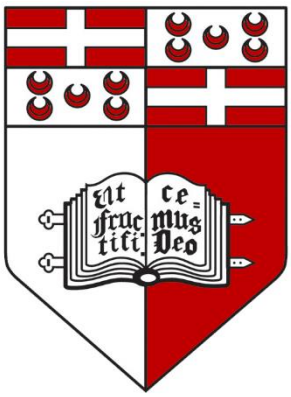
Side view car detection with hierarchical COSFIRE filters

Fabrizio Farrugia

Supervisor: Dr George Azzopardi



University of Malta
L-Università ta' Malta

Faculty of ICT

University Of Malta

June 2016

Submitted in partial fulfilment of the requirements for the degree of B.Sc. ICT in Artificial Intelligence (Hons.)

Contents

List of Figures

List of Tables

Acknowledgements

I would like to express deepest gratitude to my tutor Dr. George Azzopardi for his full support, guidance, understanding and patience throughout my study and research. Without his counsel, my thesis would have been a frustrating experience.

Finally, I would like to thank my parents for their unconditional support during my studies as well as for the help during proof reading.

Abstract

The problem for this project is to find a suitable configuration for side view car detection and localization. The cars need to be detected in complex scenes, which means that the cars will not always be displayed fully and therefore the full shape of the car is not entirely visible.

The method of object recognition which is used is called COSFIRE. Another modification of the COSFIRE filters, called S-COSFIRE, was used as well. Currently, both these methods work based on a single training image. Therefore, the first part of this project involved developing a system whereby both these methods can make use of multiple training images.

The default COSFIRE model is a 2 layer system, whereby the Gabor filters' responses are the input for the COSFIRE filters of the whole prototype. The COSFIRE filters were configured with the shape of a whole car.

Meanwhile, a 3 layer system is when the initial Gabor filters' responses are the input to the first set of S-COSFIRE filters. The input for the last layer of COSFIRE filters is the polar coordinates of the second layer. The fact that there are 3 layers instead of the default 2, forms a hierarchy of S-COSFIRE filters. The keypoints for the S-COSFIRE filters were chosen to be the wheels of the car.

The data set which was used is the UIUC side view car data set, which is used as a benchmark by several research papers. During the evaluation part of this project, the number of considered training images was altered so that a fair comparison between both methods could be accomplished.

The COSFIRE and S-COSFIRE models were both configured to work with multiple prototypes from the positive training examples. More prototypes were needed for the COSFIRE model, since the shapes of cars varies a lot more than the types of wheels found in the data set. The results showed a significant better performance when using the S-COSFIRE model over the COSFIRE model.

Chapter 1 - Introduction

Object recognition is the process of identifying objects of interest in visual types of data such as images or videos. Humans can recognize a considerable amount of objects, animals and people in an image quite easily. Humans can also recognize objects if they are rotated, with a different scale and even if they are partially occluded in the images. However, object recognition procedures depend on pattern matching, learning and pattern recognition procedures based on appearance or feature centred techniques, such as edges, gradients and linear binary arrangements. Object recognition is convenient in applications such as automated parking systems and face detection [1].

Car detection and localization from a side view, is the main area of object recognition of this project. The shape of most cars from a side view is rather similar and therefore a form of generalization is possible. This can be done by either taking the shape of the cars as a whole or by focusing on particular components found in most cars, such as wheels, windows, lights and bumpers.

1.1 Problem Definition

The problem for this project is to find a suitable configuration for side view car detection and localization. The cars need to be detected in complex scenes. This means that the cars will not also be displayed fully and therefore the full shape of the car is not entirely visible, as seen in Fig.1(c). Multiple cars can also be detected in an image since the data set used contains certain images with multiple cars present, as seen in Fig. 1(b).

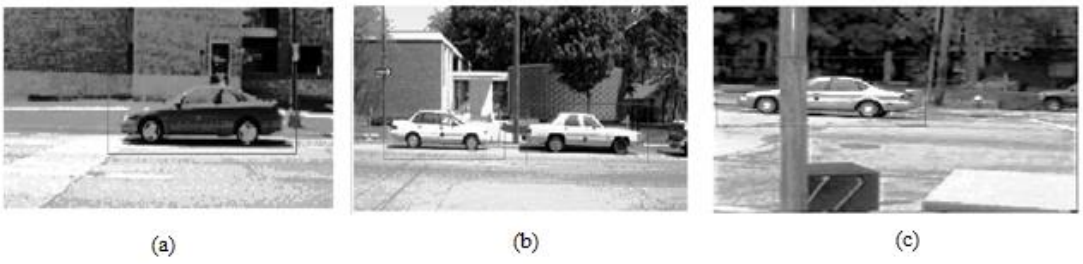


Figure 1: (a)A single car detected in the image. (b)Two cars detected in the image.(c)A single car detected in the image, even though the car is not fully visible.

The method of object recognition which will be used is called COSFIRE [2]. Another modification of the COSFIRE filters, called S-COSFIRE [3], will also be used. Currently, both these methods work based on a single training image. Therefore, the first part of this project is to make both the models work with multiple training examples. COSFIRE and S-COSFIRE filters have been found to be very effective in a number of different applications, however they have never been used for car detection applications.

The keypoint detector known as the Combination Of Shifted FIlter REsponses (COSFIRE) [2] computes a response at a given point as a function of the shifted responses of simpler filters. Gabor filters are used as simpler filters of the COSFIRE filter. The Gabor filters are combined with their corresponding supports at different positions so that a further refined filter is obtained. The filter responses are combined by a weighted geometric mean function. A COSFIRE filter outputs a response if and only if all the essential components of a particular pattern are present.

COSFIRE filters are very useful to detect and identify objects in complicated scenes. A COSFIRE filter is configured to identify an object that a user stipulates in the training stage. At the moment, a COSFIRE filter is configured by analysing the geometrical properties of a single training instance. This means that only one image is used as a training example. Even though this is convenient in applications where there is only a single training example available, it lacks the capability to generalize whenever several training examples are accessible. In the proposed project I will investigate how a COSFIRE filter can be configured by analysing multiple prototype patterns.

Another aspect of this Final Year Project is whether a normal 2 layer COSFIRE filter system yields the same results as a 3 layer COSFIRE filter. A 2 layer system is the default system, whereby the Gabor filters' responses are the input for the COSFIRE filters of the whole prototype. Meanwhile, a 3 layer system is when the initial Gabor filters' responses are the input to the first set of COSFIRE filters. The input for the last layer of COSFIRE filters is the polar coordinates of the second layer. The fact that that there are 3 layers instead of the default 2, forms a hierarchy of COSFIRE filters. This multiple layer system is referred to as S-COSFIRE [3].

The results of configuring the COSFIRE filters over the whole prototypes are compared with the results obtained when using the S-COSFIRE filters, whereby first the configuration is done with specific parts of the prototypes and the whole prototypes are configured.

1.2 Motivation

The main motivation for this Final Year Project is that cars can be detected and located from side view images by using the COSFIRE and/or the S-COSFIRE models whilst using multiple training examples. This means that instead of having just one training example, which is how the current COSFIRE and S-COSFIRE models work currently, the proposed system works with multiple training examples. The efficiency of using the COSFIRE model, which is configured with cars as whole objects will be compared with the S-COSFIRE model, where key components of cars are part of the hierarchical approach for configuration.

The system developed during this Final Year Project can be used for multiple side-view car detection systems. For example, the proposed system can be used in CCTV cameras to detect cars and then whenever a car is identified and is doing something against road regulations, focus on the number plates. The most probable use of this side-view car detection model is for parking tickets. In this particular usage, the cars are detected at the time that they were parked in a specific parking space and the time that the car was left parked is recorded. Therefore, the correct price of the parking ticket can be calculated. Another use for this system, is for security purposes. With the use of cameras, cars can be detected near the premises. Therefore, there is a record of which cars were present at the premises at all times.

1.3 Approach

A COSFIRE filter is theoretically simple and rather straightforward to execute, since it needs the application of certain Gabor filters, Gaussian blurring of their responses, shifting of the blurred responses by explicit vectors, and multiplying the shifted

responses. To determine which Gabor filters to apply and how much to blur and shift the blurred responses are determined in a COSFIRE filter configuration procedure.

For the configuration part of the proposed system, an object detection benchmark data set which consists of 550 cars and 500 non-car objects, is used. The prototypes for the normal COSFIRE model are chosen from the training images of the data set. The prototypes for the S-COSFIRE model are also chosen from the training images.

Figure 2: A training image of a car from the data set.

At first, a prototype is chosen at random and applied on the rest of the training data sets, meaning on both the positive training examples and the negative training examples. The responses for each image are recorded. The True Positives which have a higher response than the highest False Positive are removed from the potential future prototypes. In each iteration, a new prototype is chosen until all prototypes cover the positive training examples.

When configuring the COSFIRE filters with full car prototypes, the area of the car needs to be specified by creating a polygon shape around the car. The cars from the data set have 4 main types, which are dark and white cars and cars facing both left and right. The evaluation of this part is done by applying these operators on the testing images. For each testing image, the maximum response is recorded. The maximum response is achieved by applying each operator once on the image and obviously, the highest response is recorded.

(a) (b)

Figure 3: (a)The COSFIRE filter is configured to represent the car as a whole. The red polygon showcases the outline of the car.(b)The S-COSFIRE filters are configured with the shape and spatial arrangement of the wheels.

For the configuration of the S-COSFIRE filters, parts of a car need to be identified as key components for all cars from a side view. These key components may be the two wheels, the windows, the lights and any combination of them. These components will form part of the first layer of S-COSFIRE filters. The second layer of S-COSFIRE filters will take as input the polar coordinates of the mean of the previous layer, and thus a hierarchy of S-COSFIRE filters is formed. The evaluation of this part will be done in the same manner as the COSFIRE filters such that the results can be compared.

