MASSACHUSETTS INSTITUTE OF TECHNOLOGY
ARTIFICIAL INTELLIGENCE LABORATORY

A.I. Technical Report No. 1685 May 2000

A Trainable System for Object Detection in Images and Video Sequences

Constantine P. Papageorgiou
cpapa@ai.mit.edu

This publication can be retrieved by anonymous ftp to publications.ai.mit.edu.

Abstract

This thesis presents a general, trainable system for object detection in static images and video sequences. The core system finds a certain class of objects in static images of completely unconstrained, cluttered scenes without using motion, tracking, or handcrafted models and without making any assumptions on the scene structure or the number of objects in the scene. The system uses a set of training data of positive and negative example images as input, transforms the pixel images to a Haar wavelet representation, and uses a support vector machine classifier to learn the difference between in-class and out-of-class patterns. To detect objects in out-of-sample images, we do a brute force search over all the subwindows in the image. This system is applied to face, people, and car detection with excellent results.

For our extensions to video sequences, we augment the core static detection system in several ways – 1) extending the representation to five frames, 2) implementing an approximation to a Kalman filter, and 3) modeling detections in an image as a density and propagating this density through time according to measured features. In addition, we present a real-time version of the system that is currently running in a DaimlerChrysler experimental vehicle.

As part of this thesis, we also present a system that, instead of detecting full patterns, uses a component-based approach. We find it to be more robust to occlusions, rotations in depth, and severe lighting conditions for people detection than the full body version. We also experiment with various other representations including pixels and principal components and show results that quantify how the number of features, color, and gray-level affect performance.

Contents

List of Figures

Chapter 1

Introduction

Until recently, digital information was practically limited to text. Now, we are in the midst of an explosion in the amount of digital *visual* information that is available. In fact, the state of technology is quickly moving from where databases of *images* are standard to where the proliferation of entire *video* databases will be de rigeur.

With this increase in the amount of online data available, there has been a corresponding push in the need for efficient, accurate means for processing this information. Search technology thusfar has been almost exclusively targeted to processing textual data [Yahoo!, Inc., 1994, Compaq Computer Corp., 1995, Lycos, Inc., 1995, Excite, Inc., 1995]; indeed, the amount of online data was heavily weighted towards text and the automatic processing of image information has significantly lagged. Only recently, as the amount of online image data has exploded, have there been systems that provide indexing and cataloging of image information in mainstream search services [IBM, 1993, Virage, Inc., 1994]. These systems provide various means for searching through mainly static online image libraries but are fairly limited in their capabilities. We can expect that, as the standard in available online visual information transitions from images to entire video libraries, effective techniques and systems for searching this data will quickly become imperative. Consider this hypothetical problem: a digital historian is looking for all of the CNN video footage from the past five years that shows Bill Clinton and Monica Lewinsky together in the same scene. Manually searching this data would be a daunting if not impossible task. A system that would be able to automatically search for specific objects and situations would be indispensable.

In addition to the Internet as a motivator and forum for video search, the improved, cheaper processing power of computers has opened the door to new applications of image and video searching that were previously infeasible, if not inconceivable. In the near future, we can expect on-board automotive vision systems that inform or alert the driver about people, track surrounding vehicles, and read street signs. Security systems will soon be able to detect when a person is in the field of view, and will be able to intelligently ignore benign "intruders" like domestic animals. Robots will be able to autonomously navigate through complicated terrain, by detecting different landmarks and comparing these to internal maps. One day, there may even be smart bombs that are able to home in on specific parts of buildings or bridges by "seeing" the target and determining where the correct point of impact should be. Devices that implement object detection could also be useful for applications that aid the blind and deaf.

1.1 Object Detection

Fast, robust object detection systems are fundamental to the success of these types of next-generation image and video processing systems. Object detection systems are capable of searching for a specific class of objects, such as faces, people, cars, dogs, or airplanes. In contrast, the problem of recognition, which is the ability to identify specific instances of a class, deals with understanding the difference between my face and your face [1] not the difference between faces and things that are not faces; see [Murase and Nayar, 1995] for relevant work in object recognition.

On the surface, this may seem trivial; people are able to immediately detect objects, with little, if any, thought. People, though, have had the benefit of millions of years of evolution and development. In contrast, object detection for computers is a nontrivial task as we are faced with the problem of giving a mass of wires and mosfets the ability to see. We must deal with questions such as: How are images represented digitally? What are the characteristics of people, for instance, that distinguish them from similar looking objects like columns and fire hydrants? How do we tell a computer program to look for this information?

In this thesis the reader will find an analysis of these questions and several proposed solutions to the problem of automatic object detection in images and video sequences.

The work in this thesis addresses the problem of object and pattern detection in video sequences of cluttered scenes. The general problem of object detection by computers is a difficult one as there are a number of variables that we cannot account for or model in any but the most contrived situations. The system should not make any assumptions about the scene lighting, the number of objects present, the size or pose of the objects, or motion, among other characteristics.

There are two basic angles this problem could take: static images or video sequences. If we would like to detect objects in static images, the problem becomes a pure pattern classification task; the system must be able to differentiate between the objects of interest and "everything else." With the variability in the scene and the uncontrollable conditions identified above, the model of a person must be rich enough to cope with these variations.

On the other hand, if the problem is to detect objects in video sequences, there is a richer set of information available, namely the dynamical information inherent in the video sequence. However, for a general purpose system that does not make limiting

[1] These two types of systems often complement one another: the first step in a recognition system usually is to locate the object.

Figure 1-1: An illustration of the inherent differences in doing detection in static images and video sequences. In static images, the detection task becomes a pure pattern classification problem. Dynamical information available in video sequences (usually) makes the detection problem easier.

assumptions about the objects, we cannot exclusively rely on *motion* information per se. What if the particular scene is of a group of people standing at a bus stop? A system that relies on motion to detect people would clearly fail in this case. Figure 1-1 contrasts the two types of visual data.

What we need is a technique or combination of techniques that use a model that is rich enough to both a) describe the object class to the degree that it is able to effectively model any of the possible shapes, poses, colors, and textures of the object for detection in static images, and b) when we know that we are processing video sequence, harness the dynamical information inherent in video sequences without making any underlying assumptions about the dynamics.

1.2 Pattern Classification and Machine Learning

Pattern classification encompasses a wide variety of problems. We assume that some system is presented with a certain pattern, $\mathbf{x}$, and a set of possible classes, y_i, one of which is the true class of the pattern. The elements of the pattern are individual features that can encode characteristics of the pattern like height, color, length, pixel value, center of mass, mood, etc. – literally, anything that can describe the thing we are trying to classify. The goal of the system is to decide to which class $\mathbf{x}$ belongs. Put forth in this manner, it is essentially equivalent to asking "what kind of thing is the pattern?". In this thesis, we will focus on the two class classification problem, where every pattern falls into exactly one of two possible classes.

There are several ways that the system can encode the knowledge needed to tackle this problem. Using a rule-based approach, a user can describe a set of rules that the system should follow in the decision process. While these systems have been successful for certain types of problems, they typically involve significant effort in engineering the rules and hence are quite expensive to develop. A more promising approach is one where the system *learns* to classify the patterns. This is exactly the approach we will take.

Machine learning describes a large set of techniques, heuristics, and algorithms that all share a single common characteristic: by using a set of examples, they somehow impart upon a system the ability to do a certain task. In the context of our pattern classification problem, we are interested in presenting the system with a set of example patterns of both classes from a set of *training* data and have it automatically learn the characteristics that describe each class and differentiate one class from the other. The positive examples are labeled as +1 and the negative as -1. The goal, and measure of success, is the degree of performance that the trained system achieves on a set of examples that were not present in the training set, or test set. What we are determining when using the test set to evaluate the performance is how well the system is able to *generalize* to data it has never seen.

The problem of learning from examples is formulated as one where the system attempts to derive an input/output mapping, or equivalently, a model of the domain, from a set of training examples. This type of approach is particularly attractive for several reasons. First and foremost, by learning the characteristics of a problem from examples, we avoid the need for explicitly handcrafting a solution to the problem. A handcrafted solution may suffer from the users imposition of what he thinks are the important features or characteristics of a decision problem. With a learning based approach, the important features and relationships of a decision problem are automatically abstracted away as a trained model. On the other hand, learning based approaches may suffer from the problem of poor generalization on account of overfitting, where the model has learned the decision problem "too well" and is not able to generalize to new data.

For a given learning task, we have a set of ℓ N-dimensional labeled training examples:

$$(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_\ell, y_\ell) \;\; \mathbf{x}_i \in R^N, y_i \in \{-1, +1\} \tag{1.1}$$

where the examples have been generated from some unknown pdf, $P(\mathbf{x}, y)$. We would like the system to learn a decision function $f_a : \mathbf{x} \to y$ that minimizes the expected risk,

$$R(\alpha) = \int |f_\alpha(\mathbf{x}) - y| dP(\mathbf{x}, y) \tag{1.2}$$

In most cases, we will not know $P(\mathbf{x}, y)$; we simply see the data points that the distribution has generated. Thus, direct minimization of Equation 1.2 is not possible. What we are able to directly minimize is the empirical risk, the actual error over the training set,

$$R_{emp}(\alpha) = \frac{1}{\ell}\sum_{i=1}^{\ell} |f_\alpha(\mathbf{x}_i) - y_i| \tag{1.3}$$

Learning engines that minimize empirical risk tend to *overfit* the training data. This means that, though the error rate on the training data may be extremely low, the error rate on out-of-sample test data could be quite high. There are a variety of techniques to overcome this including introducing a prior [MacKay, 1992, Wolpert, 1995], early stopping [Nelson and Illingworth, 1991], and using a hold-out data set [Bishop, 1995].

These methods for improving generalization are largely ad hoc. Recently, a new training technique for classifiers, called Support Vector Machines, has emerged that directly minimizes both the empirical risk and the complexity of the classifier at the same time. We will be exclusively using SVM training for our system. This technique is described in more detail in Section 2.4 and Appendix B.

1.3 Previous Work

In this section, we describe prior work in object detection in static images and in video sequences that is relevant to our technique. The descriptions are grouped according to application area: faces, people, and cars, from the simpler to more complex systems. The descriptions are very roughly chronological.

