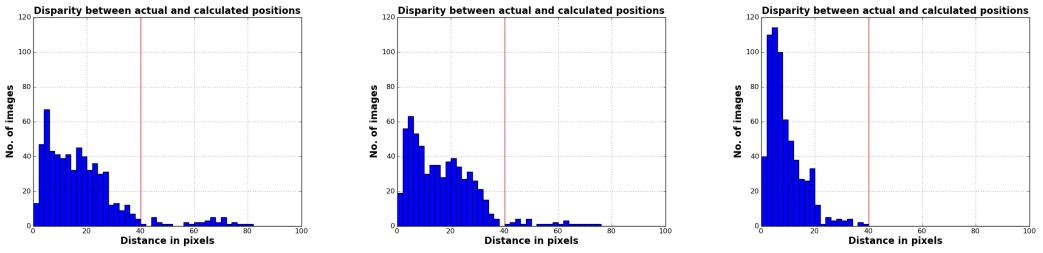
	Seq-7		Seq-11		Average	
Trained Model	TDR	FDR	TDR	FDR	TDR	FDR
SCOVIS trained	93.8%	6.2%	98.8%	1.2%	96.3%	3.7%
SOTON trained with automatic labelling	97.2%	2.8%	99.4%	0.6%	98.3%	1.7%
SOTON trained	99%	0.3%	100%	0%	99.5%	0.15%

In this table the first column describes the trained model used for detecting people in test images. In the next two pairs of columns we have shown the results of two test sequences. The average results of these test sequences are shown in the last pair. In

the first and third rows of the table the results are shown using SCOVIS and SOTON trained models. The second two rows of the table show the results of using automatic labelling with SOTON data. The best detection results are produced when we use the SOTON model trained with manual labelled images. We achieve an overall detection rate of 99.5% with only 0.15% false detections per image.

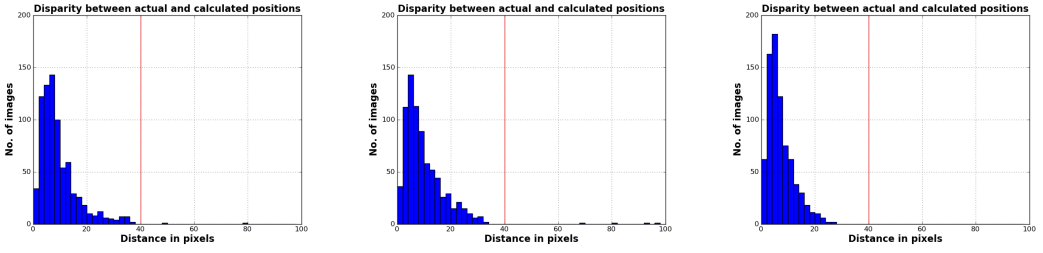
The generalisation performance can be observed by comparing the SOTON-trained model results with those of the SCOVIS trained model. The detections results of sequence 11 using both models are very close with only a difference of only 1.2% as compared to sequence 7 where a reduction of 5% can be seen in detection rate using the SCOVIS-trained model. Overall results show that the SOTON-trained model has a 4% better detection rate than the SCOVIS one. We believe that using the SCOVIS-model to test completely different images of the SOTON dataset with an overall 96.3% accuracy is a very significant achievement in results. This shows that our algorithm has an excellent generalisation capability.

The disparity measurement of these two sequences using two different models can be seen in Figure 7.7 and Figure 7.8. The red line shows the chosen threshold of 40 pixels. In the case of the SOTON manual labelled model, the peak in both graphs shows that detected positions are very close to the ground truths. The majority of the detected positions are only 20 pixels away from the actual positions.



(a) Using SCOVIS manual labelled (b) Using SOTON auto labelled (c) Using SOTON manual labelled

FIGURE 7.7: Disparity using test sequence 7 (637 images).



(a) Using SCOVIS manual labelled (b) Using SOTON auto labelled (c) Using SOTON manual labelled

FIGURE 7.8: Disparity using test sequence 11 (783 images).

Comparing these results with the SCOVIS results, the distributions of positions are a bit wider in the graph, but still most of the population (detected positions) are within

a 40 pixel distance from the ground truth.

The second main finding is about the automatic labelling ability of our algorithm. The results shown in the last two rows of Table 7.6 are about the model trained with manually and automatic labelled positions of people respectively using images of the same SOTON dataset. The results of both models are very similar and, on average, there is about 1% improvement in detection rate using manual labelling. Overall, using automatic labelling with a 98.3% detection rate is a very significant result; this result not only shows how excellent the detection performance of our algorithm is, but also tell us about the confidence of using automatic labelling.

7.3.8 Change in Person Size and Camera Position

In the previous experiments we have shown that our proposed algorithm produces significant results of person detection and also has an excellent generalisation capability. In this experiment we address the effect of taking different height and width of a person than the default values of height and width used in our algorithm. In our all previous experiments we used Equations 7.1 and 7.2, to measure the height and width of the bounding box. The real height of the person in the SOTON images is 5.6 feet.