The number of cars detected and their locations in each test image are recorded and compared with the data from the data set. The evaluator from the data set allows for some tolerance with respect to the exact locations.

1.4 Scope

The system works in an off-line manner since the results are not given instantaneously. The scope of this project is to compare 2-layer COSFIRE filters which are selective for the entire shape of a car with 3-layer COSFIRE filters that are selective only for parts of the cars. These results are also compared with other recognition models which used the same data set, which contains images of cars taken from a side view, which are of the same size and therefore scale invariance is not used.

1.5 Aims and Objectives

The main aim of this project is to detect side-view cars. This will be done using the COSFIRE and S-COSFIRE models. These models will be modified to make use of multiple training images. By the end of this project, the use of a hierarchy of S-COSFIRE models is compared with the use of the basic 2-layer COSFIRE model.

- Configure 2-layer COSFIRE filters in the default manner, i.e. with the full shape of cars.
- Configure S-COSFIRE filters with three layers for parts of the prototype cars.
- Configure the COSFIRE filters with multiple prototypes for both the 2-layer COSFIRE model and the multi-layer S-COSFIRE model

- Test and compare the accuracy of both proposed COSFIRE filters.

1.6 Report Layout

The rest of this project is organized as follows: In the Background and Literature Review section, the current COSFIRE filters system and the original paper are analysed as well as other Object Recognition methods that have already been proposed and/or done. In the Specification and Design, a clear picture of the system is given such that it could be implemented with the description available in this section. In the Implementation section, the system is described in much more detail than in the previous chapter. In the Evaluation chapter, the results of the 2-layer COSFIRE model and the 3-layer S-COSFIRE model are compared and analysed. Finally, the Future Work and Conclusions sections discuss any possible future developments and the objectives achieved during this project, respectively.

Chapter 2 - Background and Literature Review

2.1 Background

This project is based upon the existing COSFIRE filters and thus it is quite important to understand the original COSFIRE system and the research paper regarding it [2].

Gabor filters are recognised for their orientation selectivity. Border suppression can also be employed to Gabor filter responses so that responses to texture is reduced as well as improving the detectability of object contours. Fig. 4 illustrates the area of support of a symmetrical Gabor filter, which is selective for stretched out line arrangements with a favoured vertical alignment and a preferred thickness of 4 pixels. The input for a COSFIRE filter are the responses of 2D Gabor filters.

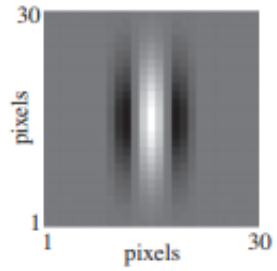


Figure 4:2D Gabor function rendered as a 2D intensity map. Image taken from [4]

Each Gabor filter is characterized by two parameter values around certain positions relating to the centre of the COSFIRE filter. The response of a Gabor filter is denoted by $g_{\lambda,\theta}(x,y)$, with λ being the preferred wavelength, θ being the preferred orientation of a line/edge at position (x,y).

A set of four parameter values, also referred to as a tuple, $(\lambda_i,\theta_i,\rho_i,\phi_i)$ describe the properties of a contour part which is present in the identified region of interest. $\lambda_i/2$ represents the width, θ_i represents the orientation and (ρ_i,ϕ_i) represents the place with respect to the centre of the COSFIRE filter, with ρ being the radius and ϕ being the orientation (angle) as in polar coordinates form. The Gabor filters are blurred before any shift operations are performed. The blurring operation is defined as the calculation of the highest value of the weighted thresholded responses of a Gabor filter. The Gaussian

function $G_\sigma(x,y)$ is used for weighting, with σ representing the standard deviation of which is a function of the distance ρ.

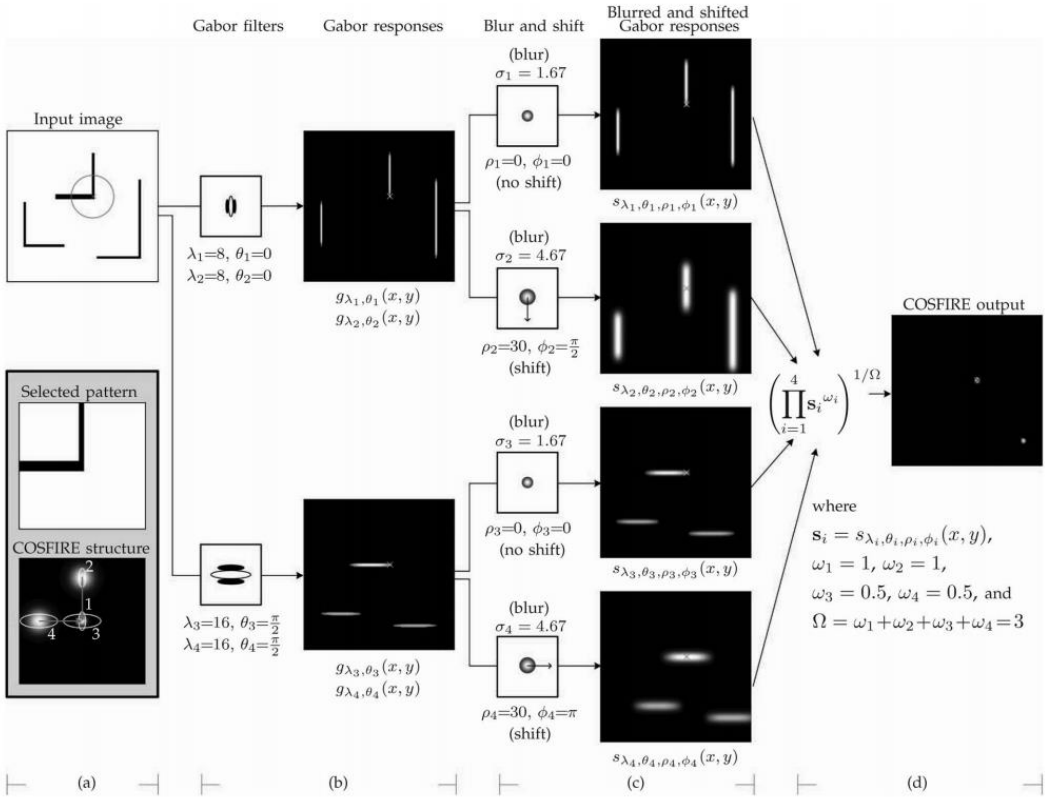


Figure 5: (a) Input image, the selected pattern and the COSFIRE structure (b) Gabor filters and their responses (c) Blurred and shifted responses (d) the final COSFIRE output. This diagram is taken from [2].

In fig. 5(a), an input image of size 256x256 pixels is used. The selected pattern is an expanded sample feature of importance, which is the encircled vertex from the input image. The COSFIRE filter is trained to identify the spatial organization of four contour sections, as shown in the bottom inlayed image. The ellipses show the respective wavelengths and orientations of the involved Gabor filters. Meanwhile, the bright spots are intensity plots for the Gaussian functions which are used to blur the responses of the Gabor filters. Their standard deviations are hand-crafted and are determined as a function of the distance from the centre of the filter. The standard deviation increases with increasing distance from the centre of a COSFIRE filter.

In fig. 5(b), every contour section of the input pattern is distinguished by a Gabor filter with a given preferred λ_i and θ_i. In this case, two distinct Gabor filters were selected from the group of filters, since each of these two Gabor filters detected two parts. In fig. 5(c), the thresholded response for each Gabor filter is blurred (at t_1 =0.2). The resulting blurred responses are consequently shifted by their respective polar coordinate vectors.

In fig. 5(d), the result of the COSFIRE filter is obtained by computing the weighted geometric mean of all the blurred and shifted thresholded Gabor filter responses. The x marker signifies the position of the indicated point of interest, whilst the two local maxima in the output of the COSFIRE filter represent the two comparable vertices from the input image.

In general, COSFIRE filters are applicable to detecting any specific object of interest in complex scenes. The usefulness of the COSFIRE filters was applied in three real-world functions, which are the recognition of handwritten numbers, the detection of vascular bifurcations in retinal fundus images and the detection and recognition of traffic signs in complex scenes [2].

A further development of the COSFIRE methodology is the S-COSFIRE [3] (S stands for Shape), which is a trainable hierarchical object recognition model. The normal COSFIRE model involves a 2 layer system, whereby the Gabor filters' responses are the input of the COSFIRE filters of the whole prototype. The S-COSFIRE filters may form more layers on top of each other such that a hierarchy is formed. The bottom layer of the system remains the same. The initial Gabor filters' responses are the input to the first layer of COSFIRE filters, however, the inputs for each S-COSFIRE layer are the polar coordinates of the previous S-COSFIRE layer.

The bottom layers of S-COSFIRE filters are configured with more basic shapes and components of the full prototype. Taking the example in Fig. 3(b), we may configure an S-COSFIRE that combines the responses from wheel-selective 2-layer COSFIRE filters. The S-COSFIRE model makes use of all the COSFIRE features, including invariance to rotation, scale and reflection.

Figure 6:(a) The circles represent the bottom layer of S-COSFIRE filters. (b) The circles represent the non-overlapping S-COSFIRE filters. Both images are taken from [3].

The S-COSFIRE model has been used in an application for keyword spotting in handwritten documents, whereby the word ‘Germany’ was detected correctly without any false positives. Small parts of the word ‘Germany’ were used as components for the bottom layer of S-COSFIRE filters, as can be seen in Fig. 6(a). Another application of S-COSFIRE filters was for vision of a home tidying robot. The model was configured with an image of a shoe. The configuration of the S-COSFIRE filters, as seen in Fig. 6(b), achieved flawless detection and recognition with all the 60 images which were tested.

2.2 Literature Review

The paper from [5], tackles a similar problem to the one from this project. This paper describes a statistical method for 3D object detection, where the statistics of both object and non-object appearances are represented using a product of histograms. Every histogram signifies the combined statistics of a subdivision of wavelet coefficients and their location on the object. This methodology uses several of these histograms to represent a set of different visual attributes. In this paper, an algorithm which can efficiently identify human faces with out-of-plane rotation was developed, as well as an algorithm which detects cars over a widespread assortment of perspectives.