The types of technologies used for object detection have largely been dictated by computing power. For instance, early systems for object detection in static images generally used edge detection and simple heuristics, while recent systems that have access to much more storage and processing power are able to use large sets of training data to derive complex models of different objects. The first systems for object detection typically used motion information to segment out the moving object. While assuming the availability of this type of information can be seen as restrictive for certain applications, motion remains a powerful source of information used to both provide better detection and faster processing even in some of the more recent systems.

1.3.1 Face Detection

Much of the work in object detection in static images has been concentrated in face and people detection. These choices of domains are obvious ones as detecting faces and people are important steps in most systems where there is some sort of human-computer interaction.

Most of the early systems that find faces in images use simple shapes and constraints. Yang and Huang, 1993 and 1994 developed a system that uses local constraints on an image pyramid to detect components of faces. A similar rule based approach is described in [Kotropoulos and Pitas, 1997]. These rule based approaches have the benefit of low computational cost and work well for faces on account of the regular interior structure across faces. This concept was extended by [Sinha, 1994a, Sinha, 1994b] who introduced the idea of the "template ratio" — encoding a human face as a set of binary relationships between the average intensities of 11 regions. The assumption behind the template ratio approach was that these relationships will hold regardless of significant changes in illumination direction and magnitude. For example, the eye sockets are almost always darker than the forehead or the cheeks. The success and robustness of the simple rule-based and template ratio approaches for face detection indicate that a representation based on the encoding of differences in average intensities of different regions is a promising direction.

A number of systems take advantage of the regular structure in faces by processing patterns to find fine scale edges corresponding to individual facial features and then fit the geometry of the components to a deformable template. In these systems [Yuille, 1991, Yuille *et al.*, 1992, Kober *et al.*, 1994, Kwon and Lobo, 1994, Venkatraman and Govindaraju, 1995], the deformable templates are hand-crafted based on prior knowledge of the structure of human faces.

Often, these systems are plagued by the noise and spurious information in the fine scale edges they use. One way of countering the effects of inconsistent fine scale edges is to use multiscale gradient features instead of the finer information. In [Leung *et al.*, 1995, Burl *et al.*, 1995, Burl and Perona, 1996], they use local templates based on oriented Gaussian derivatives to match eye, nose, and mouth features on the human face and determine valid arrangements of these features using random graph matching. Similar systems include [Yow and Cipolla, 1996b, Yow and Cipolla, 1996a, Yow and Cipolla, 1997] where the features are 2nd derivative Gaussians combined using a Bayesian network and [Guarda *et al.*, 1998] where gradient features are combined in a genetic algorithm framework. There have also been efforts that use Laplacian of Gaussian features [Hoogenboom and Lew, 1996] or gradient features [Qian and Huang, 1997] to compare an entire face pattern to a set of templates, instead of individual components.

Wavelets, describe in Section 2.3.1 and Chapter A, provide another formulation for multiscale intensity differences. Some motivation for using wavelets as features is provided in [Micheli-Tzanakou and Marsic, 1991], where they hypothesize that wavelets may provide excellent features for detection and recognition when combined with a classifier. Initial work showing that wavelet response peaks over objects in a fairly uniform background is shown in [Braithwaite and Bhanu, 1994] and [Chapa and Raghuveer, 1996]. In a related area of research, [Kruger *et al.*, 1997] determine the pose of a face by matching wavelet features to learned 3D head representations.

While many face detection systems have used features based on intensity differences of some sort, the regularity in the intensity between faces (see [Sinha, 1994a, Sinha, 1994b]) makes it possible for approaches using intensity features to succeed for face detection. We will see later that this is not the case with people images, where the intensities have little regularity across an ensemble of examples. Lew and Huang, 1996 use the Kullback relative information as a measure of closeness between a face candidate and a known template image of a face. Similarly, [Colmenarez and Huang, 1996, Colmenarez and Huang, 1997, Colmenarez and Huang, 1998] present face detection systems using quantized intensity images combined with maximum likelihood and cross entropy decision measures. Lakshmi Ratan *et al.*, 1998, describe a system that detects faces by using a dynamic programming approach over quantized intensity values to match a small number of prototype images.

One of the recent focuses in the literature has been trainable systems for face detection that take advantage of increases in computing power and storage. These systems for detecting unoccluded vertical frontal views of human faces in images have been developed using example-based approaches; a typically large set of training data is processed by the systems, eventually enabling them to differentiate between faces and non-faces. These view-based approaches can handle detecting faces in cluttered scenes and have shown a reasonable degree of success when extended to handle non-frontal views [Sung, 1995, Sung and Poggio, 1998, Rowley *et al.*, 1997]. These systems can be subdivided into those using density estimation based methods and those that are "pure" pattern classification techniques.

For the density based techniques, [Sung and Poggio, 1994, Sung and Poggio, 1998], [Moghaddam and Pentland, 1995], [Duta and Jain, 1998], and [Reiter and Matas, 1998] essentially model faces in a high dimensional space and do detection by finding where new patterns fall in this density. In these systems, some measure of distance from or closeness to a density is needed; typically, a Mahalanobis-like metric is used. The work of [Schneiderman and Kanade, 1998] develops a face detection system that is also based on density estimation but they model the joint statistics of local patterns. Rikert *et al.*, 1999, accomplish detection by clustering the output of multiscale, multiorientation features and model the clusters as mixtures of Gaussians, for a set of both in-class and out-of-class data. Of course, all of these approaches rely on the availability of large data sets so that density estimation is possible. The system of [Duta and Jain, 1998] is essentially equivalent to that of [Sung and Poggio, 1994, Sung and Poggio, 1998] and that of [Reiter and Matas, 1998], and is based on [Moghaddam and Pentland, 1995].

Pattern classifiers like neural networks and support vector machines can be viewed as techniques that take as input large sets of labeled data and find a nonlinear decision surface that separates the in-class (faces) patterns from the out-of-class (non-faces) patterns. The benefit of these types of systems is that no explicit modeling needs to be done. One the other hand, they are typically expensive computationally – especially during training – and it can be difficult to extract intuition from a trained system as to what they are "learning" internally. Pattern classification approaches to face detection have been developed by [Rowley *et al.*, 1995, Rowley *et al.*, 1998], [Vaillant *et al.*, 1994], and [Osuna *et al.*, 1997a]. Rowley and Vaillant use neural networks with different receptive fields and Osuna uses a support vector machine to classify the patterns.

There have been several systems that combine the previous ideas and use wavelet features as input to example based systems. Whereas the particular class of wavelets and learning techniques differ, the general structure of these approaches are similar to ours but use very small numbers of features, are developed for a particular domain, and are not rigorously tested on large out-of-sample data sets. Philips, 1994 uses matching pursuit over Gabor wavelet features to detect features on a face. This technique decomposes an out-of-sample pattern into a linear combination of wavelets and matches the linear coefficients to those of a known face. The systems of [Mirhosseini and Yan, 1996] and [Shams and Spoelstra, 1996] use a neural network trained on Gabor wavelet features to detect eyes, while [Weber and Casasent, 1997] use a similar approach to detect tanks in IR imagery.

There are other types of information that have been used in face detection systems. While these features are exclusive to faces and do not generalize to other object classes, they nevertheless can be powerful cues for face detection. Several researchers use the fact that the colors of human faces fall into a narrow range. These systems use the prior knowledge that color can be used to segment out skin regions as information for detection [Saber and Tekalp, 1996, Wu *et al.*, 1999, Garcia *et al.*, 1999]. Another system by [Crowley and Berard, 1997] uses the unique time signature of blinks as well as color to detect faces.

1.3.2 People Detection

The existing work in people detection has largely not addressed the detection problem *per se*; rather, most existing systems use some sort of prior knowledge, place heavy restrictions on the scene, assume fixed cameras with known backgrounds, or implement tracking.

The early systems that detect people focused on using motion and simple shapes or constraints; much of this is due to computational limitations of the time. Tsukiyama and Shirai, 1985, use simple shape descriptions to determine the location of leg motion against a white background and a distance measure is utilized to determine the correspondences between moving regions in consecutive images. This system can handle multiple people in an image, but requires a stationary camera and only uses leg movement to track people. Leung and Yang, 1987, use a voting process to determine candidate edges for moving body parts and a set of geometric constraints to determine actual body part locations. This architecture also assumes a fixed camera. Another important restriction is that it is only able to deal with a single moving person.

The use of 3D models has been prominent in finding people in video sequences. This type of system, while adequate for particular, well-defined domains, involves using domain specific information in the development of the model and is not easily portable to new domains. Hogg, 1983, describes a system that is based on modeling a human figure as a hierarchical decomposition of 3D cylinders, using dynamic constraints on the movement of the limbs as well. Edge detection is used to determine the possible locations of body parts and a search tree is used to determine the location that maximizes a "plausibility" measure, indicating the likelihood that there is a person at this location. Rohr, 1993, develops a system using similar 3D cylindrical models of the human body and kinematic motion data. Model contours are matched with edges that are found in an image using a grid search method. A Kalman filter is used to determine the exact position and pose of the walking person across multiple frames. Both these architectures assume a fixed camera and a single moving person in the image.

A large number of systems use the fact that, if our camera is fixed and we know what the scene background is, we can effectively subtract the background from a new image of the scene and recover the moving objects. They can further restrict the domain to specify that the only objects that move will invariably be people. Once the moving bodies have been segmented out, it is possible to do tracking with "blob" models of the different body parts assigned by a maximum a posteriori approach as in [Wren *et al.*, 1995]. A similar system, [Sullivan *et al.*, 1995], models and tracks moving people using a simple deformable model. McKenna and Gong, 1997, cluster the motion information to separate different bodies of motion and subsequently use a Kalman filter to track different people.

This use of background subtraction can be extended to do more than just detection and tracking. Kahn and Swain, 1995, use motion and color to segment a person, then use prior knowledge about the geometry of the body to determine where the person is pointing. In [Haritaoglu *et al.*, 1998], they present a real-time surveillance system for "recognizing" actions that analyzes basic shapes and is able to cope with occlusions and merging bodies (a related approach is describe in [Takatoo *et al.*, 1996]). Similarly, [Davis and Bobick, 1997] present a system for recognizing actions based on time-weighted binarized motion images taken over relatively short sequences. However, the system is not used for detection.