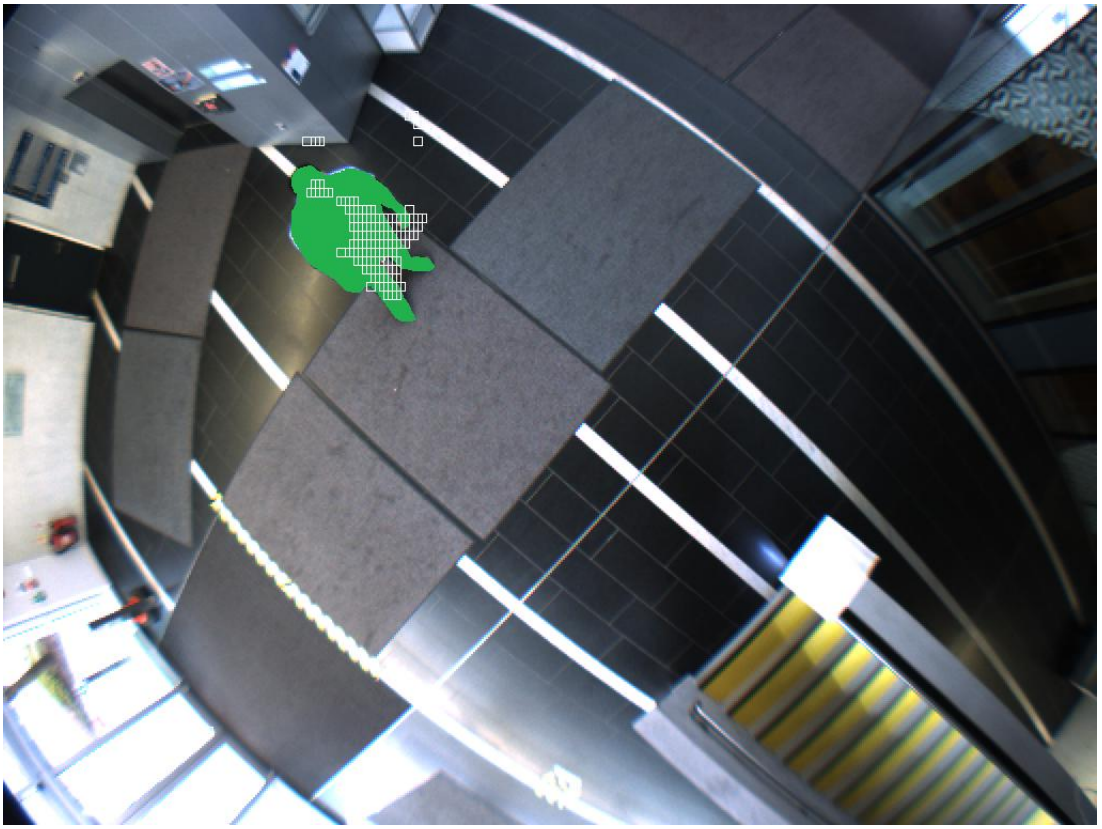


FIGURE 7.9: A detected person with different height. This image is taken with a small change in camera position. The identity of the subject has been hidden.

The author used some of the images of his friend with his permission to see the effect of change in height and a little change in camera alignment. The height of the person was 6 feet with heavier body structure than the shape of person used in the SOTON dataset. When we tested these images using the SCOVIS-trained model, we achieved excellent results with the correctly detected person in all of these images. The result of a sample image is shown in Figure 7.9. Here, due to privacy and data protection issues, only the outline of the person is shown in green colour. A cluster of detected positions are also shown as small white boxes. Furthermore, even though there is slight change in the field of view of the camera in this image, the person has been detected without any false detection. These results shows that our proposed algorithm is invariant to body shape to a certain extent, and also when there is a small change in camera position.

7.4 SOTON Trained model with SCOVIS Test set

This experiment was conducted to test the performance of person classification system using SOTON trained model tested on SCOVIS images. To do this we use a pre-trained SOTON model using 2712 positive and same number of randomly chosen negative samples from person free images from SOTON database. The test set consist of same number of positive and negative samples from SCOVIS dataset. Doing this a TPR of 22% is achieved with a FPR of 0.5%. There are many possible explanations for this and two of which are:-

- The SCOVIS images contain occluded people whereas Southampton dataset does not. This is born out by inspecting the true positive rate with respect to occlusion. 0% occluded samples in the test images has a TPR of 36% where the remainder had on average TPR of 17%.
- In the SCOVIS environment people perform different activities by carrying, handling and welding different parts of the car and interacts with racks and heavy machinery. During these activities there is great availability in the poses of people. The trained model with SOTON images only has a single person walking.

Discussion

The main focus of this chapter is the generalisation performance of our algorithm. We have shown by experiments that our algorithm has an excellent generalisation capability with only limited intervention of calculating bounding box size and some randomly chosen background images. To do this we use a pre-trained model of the SCOVIS dataset and test images from the SOTON dataset, and achieved very attractive results. Another finding is that our algorithm has the capability to carry out automatic labelling

of people when scanning test images containing people. We use two models trained with the SOTON dataset using manual and automatic labelling, respectively, and we attained excellent detection results for both of these models when tested with positive images of the SOTON dataset. There was a decrease of only 1.7 % in detection rate when we use automatic labelling. We also have shown that our proposed algorithm is invariant to a certain extent to the size of a person and slight changes in the position of the camera. Finally, it is noteworthy that we one use one classifier in the in the whole outer region of the image and the classifier did not show any resistance which was observed in case of the SCOVIS images.

Chapter 8

Conclusions and Future Directions

People detection in images has become one of the most active area research area in computer vision because of its numerous potential applications in different areas. Although there are currently a number of robust methods in use for detecting people in images from a horizontal view, these methods tend to fail in extreme environments, particularly when people are highly occluded and assume a variety of poses. In such environments even manual monitoring of people becomes extremely difficult. To overcome this, we explore a new dimension of research in which person detection is done using an overhead view. This thesis has described a complete framework for the problem of detecting people in images using an overhead view.

8.1 Conclusion

The main contributions of the thesis are as follow:

- **A Robust Person Detector for Overhead View**

  We explore a new dimension of research in which the person is viewed from an overhead camera with a wide angle lens. Due to perspective change a person can have different orientations and sizes relative to the position of the camera. We exploit this property of the overhead camera and develop a novel algorithm which uses the variable size bounding boxes with different orientations on the basis of the person's position in the image and radial distance from the centre of the image. To achieve this we use different geometric transformations in our algorithm which orients the bounding box according to the orientation of the person in the image. We compare the results of our proposed algorithm (rHOG) with those of

the standard Histogram of Oriented Gradient (sHOG) algorithm which uses fixed orientation and fixed square sized bounding boxes. In both of these algorithms we use our overhead dataset for training and testing.

In the Person Classification results, a TPR of 98% with FPR of 2% was achieved using our proposed algorithm, while for the sHOG algorithm the result was a TPR of 96% with a FPR of 7%. Although the classification results were encouraging for both of these algorithms, when we used these classifiers to detect people in images the results were radically different. Our algorithm has shown a significant improvement with a detection rate of 95% compared to a detection rate of 59% using the sHOG algorithm. Another important finding is that with this detection rate, the sHOG algorithm produced 9 spurious or falsely detected people per image whereas our proposed algorithm produced only one falsely detected person in 50 images.

- **A Simple Person-tracking-by-detection Algorithm**

  Our second contribution is to develop a simple person-tracking-by-detection algorithm to demonstrate how good our person detection algorithm works. To reduce the exhaustive search of all positions of the image, we incorporate a simple motion detection framework in our tracking algorithm. These motions blobs are produced using some statistical analysis and a connected component analysis after consecutive frame differencing. A search for the existence of a person is made only on detected blob positions. The expected position of the person in the next frame is estimated using the history of the already detected person. We compare results of our proposed tracking algorithm with the existing four standard tracking algorithms of OpenCV using five different sequences of a total of 2172 frames. The results have clearly shown that our algorithm outperforms others; can track people with an accuracy of 99% while the best results produced by the Mean Shift algorithm have an accuracy of only 48%. Our proposed tracking algorithm has the additional advantage of handling big missing data gaps and occlusion, and detecting a stationary person in the image.