Cars vary in shape, size, colour as well other small details or parts. These type of differences make object detection more challenging. Other factors that make object detection harder include light sources, shadows and the orientation of an object with respect to the camera. An object detection system should be able to overcome all these challenges so that the object is still distinguished from any other pattern.

To manage these variations, a view-based approach where multiple detectors were applied in parallel and then their results were combined. Each detector was developed separately and each one was specialized to a particular orientation of the object. Eight detectors were used for cars. The left side views are detected by using the seven right side detectors on mirror inverted images.

The image is searched thoroughly in position and scale to detect objects. A heuristic coarse-to-fine strategy was adopted since it faster than doing a direct implementation. The probability ratio for each probable object position is evaluated partially using low resolution visual features. Afterwards, the object candidates which are promising are evaluated further with a higher resolution. The promising objects are those that are beyond a minimum threshold for the partial estimation. Using this methodology to search a 320x240 image over 4 octaves of candidate size takes nearly 5 minutes for cars.

The accuracy of the algorithm was tested by using a test set of 104 images that include 213 cars. These cars cover a vast diversity of models, sizes, orientations, background surroundings and also contain some partial obstruction from numerous cameras. The results are 83%, 86% and 92% detections for when γ is 1.05, 1 and 0.9, respectively. The false detections are 7, 10 and 71 at the same order. The greater the value of γ is, means

that the object is more likely to be present. When γ is 0.9, there are more false detections since the detector is 90% sure that there is a car at that particular location.

Another paper [6], also includes a part with object recognition related to cars. A group of features for object recognition are introduced. This paper claims that template based methods, like the one used in the previous paper [5], have the limitation that it may not satisfactorily capture discrepancies in object appearance, since they tend to be highly selective for a target form and thus lack invariance in regard to object transformations. This paper also claims that SIFT-based features tend to not perform well enough on a generic object recognition task.

The paper [6] introduces a new set of biologically inspired features. Each element from this set being a complex feature obtained by the combination of position and scale tolerant edge detectors over adjacent positions and multiple orientations. The features permit for slight alterations of the input and thus are more flexible than template-based approaches, as well as being more restrictive than histogram-based approaches, such as in [7], since they conserve local feature geometry. This method does not scan over all positions and scales and therefore it is also simpler than most histogram-based approaches.

For every input image, this approach works by first computing a set of features learned from the positive training set and then running a standard classifier on the vector of features obtained from the input image. This system follows the typical normal of object recognition in primate cortex in its most basic form.

The system was tested with datasets which consist of images that either include or do not comprise a sole occurrence of the goal entity. The system was compared with benchmarks from other computer vision systems. For the car object recognition, MIT-CBL datasets were used which included a multi-view car dataset. This dataset included street view images with different types of vehicles.

The benchmark for the car dataset was 75.4%, however when the system was used to a gentle Ada Boost it achieved a 95.1% accuracy. For both the system and the standards, the error rate was reported at the balance point, which is the error rate at which the false positive rate is equal to the miss rate. The results with the C2 structures are better than other component-based [8] and fragment based systems [9]. The system also outperforms SIFT-based systems [7].

The proposed method from the next paper [10], detects and locates the required object. Object localization is done by using the sliding window principle, which means that localization is treated as localized detection. Localized detection means that a classifier function is applied afterwards to sub-images contained by an image and the maximum of the classification score is chosen as the suggestion for the occurrence of an object in that particular area.

The method of this paper [10], depends on a branch-and-bound arrangement that finds the global peak of the fitness function over all potential sub-images and therefore returning the same object positions that an extensive sliding window method would. This method requires significantly less classifier evaluations than candidate regions in the image and normally runs in linear time. The method was called the Efficient Subwindow Search (ESS). The target of ESS is to find a bounding box covering the object.

The paper claims that the sliding windows approach is the most used in object location with bounding boxes. Heuristics are used to speed up the search since going through all sub-images of an n*n image is computationally expensive. The paper claims that ESS guarantees to find the global maximum region and is fast since it depends on a branch-and-bound search, which allows the use of superior classifiers. ESS is adaptable in the choice of fitness function and therefore it not restricted like other object recognition methods which can only find lines and circles in drawings.

The branch-and-bound framework follows the principle that since only a few of the candidate regions contain object instances, the regions with the highest score should be identified and thereafter targeted in a best first manner. The parameter space is split hierarchically into disjoint sets, such that the optimal bounds are kept on all the subsets and unwanted regions are discarded. In ESS, the parameter space is the group of all potential rectangles in an image and for each rectangle set, a bound for the maximum score is calculated. ESS terminates when a rectangle with a quality value as good as the upper bound of the remaining contender sections is identified, which confirms that a global maximum is found. The search is halted if the most favourable group covers only an individual rectangle with the assurance of being the global maximum.

The system was tested with the same dataset which is used to test the proposed system of this Final Year Project. The first test was done with 170 images which contain 200 cars of size 100*40, while the second test consisted of 107 images which contain 139 cars of varied sizes. Since the training examples either represent a car or not, supplementary bounding box information was not required and the SVM was trained with hierarchical spatial pyramid kernel on the complete set of training images. The amount of pyramid levels was varied between L=1 and L=10.

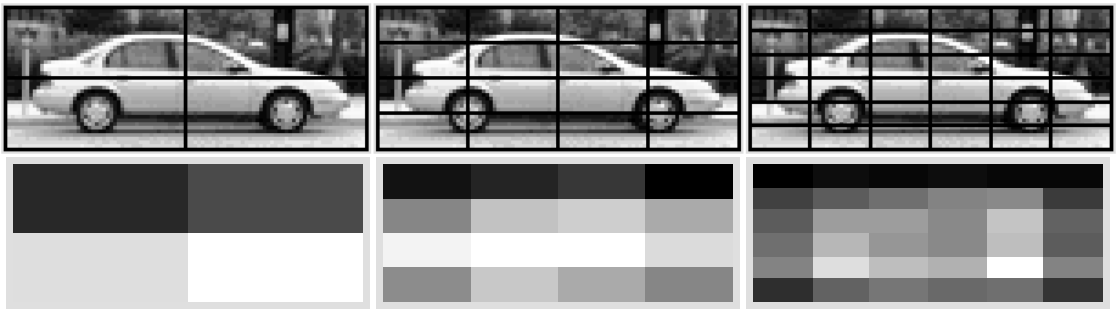


Figure 7: (Top row) Example of a training image with its pyramid segments for levels 2, 4 and 6. (Bottom row) The matching pyramid segment weights, which seem to indicate that the wheels are vital to identifying a car whilst the corners of the image can be ignored. This diagram is taken from [10].

The results were compared with other object recognition models with the same dataset. The ESS achieved an accuracy rate of 98.5% with a 10*10 pyramid for single-scale and 98.4% for multi-scale. ESS with the bag of visual words has an accuracy rate of 90% for the single-scale and only 28.8% for the multi-scale tests. These results confirm that the ESS with finer spatial grid outperforms the other object recognition methods except for the Much/Lowe [11] for the single-scale.

In the paper from [12], a three-Dimensional Scale-Invariant Feature Transform (SIFT) descriptor for video or 3D images is introduced. The bag of visual words method is used to denote videos and to determine relationships among spatio-temporal words is presented. Action recognition can be a complex task and may involve similar actions which are not easy to differentiate, such as running and jogging. The paper claims that the bag of words approach [13] has exposed significant performance for image classification and object recognition in single images.

The bag of words model is extended from to 2D to 3D, with the third dimension being time. The 3D SIFT descriptor translates the data in both time and space in such a way that permits robustness to orientations and noise. Interest points are chosen at random to

allow a more rapid processing time. The complete orientation of the neighbourhood is calculated. Afterwards, the sub-histograms are used to translate local time and space data so that 3D SIFT can generalize the spatio-temporal data.

The system was tested with a dataset of 92 videos of diverse people doing 10 actions, including running, walking, waving with one hand and waving with two hands. The task was to classify the actions that were presented in a particular video. The word vocabulary was calculated from a subset of videos and then the signatures for each video were generated. It was observed that sensible performance on the majority of the actions was obtained excluding for the 'jump' and 'skip' actions, since they were too similar. An average precision of 30.4% was achieved with the 2D SIFT descriptor, whilst an average precision of 82.6% was achieved with the 3D SIFT. The results have shown that this method outperforms other current descriptors on a publicly accessible action dataset.

In the paper from [14], a large, deep convolutional neural network was used to classify the 1.2 million images in the ImageNet LSVRC-2010 competition into their respective classes. ImageNet [15] consists of over 22000 categories of images. A model with a large learning capacity is needed so that it can learn from such a big dataset. Convolutional Neural Networks (CNNs) can have their capacity controlled by changing their depth and breadth. CNNs also make quite accurate assumptions about the nature of the images. A disadvantage of such a model is that it is rather expensive to apply in an enormous dataset of high-resolution images.

The CNNs were trained on subsets of ImageNet, which made overfitting a noteworthy problem. To reduce the problem of overfitting, data augmentation was done. Data augmentation means that the dataset is enlarged artificially using label-preserving transformations, which are computationally free. The final network comprises of five convolutional and three fully connected layers.

The model from this paper achieved a top-1 error rate of 37.5% and a top-5 error rate of 17.0% when tested on the ILSVRC-2010. These result are quite good, especially when compared to the performances of another two models, one with the error rates of 47.1% and 28.2%, respectively, and is based on a method that averages the estimates of six sparse-coding models trained on different features [16]. The other model [17], is an approach that averages the predictions of two classifiers trained on Fisher Vectors, which were computed from two types of densely-sampled features. This method had

error rates of 45.7% and 25.7%, respectively. The results from the CNNs can be improved, but may increase the training time significantly since it is highly dependent on the amount of memory available on the GPUs.