In a different vein, [Heisele *et al.*, 1997] use the clusters of consistent color to track moving objects. Initially, the system computes the color clusters for the first image in a sequence. The system recomputes the cluster centroids for subsequent images, assuming a fixed number of clusters. To track an object, the clusters corresponding to that object are manually labeled in an initial image and are tracked in subsequent frames – the user is, in effect, performing the first detection manually. The authors highlight, as future work, investigating object detection with this algorithm. An important aspect of this system is that, unlike other systems described in this section, this technique does not assume a stationary camera. This system has been combined with a time delay neural network to detect and recognize pedestrians [Heisele and Wohler, 1998].

Another approach that differs from the traditional techniques is to mark specific points on the human body with sensors or lights and to record data off of moving people. This information is more suited to understanding human motion and can subsequently be used for analysis or animation. Campbell and Bobick, 1995, take a different approach to analyzing human body motion. They present a system for recognizing different body motions using constraints on the movements of different body parts. Motion data is gathered using ballet dancers with different body parts marked with sensors. The system uses correlations between different part motions to determine the "best" recognizer of the high-level motion. They use this system to classify different ballet motions. Lakany and Hayes, 1997, also use moving light displays (MLDs) combined with a 2D FFT for feature extraction to train a neural network to recognize a walker from his/her gait.

While systems relying on motion or background information are prevalent for people detection, there have been recent efforts in developing systems that actually do detection. In [Forsyth and Fleck, 1997] and [Forsyth and Fleck, 1998], they describe a system that uses color, texture, and geometry to localize horses and naked people in images. The system can be used to retrieve images satisfying certain criteria from image databases but is mainly targeted towards images containing one object. Methods of learning these "body plans" of hand coded hierarchies of parts from examples are described in [Forsyth and Fleck, 1997]. In a direction of work more similar to ours, a recent system by [Gavrila and Philomin, 1999] describes a technique for finding people by matching oriented outlines generated by an edge detector to a large set of 5,500 people template images via a distance transform measure. The system is related to ours in that it looks at some sort of intensity difference information, but, unlike ours, only considers outlines composed of fine scale edges and uses an explicit holistic match metric.

1.3.3 Car Detection

Cars, like people, have been less heavily studied than faces as a domain for object detection. Indeed, most systems for car detection typically impose many of the same restrictive assumptions as for people detection, largely due to the variability in the patterns and shapes and the availability of characteristic motion cues in video sequences. Much of the work relies on background subtraction[Ali and Dagless, 1990, Ebbecke *et al.*, 1997]. Several systems use some other segmentation method that keys off of the fact that cars occur against fairly constant backgrounds of pavement. For example, [Kalinke *et al.*, 1998] describe a technique based on local image entropy. Beymer *et al.*, 1997, present a traffic monitoring system that has a car detection module. This portion of the system locates corner features in highway sequences and groups features for single cars together by integrating information over time. Betke *et al.*, 1997, and Betke and Nguyen, 1998, use corner features and edge maps combined with template matching to detect cars in highway video scenes. This system can afford to rely on motion since it is designed for a fairly narrow domain – that of highway scene analysis from a vehicle.

Most of these systems are applying heuristics that are specific to the fairly limited domains in which the systems are designed to run, typically from a fixed camera monitoring a piece of road, highway, or intersection, or from the front of a vehicle.

A more relevant example of a true trainable car detection system is that of Rajagopalan *et al.*, 1999, which clusters the positive data in a high dimensional space and, to classify an unknown pattern, computes and thresholds a distance measure based on the higher order statistics of the distribution. This technique has a good deal in common with the face detection system of [Sung and Poggio, 1994, Sung and Poggio, 1998]. A similar detection system is described by Lipson, 1996 , who uses a deformable template for side view car detection. In this system, the wheels, mid-body region, and regions above the wheels are roughly detected based on photometric and geometric relations. The wheels are then more precisely localized using a Hausdorf match. Processing is confined to high resolution images, which is possibly a restriction for more general detection tasks. This system has been applied to scene classification [Lipson *et al.*, 1997, Ratan and Grimson, 1997], and shares some conceptual similarity with that of [Sinha, 1994a, Sinha, 1994b].

All these systems have succeeded to varying degrees but have relied on the following restrictive features:

- explicit modeling of the domain
- assumption that the object is moving
- stationary camera and a fixed background
- marking of key moving features with sensors/lights
- implement tracking of objects, not detection of specific classes

Model-based approaches need a large amount of domain specific knowledge while marking features is impractical for real world use. The tracking systems have problems handling the entrance of new objects into the scene. To overcome this problem, a tracking system would need to emulate a detection system. This work will overcome these problems by introducing an example-based approach that learns to recognize patterns and avoids the use of motion and explicit segmentation.

1.4 Our Approach

The approach taken in this thesis has two components. First, we develop a robust, trainable object detection system for static images that achieves a high degree of

Figure 1-2: Example images of people used in the training database of our static detection system. The examples show variation in pose, color, texture, and background.

performance. We then develop several modifications and enhancements of the original system that allow us to take advantage of dynamical information when we process video sequences.

The core static detection system is one that *learns from examples*. We present it with a set of training data that are images of the object class we would like to detect and a set of data that are examples of patterns not in the object class, and the system derives an implicit model from this data. To allow the system to find a better model, we do not directly use the pixel images as training data since the pixel patterns have a high degree of variability. Rather, we transform the pixel images into a representation that encodes local, oriented, multiscale, intensity differences and provides for a more descriptive model of a variety of object classes.

The system learns using patterns of a fixed size, but in general images we do not know what size objects we will be looking for, how many of these objects will be in the scene, and where they will be located. To detect objects at all sizes and locations, we implement a brute force search in the image looking at all locations and sizes of patterns. We assume that the orientations that we are interested in must be expressed in the training data.

When we are processing video sequences, the naive approach of using our core system is to directly apply the static detection system to each frame sequentially. This, however, ignores all the dynamical information inherently available in video sequences. We can take advantage of the facts that objects that appear in one frame typically appear in approximately the same position in subsequent frames and that objects do not (usually) spontaneously appear in a frame when they are not present in previous frames. This general idea can be coded more rigorously in several ways, each which use the core static detection technique as their basis. We will describe and provide empirical results using these different methods.

1.5 Thesis Contributions

Representation

From the example images of people shown in Figure 1-2, it is clear that a pixel-based representation is plagued by a high degree of variability. A learning-based approach would have a difficult time finding a consistent definition of a person using this type of representation. We describe a new representation where the features are derived as the responses of filters that detect oriented intensity differences between local adjacent regions. This is accomplished within the framework of Haar wavelets and the particular transform we use results in an overcomplete dictionary of these Haar features.

We will also investigate the power of this representation compared with both a pixel representation and principal components analysis (PCA), local and global representations, respectively.

In addition, we present a set of experiments that capture the informational content inherent in color images versus gray level images by comparing the performance achieved with color and gray level representations. In other words, these experiments will show the exact value of color information, in the context of effective detection performance.

Learning Machine

Traditional training techniques for classifiers, such as multilayer perceptrons, use empirical risk minimization and only guarantee minimum error over the training set. These techniques can result in overfitting of the training data and therefore poor out-of-sample performance. In this thesis, we use a relatively new pattern classification technique, support vector machines, that has recently received a great deal of attention in the literature. The number of applications of SVMs is still quite small, so the presentation of SVMs as the core learning machine represents a significant advancement of the technique in the context of practical applications.

Support Vector Machines (SVM) [Vapnik, 1995, Cortes and Vapnik, 1995, Burges, 1998] approximates structural risk minimization which is a well-founded learning method with guaranteed theoretical bounds on the generalization performance. SVMs minimize a bound on the generalization error by simultaneously controlling the complexity of the machine and the performance on the training set, and therefore should perform better on novel data. The SVM framework is characterized by the use of nonlinear kernels that maps data from the original input space into a much higher dimensional feature space in which the learning capability of the machine is significantly increased. In the higher dimensional feature space, an SVM classifier finds the optimal separating hyperplane, that is, the one that maximizes the margin between the two classes.

Faces, People, Cars

Much of the previous work in object detection in static images has focused on the problem of face detection [Sung and Poggio, 1998, Rowley *et al.*, 1998, Vaillant *et al.*, 1994, Moghaddam and Pentland, 1995, Osuna *et al.*, 1997a]. While this is an important domain for static object detection systems, we consider frontal face detection to be essentially a solved problem[2].

To explore the generality of our system and highlight its performance in less studied domains, we provide in-depth results on people and car detection, as well as face detection. To our knowledge, this work is the first exposition of people detection in static images, without making any assumption on motion, scene structure, background, or the number of people in the scene. In addition, while there have been several car detection systems developed in the literature, they too typically require the use of dynamical information.

This thesis explores the three object detection domains – faces, people, and cars – and shows that our single, general purpose, trainable architecture is able to handle each of these classes of objects with excellent results.

Detection by Components

The most prevalent problem with our static detection system is its difficulty in detecting objects when a portion of the pattern is occluded or there is little contrast between the background and part of the pattern. This is a consequence of the fact that we are training our system on the complete patterns of our objects of interest. Many object classes are decomposable into a *hierarchy* of constituent elements. For instance, we know that a person is composed of a head, left arm, right arm, torso, and legs. When we see these components in the proper configuration, we know that we are looking at a person.

One would expect that if we knew there was a head, left arm, and legs in the proper configuration, but we could not see the right arm, that this may still be a person. In other words, if we look for the core building blocks, or *components* of an object, and allow some leniency in allowing one or two of the components that make up the object to be missing, this may result in a more robust detection system than our full pattern approach.

This thesis presents a component based framework for object detection and shows that for a test domain of people detection, the system performs better than the full body detection system.

[2]We note that the more general problem of *pose invariant* face detection still has not been sufficiently dealt with.

Dynamical Detection

This thesis presents experiments of object detection in video sequences using several different methods that take advantage of dynamical detection information in different manners. We will use the brute force technique of applying a static detection system to individual frames of a video sequence as our baseline.

Our first system is a pure pattern classification approach to dynamical object detection; here we seek to circumvent the need for 1) the extensive engineering that is quite typical in current dynamical detection systems and 2) assuming particular underlying dynamics. We will modify our base static detection approach to represent dynamic information by extending the static representation into the time domain. With this new representation, the system will be able to learn the dynamics of people, with or without motion. The system will learn what a person looks like and what constitutes valid dynamics over short time sequences, without the need for explicit models of either shape or dynamics.