- **Generalising the Person Detector**

  Our third contribution relates to the generalisation frame work of the person detector. We have shown experimentally that training with one set of overhead data our algorithm can work perfectly in another completely different overhead dataset with a little manual intervention of measuring the bounding box size using a few images containing people from the new dataset. When we use the SOTON-trained model (with manual labelling of people) to test 1420 test images containing people from the SOTON dataset, we achieve a detection rate of 99.5%. When we use a SCOVIS-trained model to test 7071 test images containing people from the

SOTON dataset; the result was 96%. Finally, when we use the SOTON-trained model (with automatic labelling of people) to test 1420 test images containing people from the SOTON dataset We achieve a detection rate of 98.3% .

We have demonstrated that our proposed algorithm not only locates the person in the given test image but also has the potential of automatic labelling of the centroid of the person. This suggests that the said generalised frame work can be applicable to any overhead dataset without using any manual annotation of people in images. We also have shown that our proposed algorithm has a certain amount of invariancy regarding body size and camera alignment.

Finally, a noteworthy factor is that we use just one classifier in the outer radial bound of the image compared to the SCOVIS images where a single classifier was not sufficient. Perhaps this indicates the challenging nature of the SCOVIS dataset.

Following a self-critical analysis of our research, we observe the following points:

- The first self-criticism is that our algorithm has only been evaluated on two overhead datasets. Due to data protection and privacy issues, we were unable to record more data and in the SOTON dataset there is a maximum of one person per image. The generalisation performance has not been checked in case of the presence of multiple persons in images.
- Secondly, although there were many other advanced algorithms available, we elected to use the HOG algorithm as a baseline to sort out the problem of person detection in the overhead view. These algorithms mostly discussed the improvement in classification rate. We have shown experimentally that there is a very weak co-relation between the classification results and detection results; it is not necessary that good classification results guarantee good detections results as well. The classification results generated by the sHOG and rHOG algorithms using our database were similar, but when we tested images with these classifiers the results differed significantly. We conclude that among multiple advanced algorithms with algorithmic and time complexity, by selecting the HOG algorithm, we achieve upto 99% detection results. This confirms that our decision to choose the HOG algorithm was a good one.
- Finally, although our tracking algorithm shows a good control of person detection in the case of missing frames, an occluded person with a cluttered background, a dynamic change in environment and a stationary person, it still lacks a property whereby it can identify people when they pass each other in the frame, or appear at completely different positions after a long time lag. Recent work in gait-biometrics has shown that the top-view has latent potential for identification [99]. This could be implemented and might alleviate some of the problems mentioned above.

8.2 Future Directions

Following are the possible future directions to extend this research work:

- **More Investigation about Generalisation**

  We have shown experimently that our proposed algorithm has an excellent generalisation performance, but due to data protection and privacy issues we use only two datasets in our experiments. Each of these dataset has some limitations. For example, the images of the SCOVIS dataset are highly compressed and very noisy. Conversely. Although our newly recorded SOTON dataset has good quality images, it only contains one person per image. Therefore, further evaluation is needed with some more newly recorded, good quality datasets which have multiple persons per image. The opportunity to record a new dataset in Pakistan may arise.

- **Evaluations of other Object Classes**

  We have limited our discussion to detection of people only. In our proposed algorithm we utilise the property of an overhead camera to detect people. This property is not restricted to the person only, and is applicable to any object viewed from an overhead camera with a wide angled lens.

  Dalal and trigg [29] used the HOG algorithm for person detection, and also tested 10 others different object classes using the same algorithm; they achieved comparable results with the conclusion that their algorithm works equally on other object classes.

  We did not apply our proposed algorithm on other object classes but we see no reason why it should not work on other object classes. Therefore, some new overhead datasets for different object classes other than people are also needed to evaluate the performance of our algorithm.

- **Use of Asymmetric Overhead View**

  Finally, we can explore the overhead view further, by switching from a symmetric to an asymmetric view. In a symmetric view the camera is aligned at the right angle from the ground, but in an asymmetric view this angle can be changed from the vertical right-angle. One such example of an asymmetric view is shown in Figure 8.1. This image is taken from another dataset of the SCOVIS project where an industrial environment is shown. Three people are shown as white ellipses. We can use some newly recorded datasets of people and possibly some other objects as well. Our proposed algorithm can be used with some modifications in the geometric transformation module to tackle asymmetric views.

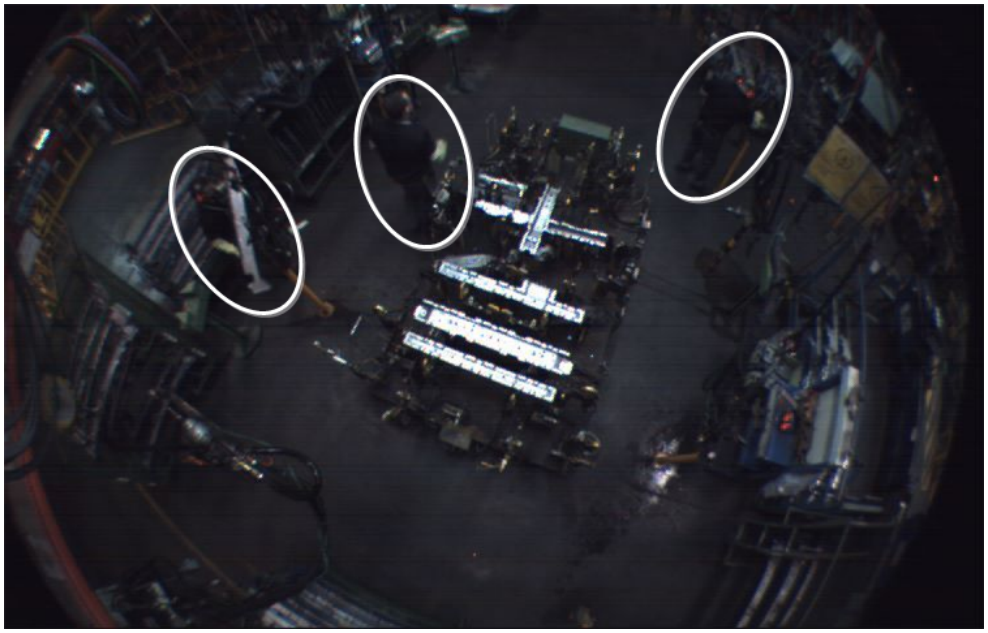


FIGURE 8.1: An example of asymmetric overhead view. Three people are shown as white ellipses.