Table 1: Detection rates of different models with the UIUC car test set [30].

Method	**Detection Rates for single-scale**
Agarwal et al.[18]	76.5%
Fergus et al.[19]	88.5%
Fritz et al.[20]	88.6%
Kappor and Winn [21]	94.0%
Much and Lowe [11]	99.94%
Wu and Nevatia [22, 23]	97.5%
Leibe et al. [24]	97.5%
Kuo and Nevatia [25]	98.5%
Todorovic and Ahuja [26]	86.5%
Lampert et al. [10]	98.5%
Zheng et al. [27]	98.0%

The paper from [24] presents a fresh method for multi-view car detection using unsupervised sub categorization as an alternative of manual labelling. Cars differ in appearance depending on their type. Cars appear different when viewed from various angles, like most objects in existence. The manual classification is vague to humans as well as being time consuming. This paper proposes a method which involves a mixture of image descriptors, non-linear dimension reduction, and grouping car samples in an unsupervised manner. A strong tree-structured detector was built based on the sub groups.

This methodology comprehends two main features, which are the unsupervised hierarchical sub categorization and a boosting-based tree structured detector. During the first part, a Locally Linear Embedding (LLE) [28] was used to exemplify high dimensional car samples into a 2D environment, which are later standardized onto a circle. The car samples are clustered into numerous sub groups with the use of the k-means clustering technique. The leaf nodes are made up of these sub groups, which are gradually joined together by the agglomerative clustering technique. This process is done in order to create a tree from bottom upwards. At every node of the tree structured detector, a rejection condition is learned. The Gentle AdaBoost [29] method is used in every level of cascade of a particular node. This process is done such that important features are selected and a robust classifier is formed.

The object samples are divided into two sets in regards to the hierarchical group of sub sets and two nodes are formed, each and every time the classifier is not robust enough to differentiate between the non-object and object samples. The amount of leaf nodes is calculated by the amount of sub groups which were established during the initial section of this methodology. The training procedure halts when the preferred target training precision is achieved. This method achieved a 98.5% detection rate for the single scale test set when used with the UIUC data set [30].

The next paper [22], recommends a boosting based learning technique. This method is referred to as Cluster Boosted Tree (CBT). It is used to automatically create tree structured object detectors. The sample environment is divided by unsupervised clustering which is centered on certain image structures carefully chosen by the boosting procedure. The sub group data of the leaf nodes is processed back to improve their ancestors' classification functions. An alternative manner of doing this could have been by using already defined sub categorization based on domain information.

For the classifier model, the tree configuration recommended by Huang et al. [31] is used. The VBT approach is more efficient for detection tasks than PBT, since does not make use of the cascade detection approach. The learning technique used in [31] requires the user to establish beforehand the sub groups and thus also the tree structure. Furthermore, it requires that the training samples are labelled correctly. These are not straightforward to implement over multiple applications.

In the start of this methodology, the tree classifier comprises of a solitary null branch and nothing more. This signifies that there is only one sub group covering every sample. During every boosting loop, a basic feature based classifier is carefully chosen from the hypothesis space to attach to the branch. This is done for every individual branch of the tree. Afterwards, the efficiency of the recently created weak classifier is calculated as an approximation. Whenever the estimated efficiency is considered as not good enough, the branch is divided into two minor and simpler problems by the use of unsupervised grouping. The unsupervised clustering is executed founded upon the outputs of the designated image structures. The new sub group data is used to produce more tree branches. The classification of the parent nodes is improved whenever new branches are created since their data is transferred from the bottom upwards. At the point when the target accuracy is obtained, the branches will not be allowed to grow any further. This technique achieved 97.5% detection rates when used with the UIUC single scale car data set [30], with both the single view cascade and the multi view CBT.

The following paper [27] introduces a fast process for detecting multi-view cars in static images. The main idea behind the proposed method is that most cars have common characteristics in structure, for example wheels and bumpers. Cars typically have four wheels and two bumpers. The appearance of these characteristics may vary due to the difference in car models, however they still consist of certain geometric fundamentals such as lines and curvatures with strip arrangements. These local elements are used as inspiration for the proposed method whereby a group of image strip features are used to define the forms of these components.

An image strip feature can be considered as a prototype of a curvature section with a particular strip pattern, which is defined by numerous consecutive sections as displayed in fig. 8. The feature output replicates the dissimilarity of these image sections. An estimated procedure was developed built on the integral image technique. A complete set of image strip features can be built if a window is provided. These strip features can contain different curvature sections, strip patterns and locations. The RealBoost algorithm was used to train the cascade classifiers and the CBT classifiers constructed on the image strip features. During the detection part, the sliding window approach was used and the mean-shift clustering procedure was used to join the positive outputs of the classifier and attain the bounding boxes of the objects.

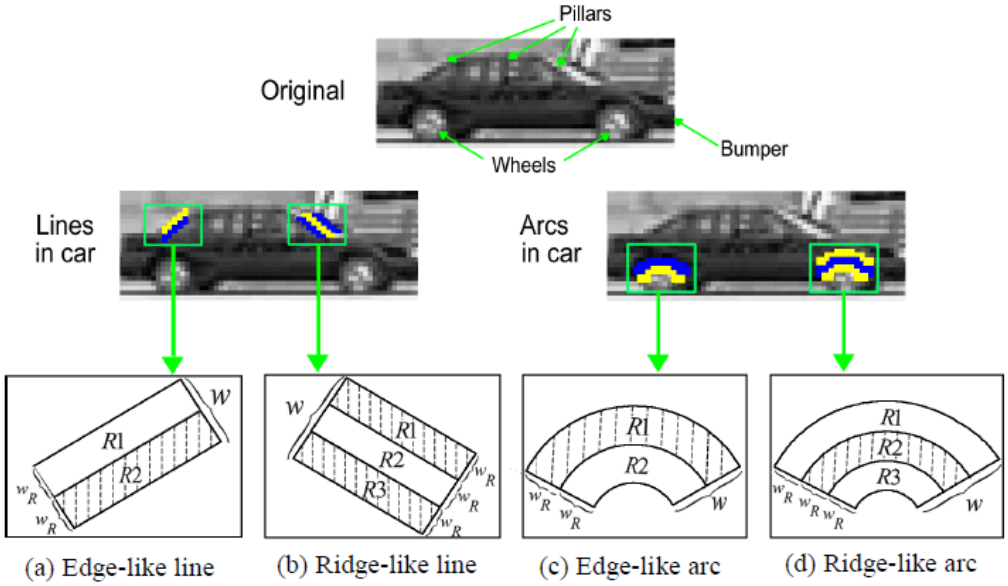


Figure 8: Image strip features vs car structure. Image taken from [27].

The main influences of this methodology are an innovative group of image strip features explicitly intended for car detection as well as a new complexity aware benchmark in boosting context to balance the proficiency of the chosen features. Both the I-Strip and D-Strip methods achieved a 98.0% detection rate in the single scale UIUC car data set [30]. The I-Strip approach only focuses on the complex features with oblique line and curvature arrangements, whilst the D-Strip approach is used when the single strip regions are small, since it calculates the average intensities of every single strip region point-by-point and thus it has a longer processing time than the I-Strip approach whenever the single strip regions are large.

The paper from [23] propositions an approach to concurrently detect and sub divide objects of a recognized category. This paper continues from a previous work [32], where the classifiers were based on edge features. Edges make use of both the size and direction of an edge and their computation is efficient.

During the training stage, a big feature bank is created and for each feature in it, a pair of basic classifiers for detection and subdivision are learned. A modification of the AdaBoost procedure [29] is used to choose decent features from the feature bank. The sub division is expressed as a binary classification problem. The input is an image sample and a pixel position within the sample box, whilst the result is the estimate. An effective neighborhood is defined for every edge feature and its distribution is learned and contained by this neighborhood.

The proposed method from this paper is not restricted to work on detecting a particular object only. The contributions of this paper are the design for the basic sector centered on local shape structures and a boosting procedure for instantaneous learning of detector and sector. When tested with the UIUC single scale car data set [30], this method achieved a detection rate of 97.5%.

Chapter 3 - Specification and Design

3.1 Data Set

The UIUC Car database [30] is the data set used to test this proposed system consists of 1050 images of fixed size 100*40 pixels, which are divided as 550 images containing a single car and 500 images containing no cars. All the images are taken from a side view. The non-car images include objects such as ships, planes and buildings. These 1050 images are labelled as the training images. The data set also contains another two folders, which are both test sets. The first test set consists of 170 images containing 200 cars, with the cars being the same size. The second test set consists of 107 images comprising 139 cars in sizes between 89*36 and 212*85 pixels. As was stated earlier, only the training sets and the first test set (cars have equal scale) are used in this project.

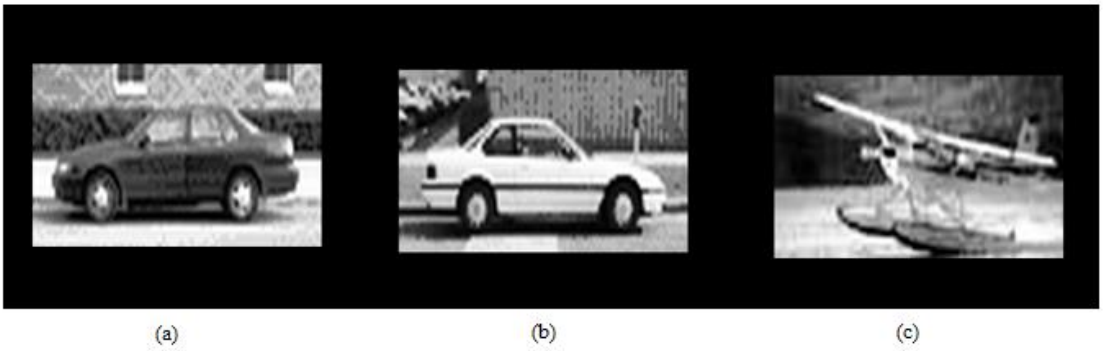


Figure 9: (a)An example of a car facing left (b) An example of a car facing right (c) An example of a non-car object, seaplane. These three images are all taken from the UIUC training data set [30].