The second system to take advantage of dynamical information is a rule based module that integrates information through time as an approximation to a Kalman filter. Kalman filtering theory assumes an underlying linear dynamical model and, given measurements of the location of a person in one image, yields a prediction of the location of the person in the next image and the uncertainty in this prediction. Our heuristic smooths the information in an image sequence over time by taking advantage of this fundamental *a priori* knowledge that a person in one image will appear in a similar position in the next image. We can smooth the results through time by automatically eliminating false positives, detections that do not persevere over a small subsequence.

The third system we will develop uses a new approach to propagating general, multi-modal densities through time, based on the so called Condensation technique [Isard and Blake, 1998]. This technique has a significant advantage over the Kalman filter, namely that it is not constrained to model a single Gaussian density.

A Practical Application

This thesis presents a real-time application of a particular optimized version of our static detection system. Our people detection technology has been integrated into a system for driver assistance. The combined system, including our people detection module, is currently deployed "live" in a DaimlerChrysler S Class demonstration vehicle.

1.6 Outline

In Chapter 2, we describe our core trainable object detection system for static images, with details on our wavelet representation and support vector machine classification and test results. Chapter 3 investigates the use of alternate representations including pixels and principal components analysis for the purposes of object detection. The chapter also describes a manual technique for feature selection, as well as experiments quantifying how training set size affects detection performance. In Chapter 4, we describe the component based approach for object detection and test it on the domain of people detection. Chapter 5 highlights a real application of our system as part of a driver assistance system in a DaimlerChrysler test vehicle and describes experiments using a focus of attention module. In Chapter 6, we extend the static system into the time domain to take advantage of dynamical information when we are processing video sequences; several different approaches are described and tested. Chapter 7 summarizes our results and provides some direction for future work.

Chapter 2

The Static Detection System

2.1 Architecture

The architectural overview of our system is provided in Figure 2-1 as applied to the task of people detection and shows the training and testing phases. In the training step, the system takes as input 1) a set of images of the object class that have been aligned and scaled so that they are all in approximately the same position and are the same size and 2) a set of patterns that are not in our object class. An intermediate representation that encapsulates the important information of our object class is computed for each of these patterns yielding a set of positive and negative feature vectors. These feature vectors are used to train a pattern classifier to differentiate between in-class and out-of-class patterns.

In the testing phase, we are interested in detecting objects in out-of-sample images. Figure 2-12 presents an algorithmic summary of the detection process. The system slides a fixed size window over an image and uses the trained classifier to decide which patterns show the objects of interest. At each window position, we extract the same set of features as in the training step and feed them into our classifier; the classifier output determines whether or not we highlight that pattern as an in-class object. To achieve multi-scale detection, we iteratively resize the image and process each image size using the same fixed size window.

This section addresses the key issues in the development of our trained pattern classifier: the representation and the learning engine.

2.2 Training Data

Our example based approach uses a set of images of an object class to learn what constitutes an in-class and out-of-class pattern. Here, we take people detection as a sample domain. Since the output of our system will be an image with boxes drawn around the people, the training process needs data that reflects what is and is not a person, so we need our positive data to be examples of people. To ensure that our classification engine will learn on data that has consistent information, we require that the example images of people be scaled to the same size and aligned so that the

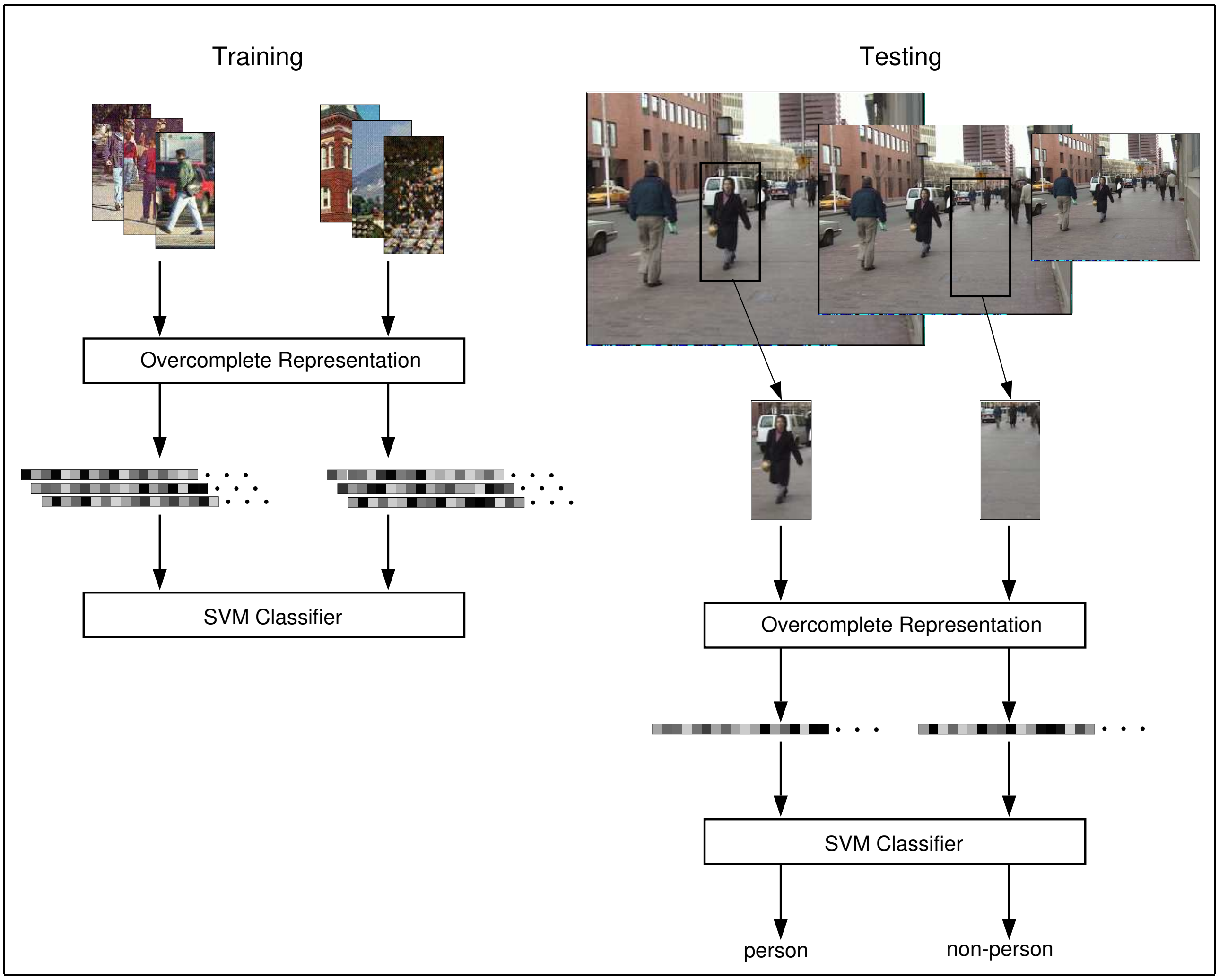


Figure 2-1: The training and testing phases of our system.

body is located in the same position in each image. We have developed a simple tool that allows a user to click on a few identifying marks of an object in an image and then automatically cuts, scales, and aligns the pattern for use in training. Negative training data is gathered by randomly sampling patterns in images that do not contain the object of interest.

Figure 2-2: Examples from the database of faces used for training. The images are gray level of size 19×19 pixels.

2.3 Representation

2.3.1 Wavelets

One of the key issues in the development of an object detection system is the representation of the object class. Even within a narrowly defined class of objects such as "faces" or "people," the patterns can show a great deal of variability in the color, texture, and pose, as well as the lack of a consistent background. Our challenge is to develop a representation that achieves high inter-class variability with low intra-class variability.

Figure 2-3: Examples from the database of people used for training. The images are color of size 128 $\times$ 64 pixels. The examples vary in pose, color, texture, and background.

To motivate our choice of representation, we can start by considering several traditional representations. Pixel based and color region based approaches are likely to fail because of the high degree of variability in the color in certain object classes like "people" and the number of spurious patterns. Traditional fine scale edge based representations are also unsatisfactory due to the large degree of variability in these edges.

The representation that we use is an overcomplete dictionary of Haar wavelets in which there is a large set of features that respond to local intensity differences at several orientations. We present an overview of this representation here; details can be found in Appendix A and [Mallat, 1989, Stollnitz *et al.*, 1994]. The Haar wavelets in their possible orientations are shown in Figure 2-5b.

For a given pattern, the wavelet transform computes the responses of the wavelet filters over the image. Each of the three oriented wavelets – vertical, horizontal, and diagonal – are computed at several different scales allowing the system to represent coarse scale features all the way down to fine scale features. In our object detection

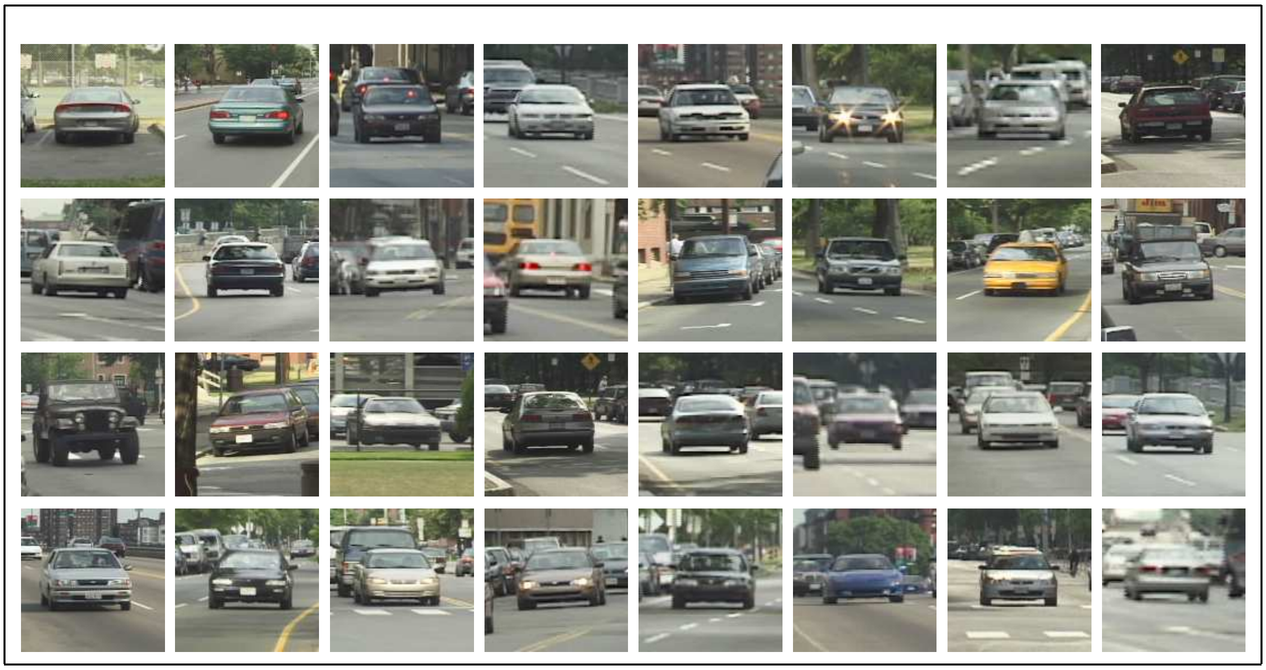


Figure 2-4: Examples from the database of cars used for training. The images are color of size 128×128 pixels, normalized so that the front or rear bumper is 64 pixels wide.