Appendix A

rHOG results

		Fold1		Fold2			Average	
Radius in pixels	Window size	TPR	FPR	TPR	FPR		TPR	FPR
0-30	64x64	297/304= 0.98	0/305 = 0.0	296/305= 0.97	1/305 = 0.003		**0.975**	**0.0015**
(609 images)	72x72	297/304= 0.98	0/305 = 0.0	292/305= 0.96	0/305= 0.0		**0.97**	**0**
	80x80	299/304= 0.98	0/305= 0.0	298/305= 0.98	0/305= 0.0		**0.98**	**0**
	88x88	298/304= 0.98	0/305= 0.0	299/305= 0.98	0/305= 0.0		**0.98**	**0**
31-90	64x72	782/812= 0.96	11/812= 0.014	771/811= 0.95	4/812= 0.005		**0.955**	**0.0165**
(1623 images)	64x80	787 / 811= 0.97	9/812= 0.011	781/812= 0.96	2/812= 0.002		**0.965**	**0.0065**
	64x88	784/811= 0.97	4/812= 0.005	783/812= 0.97	1/812= 0.001		**0.97**	**0.003**
	72x88	787/811= 0.97	1/812= 0.001	785/812= 0.97	5/812= 0.006		**0.97**	**0.0035**
	64x96	787/811= 0.97	5/812= 0.006	788/812= 0.97	4/812= 0.005		**0.97**	**0.0055**
	64x104	787/811= 0.97	1/812= 0.001	784/812= 0.97	2/812= 0.002		**0.97**	**0.0015**
	64x112	786/811= 0.97	5/812= 0.006	787/811= 0.97	3/812= 0.004		**0.97**	**0.005**
	64x120	788/812= 0.97	2/812= 0.002	786/812= 0.97	2/812= 0.002		**0.97**	**0.0025**
	64x128	788/812= 0.97	4/812= 0.005	788/812=0.97	3/812=0.004		**0.97**	**0.0045**
91-130	64x80	885/920= 0.96	18/921= 0.02	894/920= 0.97	16/921= 0.017		**0.965**	**0.0185**
(1841 images)	64x88	894/920= 0.97	16/921= 0.017	897/921= 0.97	13/921= 0.014		**0.97**	**0.031**
	64x96	894/920= 0.97	12/921= 0.013	899/921= 0.98	9/921= 0.01		**0.975**	**0.018**
	64x104	905/920= 0.984	15/920= 0.02	899/921= 0.98	11/921= 0.012		**0.98**	**0.016**
	64x112	901/920= 0.98	11/921= 0.012	899/921= 0.98	8/921= 0.009		**0.98**	**0.01**
	64x120	906/920=0.985	7/921= 0.008	905/921= 0.983	9/921= 0.01		**0.984**	**0.009**
	64x128	902/920= 0.98	5/921= 0.005	903/921= 0.98	3/921= 0.003		**0.98**	**0.004**
	72x120	899/920= 0.98	2/921= 0.002	900/921= 0.98	5/921= 0.005		**0.98**	**0.004**
131-155	64x96	739/773= 0.956	19/773= 0.025	739/773= 0.956	14/773= 0.018		**0.96**	**0.024**
(1546 images)	64x104	747/773= 0.966	19/773= 0.025	744/773= 0.962	10/773= 0.013		**0.96**	**0.019**
	64x112	742/773=0.96	16/773= 0.021	746/773= 0.965	9/773= 0.012		**0.96**	**0.017**
	64x120	750/773= 0.97	15/773= 0.02	751/773= 0.972	11/773= 0.014		**0.97**	**0.017**
	64x128	748/773=0.968	8/773= 0.01	751/773= 0.972	8/773= 0.01		**0.97**	**0.01**
	72x128	744 / 773= 0.96	8/773= 0.01	751 / 773= 0.972	7/773= 0.009		**0.965**	**0.01**
(3286 images)								
155-190	64x96	1560 / 1641 = 0.95	50 / 1643= 0.03	1564 / 1642= 0.95	56 / 1643= 0.03		**0.95**	**0.03**
	72x96	1565 / 1641 = 0.95	50/1643= 0.03	1567/1642= 0.95	74/1643= 0.05		**0.95**	**0.04**
(1999 images)								
190-230	64x96	938 / 999 = 0.94	30 / 1000= 0.03	949 / 1000= 0.95	24 / 1000= 0.024		**0.945**	**0.03**
	72x96	939 / 999= 0.94	25 / 1000= 0.03	953 / 1000= 0.95	27 / 1000= 0.03		**0.945**	**0.026**
230-	64x96	1009/1053= 0.96	28/1053= 0.027	1020/1053= 0.97	42/1053= 0.04		**0.965**	**0.034**
(2106 images)	72x96	1016/1047= 0.97	36/1053= 0.034	1026/1053= 0.98	42/1053= 0.04		**0.975**	**0.037**
	72x88	1022/1053= 0.97	34/1053= 0.032	1023/1053= 0.97	38/1053= 0.036		**0.97**	**0.034**

FIGURE A.1: Person Classification results using rHOG algorithm.

Appendix B

A tool for extracting time from images

A sample image is shown in Figure B.1 where the region of image containing time information is highlighted using a white ellipse. As shown in the figure, this region has information about the camera number and time with date, month and year. We use last eight digits in which the time is in (hh,mm,ss) format. For each of these digits we cropped a 7×10 pixels patch of the image. Some example of these cropped regions are shown in Figure B.2.

FIGURE B.1: Image region showing time.

For each of these cropped image patches a binary image is created by applying a threshold. Thus for a 7×10 pixels binary image, a feature vector with 70 features is obtained.

Overall we use ten image patches for the decimal numbers 0-9 as a training data. These features are trained with LibSVM and a multi-class classifier is built. We tested the rest of images using this classifier to extract time information for each image. The result was a 100% correct classification of digits. These results were verify manually using 100 randomly chosen images.

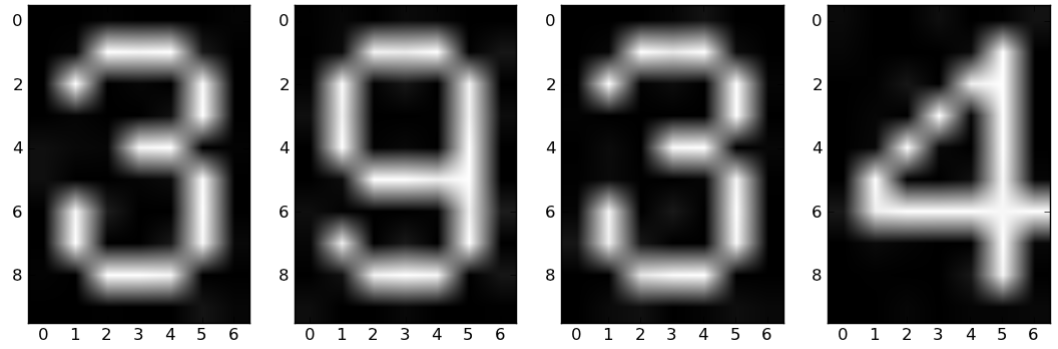


FIGURE B.2: Cropping out image regions containing time: Each of these digits is represented by a 7×10 pixels.