The same data set is also used in several research papers, as seen in Table. 1. This gives further evidence that it is an object detection benchmark data set. For example, in [10], the training images are used to extract SURF descriptors. The test sets are used to record the error rates of the method.

3.2 Configuration of the filters

The COSFIRE filters were configured based only on positive prototypes, such as fig. 10(a) and fig. 10(b). This is done since there is no applicable use in configuring COSFIRE filters for each non-car object. At first, a prototype from the positive training examples is chosen at random and used to configure a COSFIRE filter with it. Then it is applied on the rest of the training data sets, meaning on both the positive training examples and the negative training examples. The responses for each image are recorded. The value of the highest response from the negative training examples is recorded as a threshold. The True Positives which have a higher response than this threshold are removed from the training set.

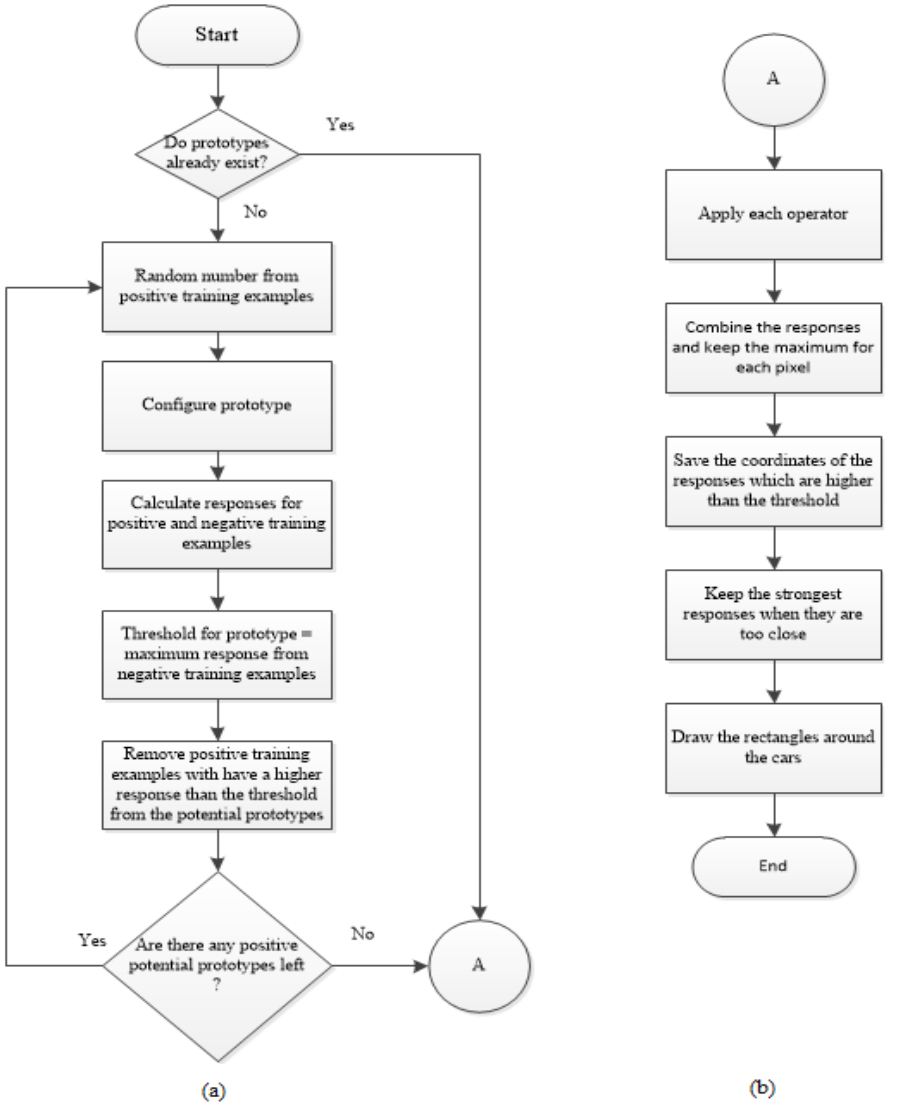


Figure 10: A flowchart which shows how the program works. (a) The configuration process. (b) The application process for each test image.

In each iteration, a new prototype from the remaining positive training examples, is chosen until a sufficient number of COSFIRE filters are configured which are able to detect all cars in the data set. This process provides the highest possible True Positive rate and the lowest False Positive rate based on the training data sets. The prototypes should be able to account for the different types of cars present in the data set. When testing this process, the number of positive training examples to use is varied from 10 to 550 (maximum), such that there could be fair comparison between the COSFIRE and S-COSFIRE models.

Since the prototypes for the COSFIRE filters contains a car background, the car needs to be specified. This was done manually such that a polygon shape was around the car. Therefore, the specified region of interest becomes only the car.

On the other hand, for the S-COSFIRE method, the components of the car for the first layer of S-COSFIRE filters were identified as being the wheels. It can be seen in Fig. 3(b) how the S-COSFIRE layers are configured. In the bottom layer there are only the key components, the wheels, and in the final S-COSFIRE layer, there are the wheels as well as the mean location.

3.3 Application of the filters

The operators for both the COSFIRE filters and S-COSFIRE filters are applied on the first test set, which contains 107 images, containing 200 cars. Scale invariance and rotation invariance were not used since the testing was done with images of the same size and the cars are always sideways.

The maximum response is achieved by applying each operator once on the test image and obviously, the highest response for each pixel is recorded, as shown in fig. 10(b). Where there is a response above a certain threshold, it means that a car was detected there. The centre point of the car is recorded and used to obtain the value of the top left corner. Whenever there are two responses which are too close, only the one with the highest response is saved.

The coordinates of the top left corner for each image and its respective car are saved in a text file, which is evaluated with an evaluator script provided by the data set. A location is considered as a correct detection if it lies within an ellipse with centre at the true location and axes 25% of the object proportions in each way.

The evaluator program outputs the resultant amount of correct detections and false detections, as well as the percentages for the recall, precision and F-measure. These values are saved to allow for comparison between both methods.

Chapter 4 - Implementation

4.1 Configuration of COSFIRE filters

The COSFIRE filters were configured based only on positive prototypes, such as fig. 9(a) and fig. 9(b). This is done since there is no applicable use in configuring COSFIRE filters for each non-car object. The number of considered training is altered between 10, 25, 50, 100 or 550 and always include the first positive training image up until one of these numbers. This is done such that there could be a fair comparison between the COSFIRE and S-COSFIRE models. During this section, the value of considered training examples is taken as 10, so that it is easier to explain in more detail.

At the start of the program, the default COSFIRE parameters are saved and several parameters are adjusted. The modified parameters are: rholist, t_1, t_2, t_3, α and Σ_0 ($sigma_0$). The values for these parameters were determined empirically on the training set. These parameters take the same values for both the COSFIRE and S-COSFIRE models, apart from the rholist. Rholist contains the radii for the concentric circles of the COSFIRE filters. The COSFIRE method requires a maximum radius of 48, which was done since the full car is the prototype and the width of all the cars is at most 100, hence 48 is a bit less than half. Meanwhile, the S-COSFIRE method requires a maximum radius of just 18, which fits perfectly well with the size of the wheels of the cars.

T_1 is changed from the default 0.01 to 0.2, so that more input filters' responses are suppressed. T_2 is set to 0.99 and T_3 is set to 0, so that there is more tolerance for the responses of the COSFIRE filters when they are applied. The value of α is changed from its default value to allow for more blurring of the input filter responses and therefore increase tolerance further away from the centre.

The prototype list contains all the remaining potential prototypes. Therefore, initially it contains the first positive training example until the specified number, in this case 10. At first, a prototype from the positive training examples is chosen at random and applied on the rest of the training data sets, up to a specified value(in this case 10), on both the positive training examples and the negative training examples.

The chosen prototype image is transformed into grayscale and its size is recorded. Since the prototypes contain a car as well a background, the car needs to be specified, for the COSFIRE method. This was done by using roipoly, whereby the user has to manually make a polygon shape around the car. Therefore, the specified region of interest becomes only the car, as can be seen in Fig. 11. This process of configuration is only done in the training stage, as shown in fig. 10 (a).

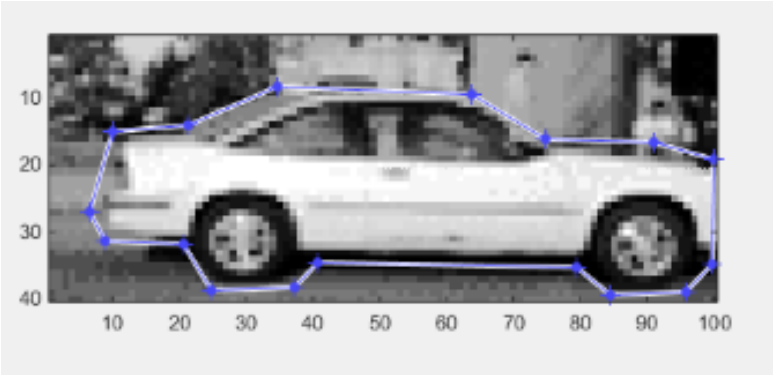


Figure 11: The blue lines represent the outline of the polygon around the car.

The responses of a bank of Gabor filters based on the full image, such as Fig. 11, are used to compute the COSFIRE tuples. This initial configuration does not account for the car and represents the full image. The four COSFIRE tuples are $\lambda_i,\theta_i,\rho_i,\phi_i$. $\lambda_i/2$ represents the width, θ_i represents the orientation and (ρ_i,ϕ_i) represents the location with respect to the centre of the COSFIRE filter, with ρ being the radius and ϕ being the orientation (angle) as in polar coordinates form.