systems, we use 2 consecutive scales of wavelets. In the traditional wavelet transform, the wavelets do not overlap; they are shifted by the size of the support of the wavelet in x and y. To achieve better spatial resolution and a richer set of features, our transform shifts by $\frac{1}{4}$ of the size of the support of each wavelet, yielding an overcomplete dictionary of wavelet features (see Figure 2-5c). The resulting high dimensional feature vectors are used as training data for our classification engine.

There is certain *a priori* knowledge embedded in our choice of the wavelets. First, we use the absolute values of the magnitudes of the wavelets. This tells the system that a dark object on a light background and a light object on a dark background have the same information content. Second, for color images, we compute the wavelet transform for a given pattern in each of the three color channels and then, for a wavelet of a specific location and orientation, we use the one that is largest in magnitude. This allows the system to use the most visually significant features.

Motivation

Our main motivation for using wavelets is that they capture visually plausible features of the shape and interior structure of objects that are invariant to certain transformations. The result is a compact representation where dissimilar example images from the same object class map to similar feature vectors.

With a pixel representation, what we would be encoding are the actual intensities

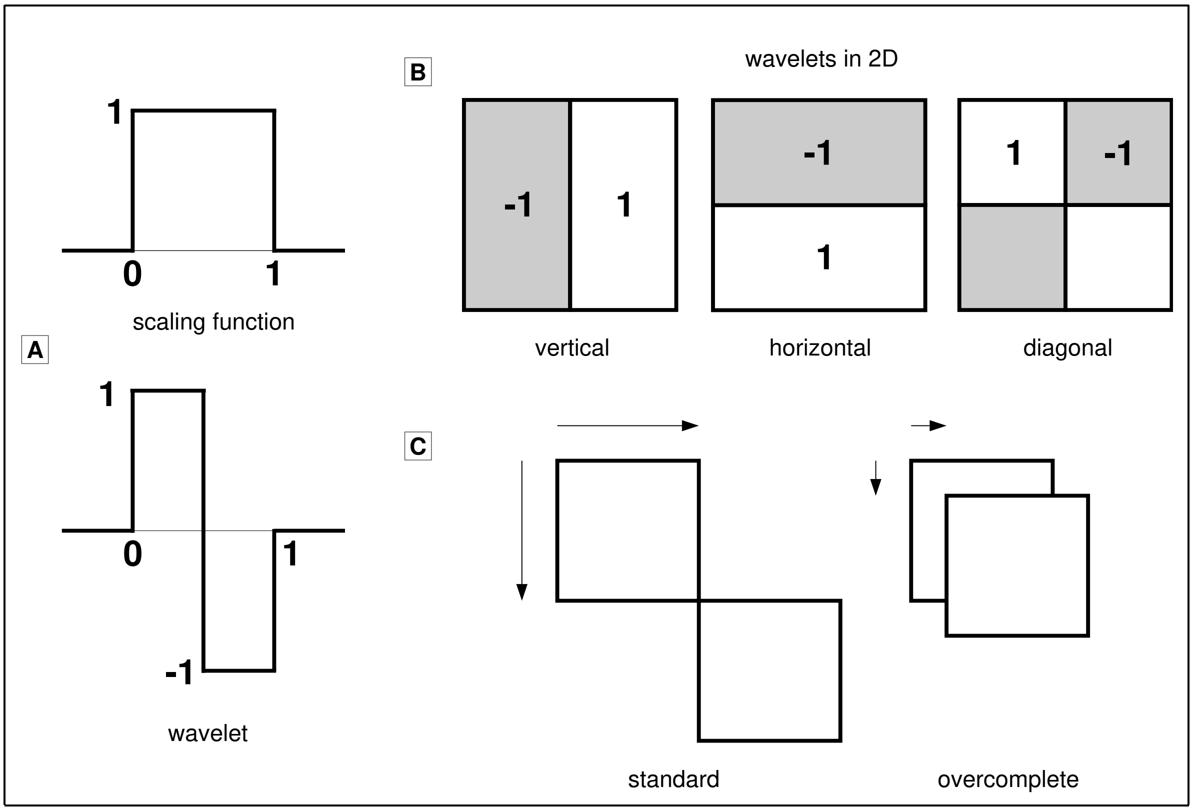


Figure 2-5: The Haar wavelet framework; (a) the Haar scaling function and wavelet, (b) the three types of 2-dimensional non-standard Haar wavelets: vertical, horizontal, and diagonal, and (c) the shift in the standard transform as compared to quadruply dense shift resulting in an overcomplete dictionary of wavelets.

of different parts of the patterns; a simple example makes it clear that this encoding does not capture the important features for detection. Take, for instance, our example of two data points of the same class where one is a dark body on a white background and the other is a white body on a dark background. With an intensity based representation (like pixels), each of these examples maps to completely different feature vectors. A representation that encodes local, oriented, intensity differences (like Haar wavelets) would yield similar feature vectors where the features corresponding to uniform regions are zero and those corresponding to boundaries are non-zero. In fact, since, in our representation, we encode only the magnitude of the intensity difference, the feature vectors for this simple two example case would be identical.

We do not use all the very fine scales of wavelets as features for learning since these scales capture high frequency details that do not characterize the class well. For instance, in the case of people, the finest scale wavelets may respond to checks, stripes, and other detail patterns, all of which are not features that are characteristic to the entire class. Similarly, the very coarse scale wavelets are not used as features

Figure 2-6: The top row shows examples of images of people in the training database. The bottom row show edge detection of the pedestrians. Edge information does not characterize the pedestrian class well.

for learning since their support will be as large as the object and will therefore not encode useful information. So, for the object detection system we have developed, we throw out the very fine and very coarse wavelets and only use two medium scales of wavelets as features for learning. These scales depend on the object class and the size of the training images and are chosen *a priori*.

2.3.2 The Wavelet Representation

The Haar transform provides a multiresolution representation of an image with wavelet features at different scales capturing different levels of detail. The coarse scale wavelets encode large regions while the fine scale wavelets describe smaller, local regions. The wavelet coefficients preserve all the information in the original image, but the coding of the visual information differs from the pixel-based representation in two significant ways.

First, the wavelets encode the difference in average intensity between local regions along different orientations in a multiscale framework. Constraints on the values of the wavelets can express visual features of the object class. Strong response from a particular wavelet indicates the presence of an intensity difference, or boundary, at that location in the image while weak response from a wavelet indicates a uniform area.

Second, the use of an overcomplete Haar basis allows us to propagate constraints between neighboring regions and describe complex patterns. The quadruple density wavelet transform provides high spatial resolution and results in a rich, overcomplete dictionary of features.

In the following sections, we show how our wavelet representation applies to faces, people, and cars. This coding of local intensity differences at several scales provides a flexible and expressive representation that can characterize complex object classes. Furthermore, the wavelet representation is computationally efficient for the task of object detection since we do not need to compute the transform for each image region that is examined, but only once for the whole image and then process the image in the space of wavelets.

Analyzing the Face Class

For the face class, we have a training set of 2,429 gray scale images of faces. This set consists of a core set of faces with some small angular rotations to improve generalization and 24,730 nonface patterns. These images are all scaled to the dimensions 19×19 and show the face from above the eyebrows to below the lips. Typical images from the database are shown in Figure 2-2. Databases of this size and composition have been used extensively in face detection [Sung, 1995, Rowley *et al.*, 1998, Osuna *et al.*, 1997a]. For the size of patterns our face system uses, we have at our disposal wavelets of the size 2×2, 4×4, 8×8, and 16×16. Instead of using the entire set of wavelets, we *a priori* limit the dictionary to contain the wavelets of scales 2×2 and 4×4, since coarser features do not contain significant information for detection purposes. At the scale 4×4 pixels, there are 17×17 features in quadruple density for each wavelet class, and at 2×2 pixels there are 17×17 features in double density for each class, for a total of 1,734 coefficients.

The raw value of a coefficient may not necessarily be indicative of a boundary; a weak coefficient in a relatively dark image may still indicate the presence of an intensity difference that is significant for the purposes of classification. To reduce these effects on the features used for classification, we normalize a coefficient's value against the other coefficients in the same area. For the normalization step, we compute the average of each wavelet's class $(\{vertical, horizontal, diagonal\} \times \{2, 4\})$ over the current pattern and divide the wavelet response at each spatial location by its corresponding class average. We calculate the averages separately for each class since the power distribution between the different classes may vary. For a given pattern p (in this case, a 19×19 pixel pattern), the class averages are:

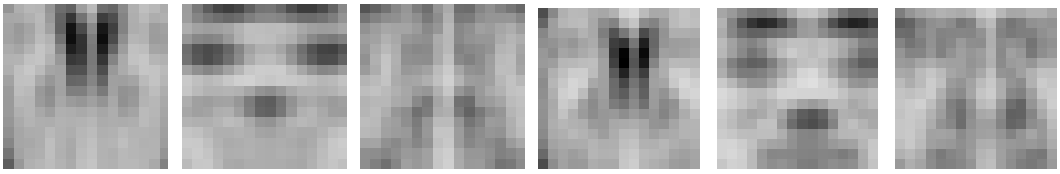
$$avg_{o,s} = \frac{1}{n} \sum_{i \in p} w_{o,s}[i] \tag{2.1}$$

where c denotes a fixed orientation, s denotes a fixed scale, i indexes into the wavelets in the pattern p, and $w_{o,s}$ denote the n individual wavelet coefficients at orientation o and scale s. The normalization for all wavelets within the pattern p is then:

$$w^*_{o,s}[i] = \frac{w_{o,s}[i]}{avg_{o,s}} \tag{2.2}$$

After the normalization, the average value of a coefficient for random patterns should be one. Three classes of feature magnitudes will emerge: ensemble average values much larger than one, which indicate strong intensity difference features that are consistent along all the examples; values that are much less than one, which indicate consistent uniform regions; and values that are close to one, which are associated with inconsistent features, or random patterns.