Appendix C

The OpenCV Tracking Algorithms

C.1 Mean-Shift with Foreground Weight Method

The FG/BG detection module of the OpenCV blob-tracking system is used to perform foreground/ background segmentation processes. There are different choices of background modelling available including adaptive background mixture models [103]. The OpenCV tracking algorithm uses [70] as a default method for foreground object segmentation and internally this method uses a change detection algorithm [94]. The result of this method is a foreground/background mask. In the Mean-Shift with Foreground Weight (MSFG) method the pixels of the foreground object are weighted during calculation.

C.2 Mean-Shift with Particle Filter Method

Particle Filtering [35] is a general Monte Carlo (sampling) method for performing inference in state-space models where the state of a system evolves over time and where information about the state is obtained via noisy measurements taken at each time step [85]. A particle filter is concerned with the problem of tracking a single object or multiple objects. A set of weighted particles is used to approximate the posterior distribution of a tracker. A particle filter weights particles based on a like-hood score and then propagates these particles using a motion model.

In the Mean-Shift with Particle Filter (MSPF) algorithm, 200 random particles are allocated for each tracked object or blob. Each particle is associated with a weight that represents the probability of that particle being sampled from the probability density function and is updated in every frame. These weights are functions of Bhattacharyya

coefficients and are calculated between the current and the candidate histogram, respectively. The weighted sum yields the new prediction for the position and size for the blob in the next frame.

C.3 Connected Component Method

The Connected Component (CC) Tracker uses the contours and shapes to represent blobs. The source gray scale image is converted into a binary image after applying some thresholding. This binary image is then analysed using the connected component analysis method. This analysis finds the region of connected pixels which have the same values using 4-neighbours or 8-neighbours from the source pixel. The first connected region is labelled as 1 and the process continues until the search for all pixels of an image is completed. One such example has already been discussed in Section 6.1.2 and is shown in Figure 6.3(e), where three separate regions are formed using the connected component method.

C.4 Connected Component, Mean-Shift and Particle Filter Method

This algorithm uses the combination of connected component, Mean-Shift and particle filter methods, as discussed above for blob tracking.

Bibliography

[1] Artificial Vision Technology. http://www.mobileye.com/technology/. [Online; accessed 23-Sep-2013].

[2] OpenCV Blob Tracking Modules. http://opencvlibrary.svn.sourceforge.net/viewvc/opencvlibrary/trunk/opencv/doc/vidsurv/Blob_Tracking_Modules.doc. [Online; accessed 17-Sep-2013].

[3] Blob Tracking - Video Surveillance Demo. http://experienceopencv.blogspot.co.uk/2011/03/blob-tracking-video-surveillance-demo.html, 01-03-2011. [Online; accessed 17-Sep-2013].

[4] The OpenCV Video Surveillance / Blob Tracker Facility. http://opencv.willowgarage.com/wiki/VideoSurveillance, 2009. [Online; accessed 17-Sep-2013].

[5] TUD Pedestrians dataset. http://www.d2.mpi-inf.mpg.de/tud-brussels, 2009. [Online; accessed 17-Sep-2013].

[6] Timo Ahonen, Abdenour Hadid, and Matti Pietikainen. Face description with local binary patterns: Application to face recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(12):2037–2041, 2006.

[7] Hrishikesh Aradhye, Martin Fischler, Robert Bolles, and Gregory Myers. Headprint–person reacquisition using visual features of hair in overhead surveillance video. In *Audio-and Video-Based Biometric Person Authentication*, pages 879–890. Springer, 2005.

[8] Banafshe Arbab-Zavar, Imed Bouchrika, John N Carter, and Mark S Nixon. On supervised human activity analysis for structured environments. In *Advances in Visual Computing*, pages 625–634. Springer, 2010.

[9] Banafshe Arbab-Zavar, John N Carter, and Mark S Nixon. On hierarchical modelling of motion for workflow analysis from overhead view. *Machine Vision and Applications*, pages 1–15, 2013.

[10] Hou Beiping and Zhu Wen. Fast human detection using motion detection and histogram of oriented gradients. *Journal of Computers*, 6(8):1597–1604, 2011.

[11] Kristin P Bennett and Colin Campbell. Support vector machines: hype or hallelujah? *ACM SIGKDD Explorations Newsletter*, 2(2):1–13, 2000.

[12] Massimo Bertozzi, Alberto Broggi, Mike Del Rose, Mirko Felisa, Alain Rakotomamonjy, and Frédéric Suard. A pedestrian detector using histograms of oriented gradients and a support vector machine classifier. In *Intelligent Transportation Systems Conference, 2007. ITSC 2007. IEEE*, pages 143–148. IEEE, 2007.

[13] Gary Bradski and Adrian Kaehler. *Learning OpenCV: Computer vision with the OpenCV library*. O'Reilly Media, Incorporated, 2008.

[14] Heinz Breu, Joseph Gil, David Kirkpatrick, and Michael Werman. Linear time euclidean distance transform algorithms. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 17(5):529–533, 1995.

[15] Joshua Candamo, Matthew Shreve, Dmitry B Goldgof, Deborah B Sapper, and Rangachar Kasturi. Understanding transit scenes: a survey on human behavior-recognition algorithms. *Intelligent Transportation Systems, IEEE Transactions on*, 11(1):206–224, 2010.

[16] Trista P Chen, Horst Haussecker, Alexander Bovyrin, Roman Belenov, Konstantin Rodyushkin, Alexander Kuranov, and Victor Eruhimov. Computer vision workload analysis: Case study of video surveillance systems. *Intel Technology Journal*, 9(2):109–118, 2005.

[17] Yizong Cheng. Mean shift, mode seeking, and clustering. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 17(8):790–799, 1995.

[18] Cheng-Hsiung Chuang, Shih-Shinh Huang, Li-Chen Fu, and Pei-Yung Hsiao. Monocular multi-human detection using augmented histograms of oriented gradients. In *Pattern Recognition, 2008. ICPR 2008. 19th International Conference on*, pages 1–4. IEEE, 2008.

[19] Charles K Chui. *An introduction to wavelets*. Academic Press Professional, Inc., 1992.

[20] Ira Cohen, Ashutosh Garg, and Thomas S Huang. Vision-based overhead view person recognition. In *Pattern Recognition, 2000. Proceedings. 15th International Conference on*, volume 1, pages 1119–1124. IEEE, 2000.