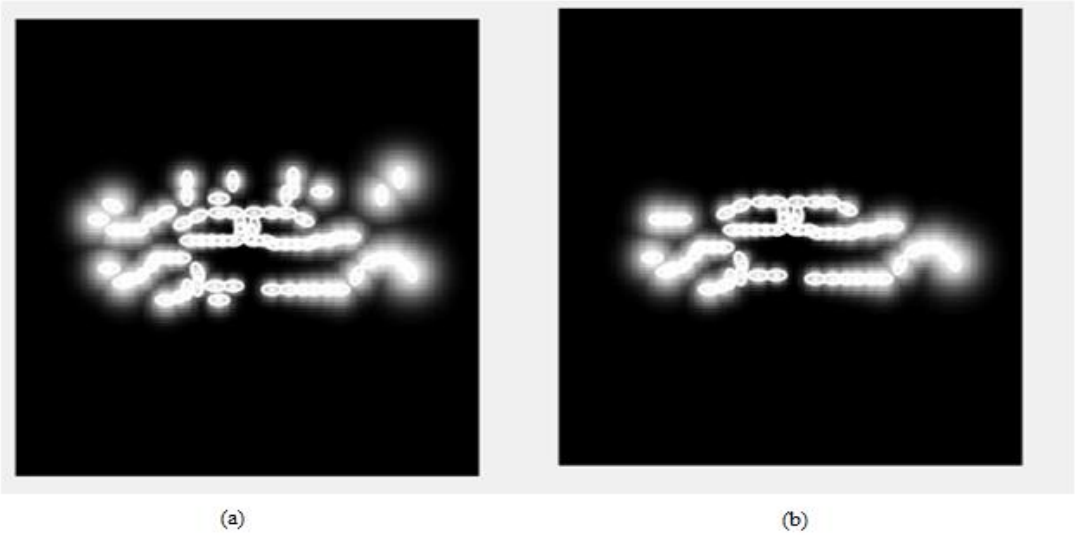


Figure 12:(a)The structure of the COSFIRE filter of the full image from Fig. 11. (b) The structure of the COSFIRE filter of the car from Fig. 11. The ellipses in these diagrams represent the positions, orientations and scale of the Gabor responses. The blobs in these diagrams are bigger when they are further from the centre, which means that there is more tolerance away from the centre.

The polar coordinates of COSFIRE tuples are transformed into their Cartesian equivalent and then into their respective index form. This is done to allow for faster processing time. The next step is removing the responses which are outside the polygon of the car. As can be seen in Fig. 12(b), the car is made clearer than in Fig. 12(a), as well as the shapes of the wheels and windows. The COSFIRE threshold 1 is set to 0 for each operator, so that all responses are available when being applied.

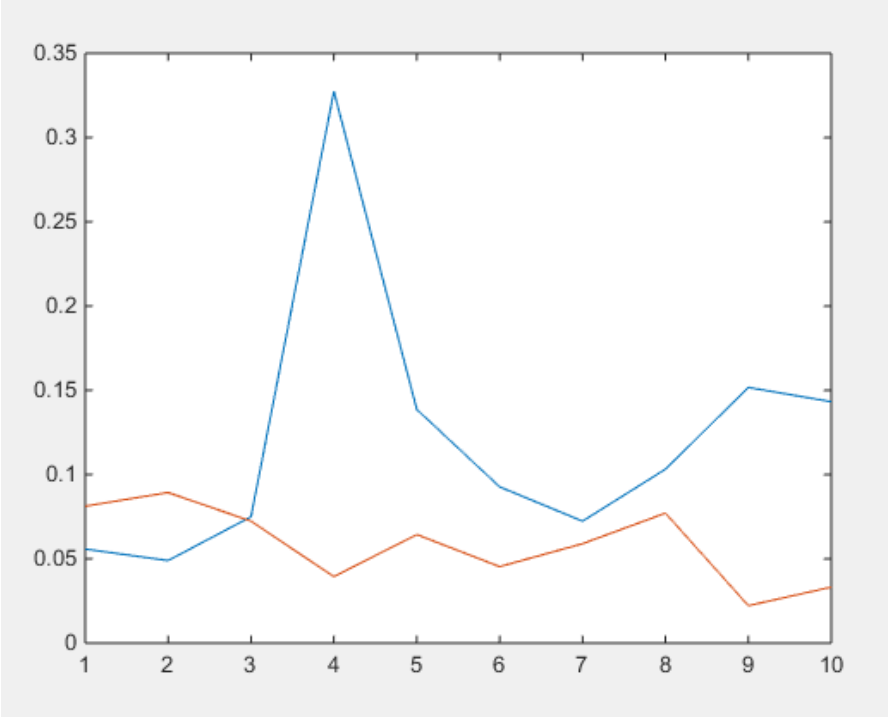


Figure 13: The blue graph represents the responses of the first 10 positive training images. The red graph represents the responses of the first 10 negative training images.

The responses for the first 10(this value can be changed) positive and negative training images are recorded. The value of the highest response from the negative training examples is recorded as a threshold. In this instance, this value corresponds to 0.0893. This value was obtained by the second negative training image, as can be seen in Fig. 13. The True Positives which have a higher response than this threshold are removed from the potential future prototypes, which in this case are images: 4, 5, 6, 8, 9 and 10. The threshold 3 for this particular operator is set to 0.0893. As a result, every operator will have a unique value for threshold 3. The cycle starts again, by choosing another random positive prototype from either 1, 2, 3 or 7. In this case, the prototype which was chosen (7), managed to detect the remaining positive training examples better than any negative training image, as shown in Fig. 14.

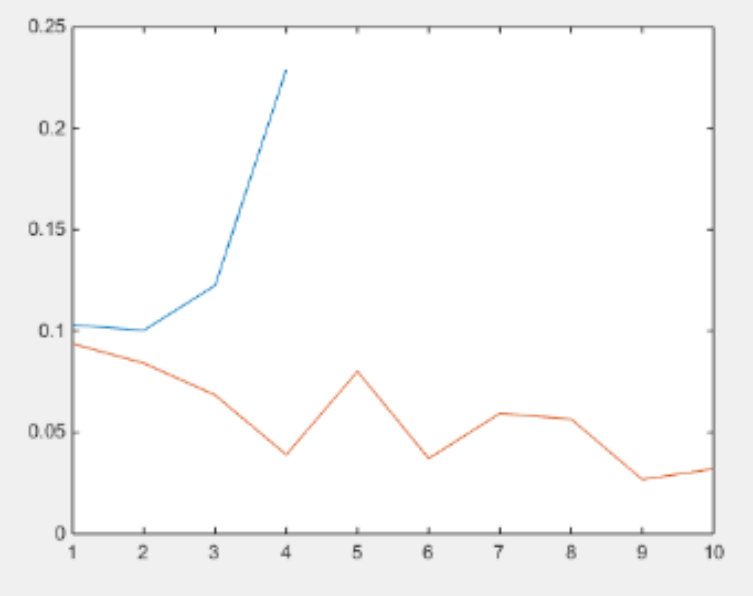


Figure 14: The positive and negative responses of the second prototype.

The S-COSFIRE model has a similar process to the COSFIRE model since the prototypes are also chosen at random from the prototype list and the potential prototype list is updated in the same manner, however it differs in the type of operators. The S-COSFIRE operators have 2 keypoints, which are placed on each wheel of a single car. The coordinates of the centre of each keypoint are saved for each operator. This will be needed when applying the S-COSFIRE filters.

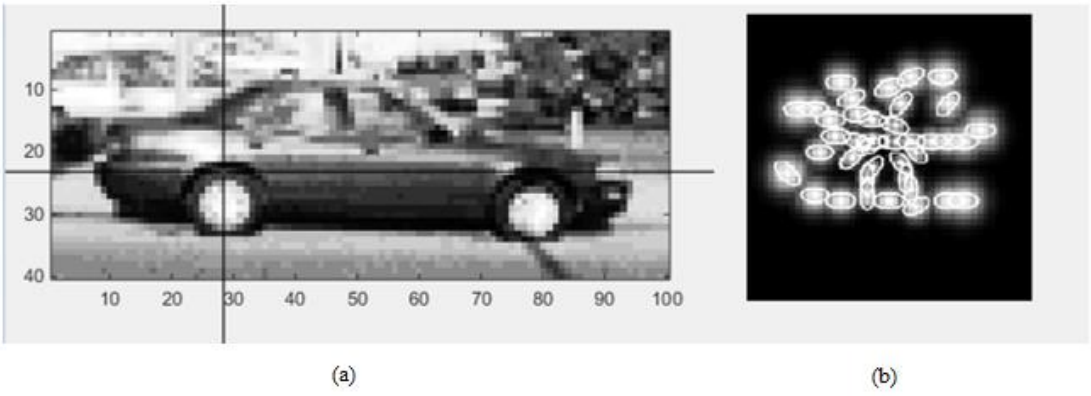


Figure15: (a) The perpendicular lines intersect on the centre of the first S-COSFIRE filter, which is the left wheel. (b) The S-COSFIRE filter output of the first keypoint, which is the left wheel.

The centre of the S-COSFIRE filters is taken at the top of the wheels, as seen in Fig. 15(a), since in a few training images, the wheels are partly obscured at the bottom part. When the number of considered training examples for the S-COSFIRE model was 10, it only required a single operator to detect the respective 10 positive training images without having any false positives.

4.2 Application of COSFIRE filters

The COSFIRE and S-COSFIRE filters are applied on the first data set. This data set has 170 images, containing 200 cars. Each operator is applied to each of these 170 images. The COSFIRE and S-COSFIRE parameter threshold 1 was set to a value of 0. Threshold 1 is used to suppress the input filters responses that are less than a certain fraction t_1 of the maximum. Scale invariance and rotation invariance were not used since the test data set includes only images were the cars are of the same size and the cars are always sideways.