To visualize the detected face features, we code the ensemble average of the wavelet coefficients using gray level and draw them in their proper spatial layout (Figure 2-7). Coefficients with values close to one are plotted in gray; those with values larger than one are darker; and those with values less than one are lighter. It is interesting to observe the emerging patterns in the facial features. The vertical coefficients capture the sides of the nose, while the horizontal coefficients capture the eye sockets, eyebrows, and tip of the nose. Interestingly, the mouth is a relatively weak feature compared to the others. The diagonal coefficients respond strongly to the endpoint of facial features.



(a) (b) (c) (d) (e) (f)

Figure 2-7: Ensemble average values of the wavelet features of faces coded using gray level. Coefficients whose values are above the average are darker, those below the average are lighter. (a)-(c) are the vertical, horizontal, and diagonal wavelets at scale 4×4; (d)-(f) are the vertical, horizontal, and diagonal wavelets at scale 2×2.

1.05	1.42	1.73	1.97	2.07	1.97	1.70	1.48	1.46	1.61	1.89	2.08	2.08	1.93	1.65	1.27	1.12
1.04	1.30	1.45	1.55	1.62	1.56	1.37	1.26	1.26	1.30	1.44	1.56	1.56	1.50	1.36	1.15	1.09
0.86	0.98	1.03	1.00	0.95	0.82	0.69	0.69	0.72	0.71	0.79	0.92	1.01	1.04	1.03	0.94	0.94
1.10	1.27	1.31	1.27	1.08	0.83	0.70	0.67	0.70	0.77	0.85	1.05	1.26	1.38	1.42	1.33	1.28
1.54	1.81	1.91	1.83	1.61	1.29	0.99	0.84	0.83	0.96	1.20	1.53	1.81	1.99	2.01	1.82	1.72
1.50	1.78	1.90	1.78	1.54	1.24	0.90	0.73	0.72	0.82	1.12	1.45	1.72	1.91	1.89	1.67	1.56
0.99	1.19	1.30	1.20	1.00	0.80	0.59	0.54	0.55	0.54	0.71	0.94	1.14	1.28	1.24	1.06	0.97
0.57	0.68	0.75	0.70	0.60	0.51	0.48	0.56	0.60	0.56	0.58	0.62	0.70	0.77	0.74	0.64	0.62
0.60	0.73	0.81	0.81	0.79	0.83	0.96	1.15	1.21	1.08	0.92	0.82	0.82	0.85	0.81	0.73	0.70
0.86	1.01	1.04	0.99	0.98	1.11	1.39	1.69	1.73	1.48	1.16	0.96	0.90	0.99	1.06	1.01	0.95
0.93	1.01	0.97	0.86	0.84	1.02	1.35	1.64	1.68	1.45	1.11	0.84	0.79	0.92	1.04	1.03	0.99
0.80	0.83	0.85	0.79	0.71	0.75	0.93	1.12	1.15	0.99	0.81	0.75	0.80	0.87	0.87	0.84	0.81
0.62	0.66	0.76	0.85	0.85	0.82	0.85	0.96	0.98	0.90	0.87	0.90	0.91	0.82	0.70	0.63	0.61
0.56	0.56	0.68	0.82	0.89	0.87	0.84	0.87	0.89	0.86	0.90	0.95	0.89	0.73	0.59	0.54	0.56
0.61	0.54	0.62	0.77	0.85	0.85	0.83	0.86	0.88	0.85	0.87	0.91	0.85	0.71	0.59	0.57	0.64
0.72	0.58	0.58	0.74	0.90	0.92	0.87	0.87	0.88	0.87	0.91	0.93	0.83	0.68	0.61	0.67	0.79
0.44	0.35	0.32	0.36	0.43	0.47	0.47	0.50	0.51	0.48	0.46	0.44	0.38	0.35	0.35	0.41	0.47

Table 2.1: Ensemble average of normalized horizontal coefficients of scale 4×4 of images of faces. Meaningful coefficients are the ones with values much larger or smaller than 1. Average values close to 1 indicates no meaningful feature.

Analyzing the People Class

For learning the people class, we have collected a set of 1,800 color images of people in different poses (Figure 2-3) and use the 1,800 mirror images as well and 16,726 nonpeople patterns. All of the images are normalized to the dimensions 128×64 and the people images are aligned such that the bodies are centered and are approximately the same size (the distance from the shoulders to feet is about 80 pixels).

As in the case of faces, to code features at appropriate scales for people detection – scales at which we expect relevant features of people to emerge – we restrict the system to the wavelets at scales of 32×32 pixels (13×5 features for each orientation) and 16×16 pixels (29×13 features for each orientation).

In our people detection system, our training database is of color images. For a given pattern, we compute the quadruple density Haar transform in each color channel (RGB) separately and take, as the coefficient value at a specific location and orientation, the one largest in absolute value among the three channels. This technique maps the original color image to a pseudo-color channel that gives us 1,326 wavelet coefficients, the same number as if we had been using gray level images.

To visualize the patterns that emerge using this wavelet representation for people, we can code the average values of the coefficients in gray level and display them in the proper spatial layout as we did for the faces. Figure 2-8 shows each average wavelet displayed as a small square where features close to one are gray, stronger features are darker, and weaker features are lighter. As with faces, we observe that each class of wavelet coefficients is tuned to a different type of structural information. The vertical wavelets capture the sides of the people. The horizontal wavelets respond to the shoulders and to a weaker belt line. The diagonal wavelets are tuned to "corner features," i.e. the shoulders, hands, and feet. The 16×16 scale wavelets provide fine spatial resolution of the body's overall shape, and smaller scale details, such as the head and extremities, are clearly evident.

Analyzing the Car Class

The car detection system uses a database of 516 frontal and rear color images of cars, normalized to 128×128 and aligned such that the front or rear bumper is 64 pixels across. For training, we use the mirror images as well for a total of 1,032 positive patterns and 5,166 negative patterns. A few examples from our training database are shown in Figure 2-4. The two scales of wavelets we use for detection are 16×16 and 32×32. Like the processing for people, we collapse the three color channel features into a single channel by using the maximum wavelet response of each channel at a specific location, orientation, and scale. This gives us a total of 3,030 wavelet features that are used to train the SVM.

The average wavelet feature values are coded in gray level in Figure 2-9. The gray level coding of the average feature values shows that the wavelets respond to the significant visual characteristics of cars. The vertical wavelets respond to the sides

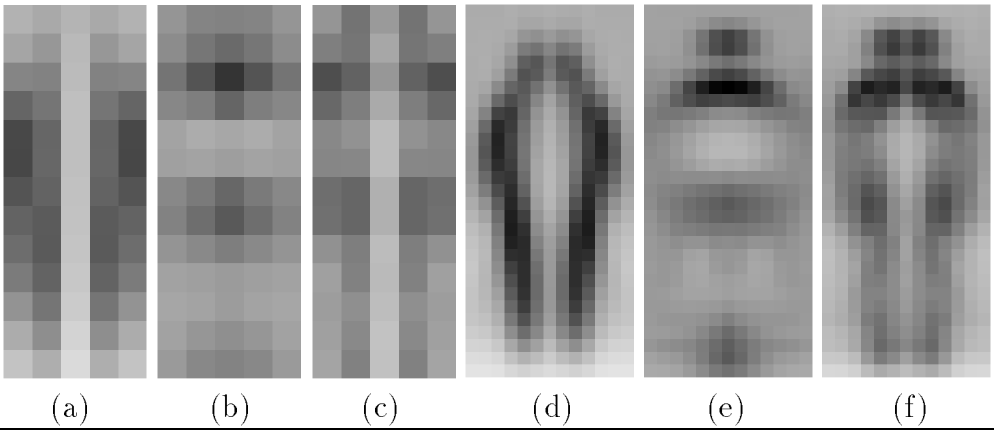


Figure 2-8: Ensemble average values of the wavelet features of people coded using gray level. Coefficients whose values are above the average are darker, those below the average are lighter; (a)-(c) are the vertical, horizontal, and diagonal wavelets at scale 32×32, (d)-(f) are the vertical, horizontal, and diagonal wavelets at scale 16×16.

of the car; the horizontal wavelets respond to the roof, underside, top of the grille and bumper area; and the diagonal wavelets respond to the corners of the car's body. At the scale 16×16, we can even see evidence of what seem to be license plate and headlight structures in the average responses.

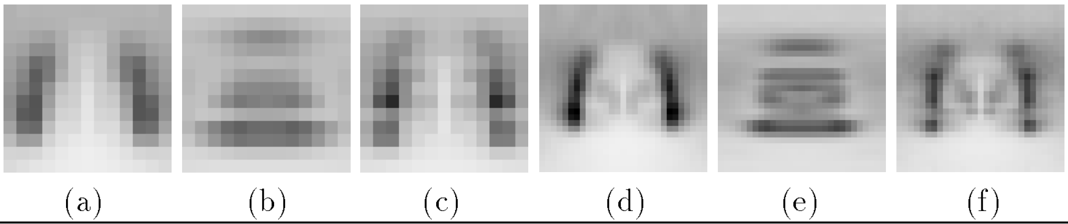


Figure 2-9: Ensemble average values of the wavelet features of cars coded using gray level. Coefficients whose values are above the average are darker, those below the average are lighter; (a)-(c) are the vertical, horizontal, and diagonal wavelets at scale 32×32, (d)-(f) are the vertical, horizontal, and diagonal wavelets at scale 16×16.