[21] R. Collins. Daimler PedestrianDataset. http://www.gavrila.net/Datasets/Daimler_Pedestrian_Benchmark_D/daimler_pedestrian_benchmark_d.html. [Online; accessed 30-january-2014].

[22] R. Collins. CAVIAR person dataset. http://homepages.inf.ed.ac.uk/rbf/CAVIARDATA1/, 2003. [Online; accessed 17-Sep-2013].

[23] R. Collins. Mean-shift Tracking. http://www.cse.psu.edu/~rcollins/CSE598G/introMeanShift_6pp.pdf, 2006. [Online; accessed 17-Sep-2013].

[24] Dorin Comaniciu and Peter Meer. Mean shift: A robust approach toward feature space analysis. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(5):603–619, 2002.

[25] Dorin Comaniciu, Visvanathan Ramesh, and Peter Meer. Kernel-based object tracking. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(5):564–577, 2003.

[26] Marco Cristani, R Raghavendra, Alessio Del Bue, and Vittorio Murino. Human behavior analysis in video surveillance: a social signal processing perspective. *Neurocomputing*, 100:86–97, 2013.

[27] Franklin C Crow. Summed-area tables for texture mapping. In *ACM SIGGRAPH Computer Graphics*, volume 18, pages 207–212. ACM, 1984.

[28] Navneet Dalal. *Finding people in images and videos*. PhD thesis, Institut National Polytechnique de Grenoble-INPG, 2006.

[29] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893. IEEE, 2005.

[30] Navneet Dalal, Bill Triggs, and Cordelia Schmid. Human detection using oriented histograms of flow and appearance. In *Computer Vision–ECCV 2006*, pages 428–441. Springer, 2006.

[31] Konstantinos G Derpanis. The harris corner detector. *York University*, 2004.

[32] Konstantinos G Derpanis. Integral image-based representations. *Department of Computer Science and Engineering York University Paper*, 1(2):1–6, 2007.

[33] Piotr Dollár, Christian Wojek, Bernt Schiele, and Pietro Perona. Pedestrian detection: A benchmark. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 304–311. IEEE, 2009.

[34] Piotr Dollar, Christian Wojek, Bernt Schiele, and Pietro Perona. Pedestrian detection: An evaluation of the state of the art. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(4):743–761, 2012.

[35] Arnaud Doucet and Adam M Johansen. A tutorial on particle filtering and smoothing: Fifteen years later. *Handbook of Nonlinear Filtering*, 12:656–704, 2009.

[36] Anastasios Doulamis, Dimitrios Kosmopoulos, Manolis Sardis, and Theodora Varvarigou. An architecture for a self configurable video supervision. In *Proceedings of the 1st ACM workshop on Analysis and retrieval of events/actions and workflows in video streams*, pages 97–104. ACM, 2008.

[37] Thanh Nguyen Duc, Philip Ogunbona, and Wanqing Li. Human detection based on weighted template matching. In *Multimedia and Expo, 2009. ICME 2009. IEEE International Conference on*, pages 634–637. IEEE, 2009.

[38] Markus Enzweiler and Dariu M Gavrila. Monocular pedestrian detection: Survey and experiments. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(12):2179–2195, 2009.

[39] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. Liblinear: A library for large linear classification. *The Journal of Machine Learning Research*, 9:1871–1874, 2008.

[40] Alessandra Fascioli, Rean Isabella Fedriga, and Stefano Ghidoni. Vision-based monitoring of pedestrian crossings. In *Image Analysis and Processing, 2007. ICIAP 2007. 14th International Conference on*, pages 566–574. IEEE, 2007.

[41] Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(9):1627–1645, 2010.

[42] Pedro F Felzenszwalb and Daniel P Huttenlocher. Efficient matching of pictorial structures. In *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, volume 2, pages 66–73. IEEE, 2000.

[43] Rogerio Feris. Part-Based and Hierarchical Models. http://rogerioferis.com/VisualRecognitionAndSearch/classes/class4.pdf, 14-02-2013. [Online; accessed 17-Sep-2013].

[44] David A Forsyth and Margaret M Fleck. Body plans. In *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*, pages 678–683. IEEE, 1997.

[45] Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *Computational learning theory*, pages 23–37. Springer, 1995.

[46] Keinosuke Fukunaga and Larry Hostetler. The estimation of the gradient of a density function, with applications in pattern recognition. *Information Theory, IEEE Transactions on*, 21(1):32–40, 1975.

[47] Tarak Gandhi and Mohan Manubhai Trivedi. Pedestrian protection systems: Issues, survey, and challenges. *Intelligent Transportation Systems, IEEE Transactions on*, 8(3):413–430, 2007.

[48] Jorge García, Alfredo Gardel, Ignacio Bravo, J LAZARO, Miguel Martinez, and David Rodriguez. Directional people counter based on heads tracking. 2013.

[49] Dariu M Gavrila. Pedestrian detection from a moving vehicle. In *Computer Vision ECCV 2000*, pages 37–49. Springer, 2000.

[50] Dariu M Gavrila. A bayesian, exemplar-based approach to hierarchical shape matching. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 29(8):1408–1421, 2007.

[51] Dariu M Gavrila and Stefan Munder. Multi-cue pedestrian detection and tracking from a moving vehicle. *International journal of computer vision*, 73(1):41–59, 2007.

[52] Dariu M Gavrila and Vasanth Philomin. Real-time object detection for smart vehicles. In *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, volume 1, pages 87–93. IEEE, 1999.

[53] DM Gavrila, Jan Giebel, and Stefan Munder. Vision-based pedestrian detection: The protector system. In *Intelligent Vehicles Symposium, 2004 IEEE*, pages 13–18. IEEE, 2004.

[54] David Gerónimo, Antonio López, Daniel Ponsa, and Angel D Sappa. Haar wavelets and edge orientation histograms for on–board pedestrian detection. In *Pattern Recognition and Image Analysis*, pages 418–425. Springer, 2007.

[55] David Geronimo, Antonio M Lopez, Angel D Sappa, and Thorsten Graf. Survey of pedestrian detection for advanced driver assistance systems. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(7):1239–1258, 2010.

[56] Amara Graps. An introduction to wavelets. *Computational Science & Engineering, IEEE*, 2(2):50–61, 1995.

[57] Steve R Gunn. Support vector machines for classification and regression. *ISIS technical report*, 14, 1998.

[58] Abdenour Hadid, Matti Pietikainen, and Timo Ahonen. A discriminative feature space for detecting and recognizing faces. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 2, pages II–797. IEEE, 2004.

[59] Ning He, Jiaheng Cao, and Lin Song. Scale space histogram of oriented gradients for human detection. In *Information Science and Engineering, 2008. ISISE'08. International Symposium on*, volume 2, pages 167–170. IEEE, 2008.