During the application, the test image is populated by a set of tuples for each operator, in this case 20, since 10 operators with reflection enabled. The responses of the bank of input filters are computed and then thresholded by threshold 1, which is set as 0.

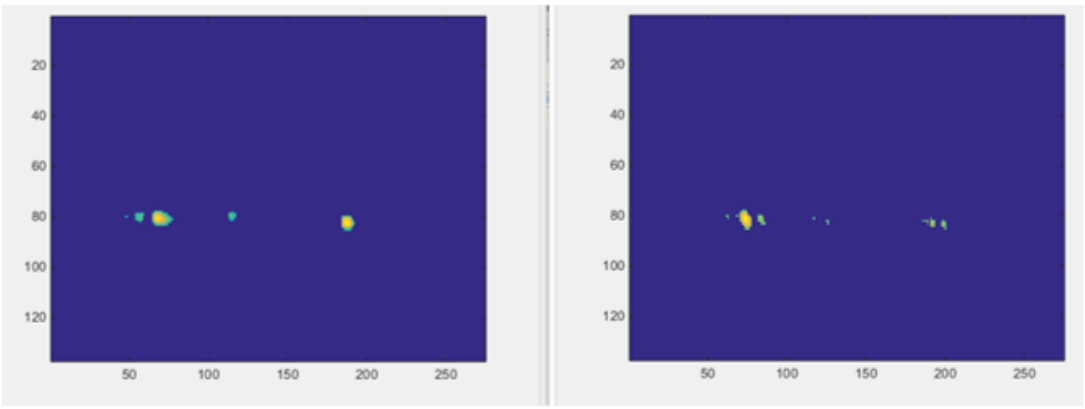


Figure 16: (a) COSFIRE filter response of the first operator on the second test image. (b) COSFIRE filter response of the second operator on the second test image.

For each testing image, the resultant responses for each operator are recorded. In Fig. 16(a), there is the COSFIRE filter response when applying the operator from Fig. 10(b). Both the operators show similar responses in this case. As can be seen in Fig. 16, the bright parts are were the presence of a car was detected the most in the test image. When the colour is closer to yellow, it means that a high level of detection was achieved.

These output responses are combined and the maximum response from all the operators for each pixel in the test image is kept as the final output. The coordinates of the maxima points are saved as the centres for a potentially detected car. In this example, two cars were detected correctly, as can be seen in Fig. 17. A red dot is displayed at the centre coordinates of the detected car and a rectangle is displayed around the car.

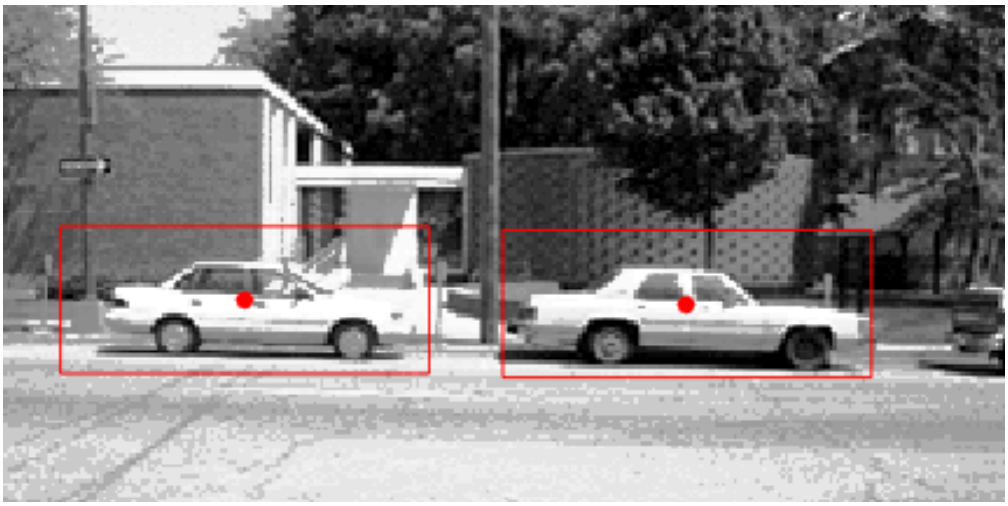


Figure 17: Two cars are detected in this image and are both covered by a red rectangle with a red dot at the centre.

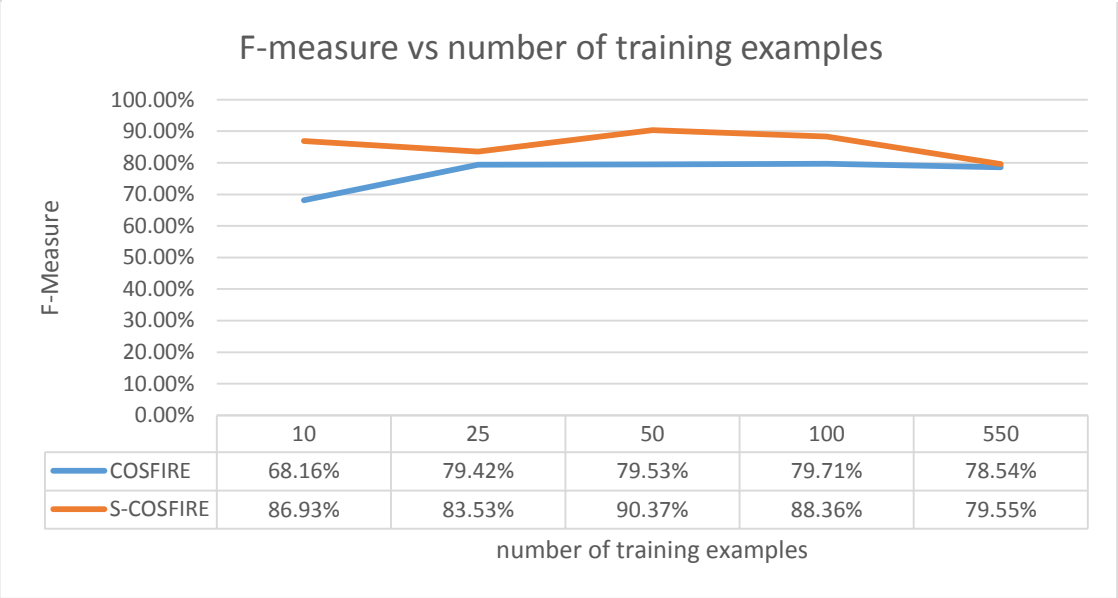
For the COSFIRE model, the top left corner coordinates are always the same distance away from the centre of every operator, since the operators are the same size. When two potential detections are too close to each other, only the detection with the highest response is recorded. Two detections are deemed to be too close if the distance between their centres is less than 80. This value was determined empirically. The top left corner coordinates of each potential detection is saved in a text, which is later used by the evaluator program to obtain the results.

The process is similar for the S-COSFIRE model, however further processing for each operator is required, since each operator has two keypoints (wheels). First of all, each keypoint of each operator needs to be applied to each image. After the application, the response from each keypoint is thresholded to remove weak responses and it is also blurred and shifted. The index of the operator which has the maximum response at a particular point is recorded, since the system of having two keypoints ends up having a different centre for each operator. This is due to the fact that the third layer of S-COSFIRE filters contains the mean of the keypoints from the second layer of S-COSFIRE filters. The creation of the rectangle around potentially detected cars and the coordinates of the top left corner are calculated relating to the centre of the S-COSFIRE keypoints for the particular operator which detected that car.

Chapter 5 - Evaluation

The coordinates of the top left corner for each detected car are saved in a text file. This text file is used by the evaluator program provided with the data set. The evaluator considers a detection as correct if it lies within an ellipse with the centre at the true location and axes 25% of the object proportions in each way. Only one detection is permitted for each object and therefore if more than one rectangle covers the same object, only one is considered as an accurate detection, whilst the others are taken as false detections.

Table 2: Graph of F-measure against the number of training examples.



The number of initial prototypes to choose from was varied from 10 to 550 for both COSFIRE and S-COSFIRE models. Table. 1 shows the results as provided by the evaluator program. The S-COSFIRE model achieved a better F-measure for each number of prototypes.

Table 3: Results for the COSFIRE model.

Training examples	10	25	50	100	550
Prototypes used	3	4	4	13	56
Correct Detections	152	164	171	163	183
False Detections	94	49	59	46	83
Recall	76%	82%	85.50%	81.50%	91.50%
Precision	61.79%	77.00%	74.34%	77.99%	68.80%

The COSFIRE model did not improve much after 25 training examples were used. This occurred since the number of prototypes remained small until 100 training examples and therefore certain types of cars remained undetected. However, with 550 training examples, there were too many prototypes which resulted in increases in both True Positives as well as in False Positives.

Table 4: Results for the S-COSFIRE model.

Training examples	10	25	50	100	550
Prototypes used	1	5	6	11	27
Correct Detections	163	175	183	186	177
False Detections	12	44	22	35	68
Recall	82%	87.50%	91.50%	93%	88.50%
Precision	93.14%	79.91%	89.27%	84.16%	72.24%

Meanwhile, the S-COSFIRE model peaked at 50 training examples with an F-Measure of 90.37%. This is mainly due to using 6 good prototypes, which is still a small number. Even though the number of correct detections was high when using this method, the number of false detections remained relatively small. This shows that using two keypoints for the wheels as a way to configure S-COSFIRE filters is better than configuring COSFIRE filters with a full car prototype.