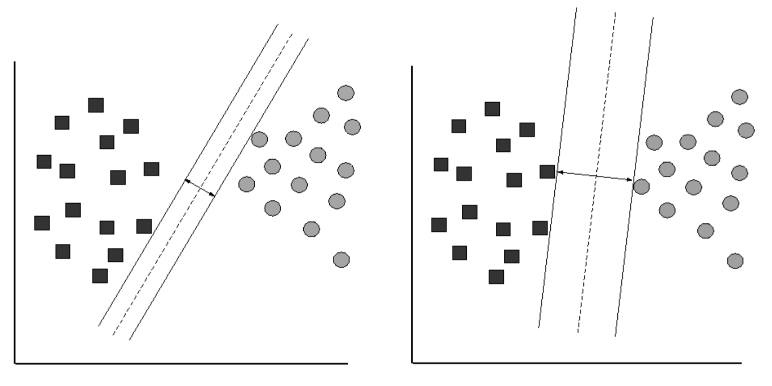
Discussion

Comparing the database of people, Figure 2-3, to the database of faces, Figure 2-2, illustrates an important fundamental difference in the two classes. In the case of faces, there are clear patterns within the face consisting of the eyes, nose and mouth. These patterns are common to all the examples. This is not the case with full-body images of people. The people do *not* share any common color or texture. Furthermore, the people images have a lot of spurious details such as jackets, ties, and bags. On the other hand, we would expect that people can be characterized quite well by their fairly similar overall body shape, or "silhouette." Our approach treats these two cases, where there is different underlying information content in the object classes, in a uniform manner. Frontal and rear views of cars have both a certain amount of common interior structure (top of grille, license plates, headlights), as well as fairly uniform outer boundaries. We will see that our wavelet representation is also suitable for car detection as well.

There is certain *a priori* knowledge embedded in our choice of the wavelets. The use of the absolute value of the coefficient is essential in the case of people since the direction of the intensity difference of a certain feature's orientation is not important; a dark body against a light background and a light body against a dark background should be represented as having the same information content. Furthermore, we compute the wavelet transform for a given pattern in each of the three color channels and then, for a wavelet at a specific location and orientation, we use the one that is largest in magnitude amongst the three channels. This is based on the observation that there is little consistency in color between different people and allows the system to key off of the most visually significant features.

Once we have generated the feature vectors for an object class and have done the same for a set of images not in our object class, we use a learning algorithm that learns to differentiate between the two classes. The particular learning engine we use is a support vector machine, described below.

2.4 Support Vector Machines



(a) small margin (b) large margin

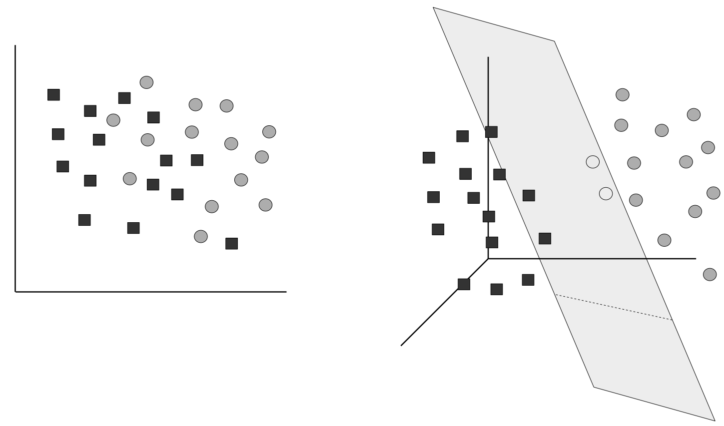
Figure 2-10: The separating hyperplane in (a) has small margin; the hyperplane in (b) has larger margin and should generalize better on out-of-sample data.

Support vector machines (SVM) is a technique to train classifiers that is well-founded in statistical learning theory [Vapnik, 1995, Burges, 1998]. One of the main attractions of using SVMs is that they are capable of learning in *high dimensional spaces* with very few training examples. SVMs accomplish this by minimizing a bound on the empirical error and the complexity of the classifier at the same time.

This concept is formalized in the theory of uniform convergence in probability:

$$R(\alpha) \leq R_{emp}(\alpha) + \Phi\left(\frac{h}{\ell}, \frac{-log(\eta)}{\ell}\right) \tag{2.3}$$

with probability $1 - \eta$. Here, $R(\alpha)$ is the expected risk; $R_{emp}(\alpha)$ is the empirical risk; ℓ is the number of training examples; h is the VC dimension of the classifier that is being used; and $\Phi(\cdot)$ is the VC confidence of the classifier. Intuitively, what this means is that the uniform deviation between the expected risk and empirical risk decreases with larger amounts of training data ℓ and increases with the VC dimension h. This leads us directly to the principle of structural risk minimization, whereby we can attempt to minimize at the same time both the actual error over the training set and the complexity of the classifier. This will bound the generalization error as in Equation 2.3. It is exactly this technique that support vector machines approximate.



(a) original data set (b) mapped feature space

Figure 2-11: The original data set may not be linearly separable. The support vector machine uses a nonlinear kernel to map the data points into a very high dimensional feature space in which the classes have a much greater chance of being linearly separable.

This controlling of both the training set error *and* the classifier's complexity has allowed support vector machines to be successfully applied to very high dimensional learning tasks; [Joachims, 1997] presents results on SVMs applied to a 10,000 dimensional text categorization problem and [Osuna *et al.*, 1997b] show a 283 dimensional face detection system.

The separating boundary is in general of the form:

$$f(\mathbf{x}) = \theta\left(\sum_{i=1}^{\ell} \alpha_i y_i K(\mathbf{x}, \mathbf{x}_i) + b\right) \tag{2.4}$$

where ℓ is the number of training data points $(\mathbf{x}_i, y_i)$ (y_i being the label ± 1 of training point $\mathbf{x}_i$); α_i are nonnegative parameters learned from the data; and $K(\cdot,\cdot)$ is a kernel that defines a dot product between projections of the two arguments in some feature space [Vapnik, 1998, Wahba, 1990] where a separating hyperplane is then found (Figure 2-11). For example, when $K(\mathbf{x}, \mathbf{y}) = \mathbf{x} \cdot \mathbf{y}$ is the chosen kernel, the separating surface is a hyperplane in the space of $\mathbf{x}$ (input space). The kernel $K(\mathbf{x}, \mathbf{y}) = \exp(-\|\mathbf{x} - \mathbf{y}\|^2)$ defines a Gaussian radial basis function [Girosi *et al.*, 1995] and $K(\mathbf{x}, \mathbf{y}) = (\mathbf{x} \cdot \mathbf{y} + 1)^n$ describes an n^{th} degree polynomial. In general, any positive definite function can be used as the kernel [Vapnik, 1998, Wahba, 1990].

The main feature of SVM is that it finds, among all possible separating surfaces of the form (2.4), the one which maximizes the distance between the two classes of points (as measured in the feature space defined by K). The support vectors are the nearest points to the separating boundary and are the only ones (typically a small fraction of the training data) for which α_i in Equation (2.4) is positive.

Using the SVM formulation, the classification step for a pattern $\mathbf{x}$ using a polynomial of degree two – the typical classifier we use for our system – is as follows:

$$f(\mathbf{x}) = \theta\left(\sum_{i=1}^{N_s} \alpha_i y_i (\mathbf{x} \cdot \mathbf{x}_i + 1)^2 + b\right) \tag{2.5}$$

where N_s is the number of support vectors, or training data points that define the decision boundary; α_i are Lagrange parameters; and θ is a threshold function. If we introduce $d(\mathbf{x})$, which returns a value proportional to the distance of $\mathbf{x}$ to the seperating hyperplane, then $f(\mathbf{x}) = \theta\left(d(\mathbf{x})\right)$.

In our case, the feature vector we use is composed of the wavelet coefficients for the pattern we are currently analyzing.

2.4.1 SVMs and Conditional Densities

The raw output of a single SVM classification, $d(\mathbf{x})$, is a real number that is proportional to the distance of the point $\mathbf{x}$ to the separating hyperplane. To facilitate comparisons between the outputs of different support vector machines and provide a probabilistic interpretation, it is necessary to normalize the outputs somehow; in their raw form, they are not directly comparable. Several methods for providing interpretations of the output of a SVM as a conditional density have emerged. While we do not use these methods in our system, they are outlined below for completeness.

Density estimation in the space of $d(\mathbf{x})$

In [Niyogi *et al.*, 1999], they describe a method for converting SVM outputs to probabilities that simply relies on the approximation $P(\mathbf{x}|y) \approx P(d(\mathbf{x})|y)$, thereby doing density estimation in the lower dimensional space of distances from the hyperplane rather than the full feature space. To estimate the posterior density:

$$P(y|\mathbf{x}) = \frac{P(d(\mathbf{x})|y)P(y)}{P(d(\mathbf{x})|y)P(y) + P(d(\mathbf{x})|\overline{y})P(\overline{y})} \tag{2.6}$$

Maximum likelihood fitting

In [Dumais *et al.*, 1998, Platt, 1999], they directly fit a sigmoid to the output of an SVM using a regularized maximum likelihood approach. The resulting posterior is:

$$P(y|\mathbf{x}) = \frac{1}{1 + e^{Ad(\mathbf{x})+B}} \tag{2.7}$$

Decomposition of feature space

Another method for estimating conditional probabilities from SVMs is presented in [Vapnik, 1998]. Here, the feature space is decomposed into 1) a direction orthogonal to the seperating hyperplane, and 2) the other dimensions. Each of these decompositions is parameterized seperately by t (a scaled version of $d(\mathbf{x})$) and $\mathbf{u}$, respectively, and the density along the orthogonal line is:

$$P(y|t, \mathbf{u}) = a_0(\mathbf{u}) + \sum_{i=1}^{N} a_n(\mathbf{u}) cos(it) \tag{2.8}$$

2.5 Experimental Results

An algorithmic summary of the out-of-sample detection process is shown in Figure 2-12.