[60] Sibt Ul Hussain, William Triggs, et al. Feature sets and dimensionality reduction for visual object detection. In *British Machine Vision Conference*, 2010.

[61] Mohamed Hussein, Fatih Porikli, and Larry Davis. A comprehensive evaluation framework and a comparative study for human detectors. *Intelligent Transportation Systems, IEEE Transactions on*, 10(3):417–427, 2009.

[62] Sergey Ioffe and David A. Forsyth. Probabilistic methods for finding people. *International Journal of Computer Vision*, 43(1):45–68, 2001.

[63] Hui-Xing Jia and Yu-Jin Zhang. Fast human detection by boosting histograms of oriented gradients. In *Image and Graphics, 2007. ICIG 2007. Fourth International Conference on*, pages 683–688. IEEE, 2007.

[64] Thorsten Joachims. *Text categorization with support vector machines: Learning with many relevant features*. Springer, 1998.

[65] Robert Laganière. *OpenCV 2 computer vision application programming cookbook*. Packt Publishing Ltd, 2011.

[66] Kelvin Lee, Che Yon Choo, Hui Qing See, Zhuan Jiang Tan, and Yunli Lee. Human detection using histogram of oriented gradients and human body ratio estimation. In *Computer Science and Information Technology (ICCSIT), 2010 3rd IEEE International Conference on*, volume 4, pages 18–22. IEEE, 2010.

[67] Bastian Leibe, Ales Leonardis, and Bernt Schiele. Combined object categorization and segmentation with an implicit shape model. In *Workshop on Statistical Learning in Computer Vision, ECCV*, pages 17–32, 2004.

[68] Bastian Leibe, Aleš Leonardis, and Bernt Schiele. Robust object detection with interleaved categorization and segmentation. *International journal of computer vision*, 77(1-3):259–289, 2008.

[69] Bastian Leibe and Bernt Schiele. *Interleaving object categorization and segmentation*. Springer, 2006.

[70] Liyuan Li, Weimin Huang, Irene YH Gu, and Qi Tian. Foreground object detection from videos containing complex background. In *Proceedings of the eleventh ACM international conference on Multimedia*, pages 2–10. ACM, 2003.

[71] Zhe Lin, Larry S Davis, David Doermann, and Daniel DeMenthon. Hierarchical part-template matching for human detection and segmentation. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. IEEE, 2007.

[72] Tony Lindeberg. Scale-space theory: A basic tool for analyzing structures at different scales. *Journal of applied statistics*, 21(1-2):225–270, 1994.

[73] Alvin Martin, George Doddington, Terri Kamm, Mark Ordowski, and Mark Przybocki. The det curve in assessment of detection task performance. Technical report, DTIC Document, 1997.

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

[75] Krystian Mikolajczyk, Cordelia Schmid, and Andrew Zisserman. Human detection based on a probabilistic assembly of robust part detectors. In *Computer Vision-ECCV 2004*, pages 69–82. Springer, 2004.

[76] Anuj Mohan, Constantine Papageorgiou, and Tomaso Poggio. Example-based object detection in images by components. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 23(4):349–361, 2001.

[77] R Morzinger, Marcus Thaler, Severin Stalder, Helmut Grabner, and Luc Van Gool. Improved person detection in industrial environments using multiple self-calibrated cameras. In *Advanced Video and Signal-Based Surveillance (AVSS), 2011 8th IEEE International Conference on*, pages 486–491. IEEE, 2011.

[78] Yadong Mu, Shuicheng Yan, Yi Liu, Thomas Huang, and Bingfeng Zhou. Discriminative local binary patterns for human detection in personal album. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.

[79] Stefan Munder and Dariu M Gavrila. An experimental study on pedestrian classification. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(11):1863–1868, 2006.

[80] Duc Thanh Nguyen, Wanqing Li, and Philip Ogunbona. A part-based template matching method for multi-view human detection. In *Image and Vision Computing New Zealand, 2009. IVCNZ'09. 24th International Conference*, pages 357–362. IEEE, 2009.

[81] Duc Thanh Nguyen, Philip Ogunbona, and Wanqing Li. Detecting humans under occlusion using variational mean field method. In *Image Processing (ICIP), 2011 18th IEEE International Conference on*, pages 2049–2052. IEEE, 2011.

[82] Duc Thanh Nguyen, Zhimin Zong, Philip Ogunbona, and Wanqing Li. Object detection using non-redundant local binary patterns. In *Image Processing (ICIP), 2010 17th IEEE International Conference on*, pages 4609–4612. IEEE, 2010.

[83] Mark Nixon and Alberto S Aguado. *Feature extraction & image processing for computer vision*. Academic Press, 2012.

[84] Timo Ojala, Matti Pietikäinen, and David Harwood. A comparative study of texture measures with classification based on featured distributions. *Pattern recognition*, 29(1):51–59, 1996.

[85] Emin Orhan. Particle Filtering. http://www.cns.nyu.edu/~eorhan/notes/particle-filtering.pdf, 11-08-2012. [Online; accessed 17-Sep-2013].

[86] Ovgu Ozturk, Toshihiko Yamasaki, and Kiyoharu Aizawa. Tracking of humans and estimation of body/head orientation from top-view single camera for visual

focus of attention analysis. In *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*, pages 1020–1027. IEEE, 2009.

[87] Yanwei Pang, He Yan, Yuan Yuan, and Kongqiao Wang. Robust cohog feature extraction in human-centered image/video management system. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 42(2):458–468, 2012.

[88] Yanwei Pang, Yuan Yuan, Xuelong Li, and Jing Pan. Efficient hog human detection. *Signal Processing*, 91(4):773–781, 2011.

[89] Constantine Papageorgiou and Tomaso Poggio. A trainable system for object detection. *International Journal of Computer Vision*, 38(1):15–33, 2000.

[90] Marco Pedersoli, Jordi Gonzàlez, Bhaskar Chakraborty, and Juan J Villanueva. Enhancing real-time human detection based on histograms of oriented gradients. In *Computer Recognition Systems 2*, pages 739–746. Springer, 2007.

[91] Nicolas Pinto, Youssef Barhomi, David D Cox, and James J DiCarlo. Comparing state-of-the-art visual features on invariant object recognition tasks. In *Applications of Computer Vision (WACV), 2011 IEEE Workshop on*, pages 463–470. IEEE, 2011.

[92] Michael Rauter. Reliable human detection and tracking in top-view depth images. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2013 IEEE Conference on*, pages 529–534. IEEE, 2013.

[93] Rémi Ronfard, Cordelia Schmid, and Bill Triggs. Learning to parse pictures of people. In *Computer VisionECCV 2002*, pages 700–714. Springer, 2006.