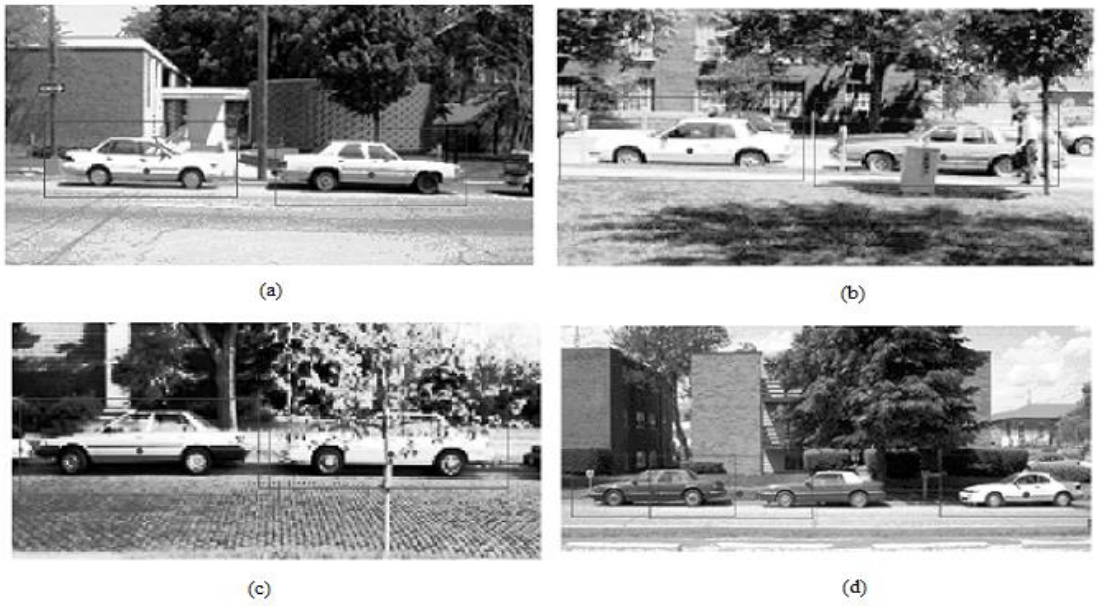


Figure 18: Resultant detections with the S-COSFIRE model: (a) Correct Detection of two cars in full view. (b, c) Correct detection of 2 partially obscured cars. (d) 2 correct detections and 1 false detection.

In Fig.18 there are a sample of four images with their resultant detections. The car on the right in Fig. 18(b) and the car in the right in Fig. 18(c) are both partially obscured, however were still detected by their wheels. This gives further evidence that having the wheels as the keypoints of the S-COSFIRE filters was a good decision. In Fig. 18(d), there is a false detection whereby a wrongly detected car has two wheels, with each wheel belonging to a different car. This was a rare occurrence where the wheels are distant apart by a certain distance which corresponds to the normal distance between the wheels of a real car. When choosing the prototypes for the S-COSFIRE model, a few were left out which only contained one visible wheel since making two identical keypoints in an operator would be pointless.

Chapter 7 - Future Work

This project proved that the use of a 3 layer S-COSFIRE model works better than the flat 2-layer COSFIRE model for detecting side view cars. This means that the S-COSFIRE model could be configured for different objects, other than cars, and obtain satisfactory results.

Further improvements for the side car detection S-COSFIRE model could include configuring the S-COSFIRE filters to work with cars of different scale as well as fixing the problem were a detected car has its wheels from two different cars, as was the case in Fig. 18(d).

Chapter 8 - Conclusions

The default COSFIRE model is a 2 layer system, whereby the Gabor filters' responses are the input for the COSFIRE filters of the whole prototype. The COSFIRE filters were configured with the shape of a whole car.

Meanwhile, a 3 layer system is when the initial Gabor filters' responses are the input to the first set of S-COSFIRE filters. The input for the last layer of COSFIRE filters is the polar coordinates of the second layer. The fact that that there are 3 layers instead of the default 2, forms a hierarchy of S-COSFIRE filters. The keypoints for the S-COSFIRE filters were chosen to be the wheels of the car.

The COSFIRE and S-COSFIRE models were both configured to work with multiple prototypes from the positive training examples. More prototypes were needed for the COSFIRE model, since the shapes of cars varies a lot more than the types of wheels found in the data set.

The results showed a significant better performance when using the S-COSFIRE model over the COSFIRE model, as seen in Table. 1 where the best F-Measure (79.53%) from the COSFIRE model is close to the worst F-Measure (79.55%) of the S-COSFIRE model. Meanwhile, the best F-Measure was 90.37% with the S-COSFIRE model with 6 actual prototypes used.

Therefore, it was concluded that using a 3-layer S-COSFIRE approach, whereby the first layer is made up of Gabor filters and the other two layers are S-COSFIRE filters, is better for side view car detection than using the 2-layer COSFIRE architecture.

Reference List

[1] http://www.mathworks.com/discovery/object-recognition.html

[2] G. Azzopardi and N. Petkov, *Trainable COSFIRE Filters for Keypoint Detection and Pattern Recognition*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 35, No. 2, pp. 490-503, 2013.

[3] Azzopardi, George, and Nicolai Petkov. *Ventral-stream-like shape representation: from pixel intensity values to trainable object-selective COSFIRE models.* Frontiers in computational neuroscience 8, 2014.

[4] G. Azzopardi and N. Petkov, *Automatic detection of vascular bifurcations in segmented retinal images using trainable COSFIRE filters*, Pattern Recognition Letters, vol. 34 (8), pp. 922-933, 2013.

[5] H. Schneiderman and T. Kanade, *A Statistical Method for 3D Object detection Applied to Faces and Cars*, IEEE Conference on Computer Vision and Pattern Recognition, Vol. 1, pp. 746-751, 2000.

[6] T. Serre, L. Wolf and T. Poggio, *Object Recognition with Features Inspired by Visual Cortex*, IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Vol. 2, pp. 994-1000, 2005.

[7] D.G. Lowe. *Object recognition from local scale-invariant features*. In ICCV, pages 1150–1157, 1999.

[8] B. Heisele, T. Serre, M. Pontil, T. Vetter, and T. Poggio. *Categorization by learning and combining object parts.* In NIPS, Vancouver, 2001.

[9] B. Leung. *Component-based car detection in street scene images.* Master's thesis, EECS, MIT, 2004.

[10] C. H. Lampert, M. B. Blaschko and T. Hofmann, *Beyond sliding windows: Object localization by efficient Subwindow search*, IEEE Conference on Computer Vision and Pattern Recognition, pp. 1-8, 2008,

[11] J. Mutch and D. G. Lowe. *Multiclass Object Recognition with Sparse, Localized* Features. In CVPR, pages 11–18, 2006.

[12] P. Scovanner, S. Ali and M. Shah, *A 3-Dimensional SIFT Descriptor and its Application to Action Recognition,* Proceedings of the 15th international conference on Multimedia, pp. 357-360, 2007.

[13] G. Csurka et al., *Visual Categorization with Bags of Keypoints*, ECCV, 2004.

[14] A. Krizhevsky, I. Sutskever and G. E. Hinton, *ImageNet Classification with Deep Convolutional Neural Networks,* Advances in neural information processing systems, 2012.

[15] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. *ImageNet: A Large-Scale Hierarchical Image Database.* In CVPR09, 2009.

[16] A. Berg, J. Deng, and L. Fei-Fei. *Large scale visual recognition challenge 2010.* www.imagenet.org/challenges. 2010.

[17] J. Sánchez and F. Perronnin. *High-dimensional signature compression for large-scale image classification.* In Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on, pages 1665–1672. IEEE, 2011.

[18] S. Agarwal, A. Awan, and D. Roth. *Learning to detect objects in images via a sparse, part-based representation.* Pattern Analysis and Machine Intelligence, IEEE Transactions on, 26(11) pages 1475–1490, Nov. 2004.

[19] R. Fergus, P. Perona, and A. Zisserman. *Object class recognition by unsupervised scale-invariant learning.* In CVPR, 2003.

[20] M. Fritz, B. Leibe, B. Caputo, and B. Schiele. *Integrating representative and discriminant models for object category detection.* In ICCV, 2005.

[21] A. Kapoor and J. Winn. *Located hidden random fields: Learning discriminative parts for object detection.* In ECCV, 2006.

[22] Wu, Bo, and Ram Nevatia. *Cluster boosted tree classifier for multi-view, multi-pose object detection.* Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on. IEEE, 2007.

[23] Wu, Bo, and Ram Nevatia. *Simultaneous object detection and segmentation by boosting local shape feature based classifier.* Computer Vision and Pattern Recognition, 2007. CVPR 2007. IEEE Conference on. IEEE, 2007.

[24] B. Leibe, A. Leonardis, and B. Schiele. *Robust object detection with interleaved categorization and segmentation.* IJCV 2008, 77(1) pages 259–289, May 2008.

[25] Kuo, Cheng-Hao, and Ramakant Nevatia. *Robust multi-view car detection using unsupervised sub-categorization.* Applications of Computer Vision (WACV), 2009 Workshop on. IEEE, 2009.

[26] S. Todorovic, and N. Ahuja. *Extracting Subimages of an Unknown Category from a Set of Images.* CVPR, 2006.

[27] Zheng, Wei, and Luhong Liang. *Fast car detection using image strip features.* Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on. IEEE, 2009.

[28] S. T. Roweis and E. Al. *Nonlinear dimensionality reduction by locally linear embedding.* Science, 290, pages 2323–2326, 2000.

[29] J. Friedman, T. Hastie, and R. Tibshirani. *Additive Logistic Regression: a Statistical View of Boosting.* The Annals of Statistics, 38(2), 2000.

[30] S. Agarwal, A. Awan and D. Roth. *UIUC Image Database for Car Detection,* http://cogcomp.cs.illinois.edu/Data/Car/

[31] J. Friedman, T. Hastie, and R. Tibshirani. *Additive Logistic Regression: a Statistical View of Boosting.* The Annals of Statistics, 38(2), 2000.

[32] B. Wu, and R. Nevatia. *Detection of Multiple, Partially Occluded Humans in a Single Image by Bayesian Combination of Edgelet Part Detectors*. ICCV, 2005.