[94] Paul Rosin. Thresholding for change detection. In *Computer Vision, 1998. Sixth International Conference on*, pages 274–279. IEEE, 1998.

[95] Robert E Schapire and Yoram Singer. Improved boosting algorithms using confidence-rated predictions. *Machine learning*, 37(3):297–336, 1999.

[96] Bernt Schiele, Mykhaylo Andriluka, Nikodem Majer, Stefan Roth, and Christian Wojek. Visual people detection: Different models, comparison and discussion. In *Proceedings of the IEEE ICRA 2009 workshop on people detection and tracking*, volume 12, 2009.

[97] Peter Schroder. Wavelets in computer graphics. *Proceedings of the IEEE*, 84(4):615–625, 1996.

[98] William Robson Schwartz, Aniruddha Kembhavi, David Harwood, and Larry S Davis. Human detection using partial least squares analysis. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 24–31. IEEE, 2009.

[99] Richard D Seely, Sina Samangooei, M Lee, John N Carter, and Mark S Nixon. The university of southampton multi-biometric tunnel and introducing a novel 3d gait dataset. In *Biometrics: Theory, Applications and Systems, 2008. BTAS 2008. 2nd IEEE International Conference on*, pages 1–6. IEEE, 2008.

[100] Edgar Seemann, Bastian Leibe, Krystian Mikolajczyk, and Bernt Schiele. An evaluation of local shape-based features for pedestrian detection. In *Proc. BMVC*, volume 4. Citeseer, 2005.

[101] Amnon Shashua, Yoram Gdalyahu, and Gaby Hayun. Pedestrian detection for driving assistance systems: Single-frame classification and system level performance. In *Intelligent Vehicles Symposium, 2004 IEEE*, pages 1–6. IEEE, 2004.

[102] Lauro Snidaro, Christian Micheloni, and Cristian Chiavedale. Video security for ambient intelligence. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 35(1):133–144, 2005.

[103] Chris Stauffer and W Eric L Grimson. Adaptive background mixture models for real-time tracking. In *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, volume 2. IEEE, 1999.

[104] Frédéric Suard, Alain Rakotomamonjy, Abdelaziz Bensrhair, and Alberto Broggi. Pedestrian detection using infrared images and histograms of oriented gradients. In *Intelligent Vehicles Symposium, 2006 IEEE*, pages 206–212. IEEE, 2006.

[105] Nguyen Duc Thanh, Wanqing Li, and Philip Ogunbona. A novel template matching method for human detection. In *Image Processing (ICIP), 2009 16th IEEE International Conference on*, pages 2549–2552. IEEE, 2009.

[106] Duc Thanh Nguyen, Philip O Ogunbona, and Wanqing Li. A novel shape-based non-redundant local binary pattern descriptor for object detection. *Pattern Recognition*, 2012.

[107] Tim van Oosterhout, Sander Bakkes, and Ben JA Kröse. Head detection in stereo data for people counting and segmentation. In *VISAPP*, pages 620–625, 2011.

[108] Vladimir Vapnik. *The nature of statistical learning theory.* springer, 1999.

[109] Senem Velipasalar, Ying-Li Tian, and Arun Hampapur. Automatic counting of interacting people by using a single uncalibrated camera. In *Multimedia and Expo, 2006 IEEE International Conference on*, pages 1265–1268. IEEE, 2006.

[110] Galina Veres, Lee Middleton, Zed Sabeur, Imed Bouchrika, Banafshe Arbab-Zavar, John Carter, Mark Nixon, Helmut Grabner, Severin Stalder, Luc Van Gool, et al. Tools for semi-automatic monitoring of industrial workflows. 2010.

[111] Paul Viola and Michael J Jones. Robust real-time face detection. *International journal of computer vision*, 57(2):137–154, 2004.

[112] Chi-Chen Raxle Wang, Jin-Yi Wu, and Jenn-Jier James Lien. Pedestrian detection system using cascaded boosting with invariance of oriented gradients. *International Journal of Pattern Recognition and Artificial Intelligence*, 23(04):801–823, 2009.

[113] Xiaoyu Wang, Tony X Han, and Shuicheng Yan. An hog-lbp human detector with partial occlusion handling. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 32–39. IEEE, 2009.

[114] Tomoki Watanabe, Satoshi Ito, and Kentaro Yokoi. Co-occurrence histograms of oriented gradients for pedestrian detection. In *Advances in Image and Video Technology*, pages 37–47. Springer, 2009.

[115] Yun Wei, Qing Tian, and Teng Guo. An improved pedestrian detection algorithm integrating haar-like features and hog descriptors. *Advances in Mechanical Engineering*, 2013, 2013.

[116] Greg Welch and Gary Bishop. An introduction to the kalman filter, 1995.

[117] Christian Wojek, Gyuri Dorkó, André Schulz, and Bernt Schiele. Sliding-windows for rapid object class localization: A parallel technique. In *Pattern Recognition*, pages 71–81. Springer, 2008.

[118] Christian Wojek and Bernt Schiele. A performance evaluation of single and multi-feature people detection. In *Pattern Recognition*, pages 82–91. Springer, 2008.

[119] Bo Wu and Ram Nevatia. Detection of multiple, partially occluded humans in a single image by bayesian combination of edgelet part detectors. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 1, pages 90–97. IEEE, 2005.

[120] Xiaozhen Xia, Wuyi Yang, Heping Li, and Shuwu Zhang. Part-based object detection using cascades of boosted classifiers. In *Computer Vision–ACCV 2009*, pages 556–565. Springer, 2010.

[121] Yuan Xin, Shan Xiaosen, and Su Li. A combined pedestrian detection method based on haar-like features and hog features. In *Intelligent Systems and Applications (ISA), 2011 3rd International Workshop on*, pages 1–4. IEEE, 2011.

[122] Tarek Yahiaoui, Cyril Meurie, Louahdi Khoudour, and François Cabestaing. A people counting system based on dense and close stereovision. In *Image and Signal Processing*, pages 59–66. Springer, 2008.

[123] Shengsheng Yu, Xiaoping Chen, Weiping Sun, and Deping Xie. A robust method for detecting and counting people. In *Audio, Language and Image Processing, 2008. ICALIP 2008. International Conference on*, pages 1545–1549. IEEE, 2008.

[124] Wei Zhang, Gregory Zelinsky, and Dimitris Samaras. Real-time accurate object detection using multiple resolutions. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. IEEE, 2007.

[125] Qiang Zhu, Mei-Chen Yeh, Kwang-Ting Cheng, and Shai Avidan. Fast human detection using a cascade of histograms of oriented gradients. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, pages 1491–1498. IEEE, 2006.