In Figures 2-13, 2-14, and 2-15, we present examples of our trainable object detection system as applied to the domains of face, people, and car detection, respectively. We reiterate that the system makes no *a priori* assumption on the scene structure or the number of objects present and does not use any motion or other dynamical information. The performance of each of these particular instantiations of detection systems could easily be improved by using more training data. We have not sought to push the limits of performance in particular domains; rather, our goal has been to show that this uniform architecture for object detection leads to high performance in several domains.

```
Let $I$ be the image on which we are running the detection system.
Let $s_r$ be the rescaling factors for our image; we use $s_r = 1.1$.
Let $s_i$ be the initial scale; we use $s_i = 0.2$.
Let $s_f$ be the final scale; we use $s_f = 1.5$.
Let $s_c$ be the current scale of image we are processing.
Let $I_c$ be the input image scaled by $s_c$.
Let $H(I)$ be the Haar transform of image $I$.
Let $q$ be a $128 \times 64$ pattern in wavelet space.
Let $fv$ be the feature vector used to classify a pattern.

1.    set $s_c = s_i$
2.    while $s_c \neq s_f$ then do
3.          $I_c \leftarrow$ resize $I$ by $s_c$
4.          $H(I_c) \leftarrow$ compute the Haar wavelet transform of $I_c$
5.          loop over all rows and columns $(r, c)$ in $H(I_c)$
6.                $q \leftarrow 128 \times 64$ pattern at $(r, c)$
7.                compute the average response of each type of wavelet
                        $scale = \{16, 32\} \times orientation = \{V, H, D\}$ in pattern $q$
7.                $fv \leftarrow$ normalize each wavelet in $q$ by its class average
8.                $class \leftarrow$ classify $fv$ using the SVM classifier
9.                if $(class == 1)$ then
10.                     pattern $q$ is a person so draw a rectangle around $q$
11.               if $(class == -1)$ then
12.                     pattern $q$ is not a person so ignore
13.         $s_c \leftarrow s_c \times s_r$
14.   end
```

Figure 2-12: Algorithm for detecting people in out-of-sample images.

Figure 2-13: Results of our face detection system on a set of out-of-sample images. A, C, E, F, G, H, I, J, K, L, M, N are from the test database of Sung and Poggio; B, D are from www.starwars.com; O is from www.corbis.com. Missed faces (B, F, I, J, K, M) are due to significant head rotations that were not present in the training data. False positives (E, F, N) are due to insufficient training data and can be eliminated by using more negative training data. The face detection system processes approximately 125,000 patterns per image.

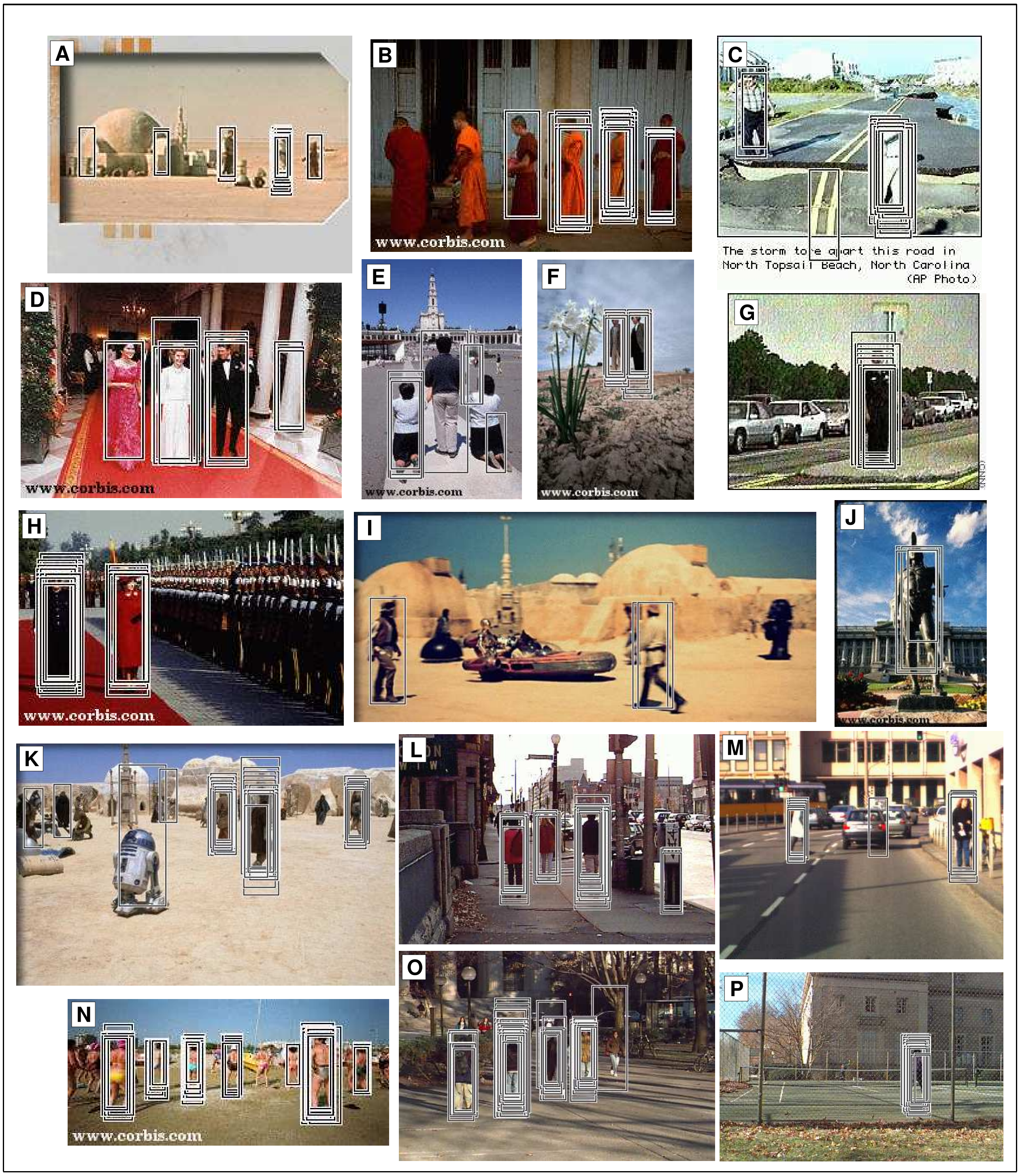


Figure 2-14: Results of people detection on out-of-sample images. A, I, K are from www.starwars.com; B, D, E, F, H, J, N are from www.corbis.com; C, G are from www.cnn.com; L, O, P were taken in Boston and Cambridge; M was provided by DaimlerChrysler. Missed detections are due to the person being too close to the edge of the image (B) or when the person has a uncharacteristic body shape not represented in the training data (I). False positives often look very similar to people (A) or are due to the presence of strong vertical and horizontal intensity differences (D, E, K, L, M, O). The people detection system processes approximately 35,000 patterns per image.

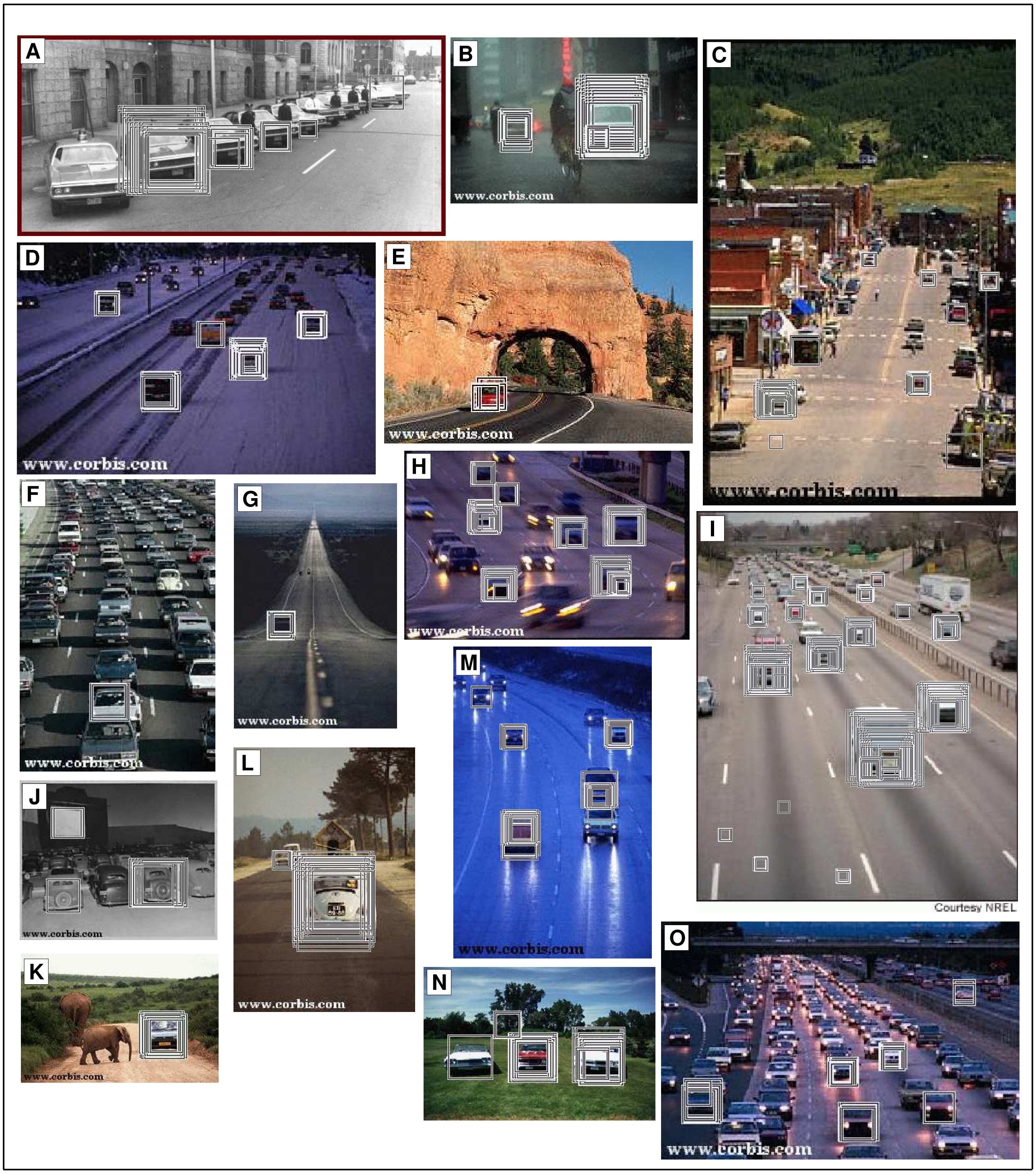


Figure 2-15: Results of car detection on out-of-sample images. A is from www.lewistonpd.com; B, C, D, E, F, G, H, J, K, L, M, O are from www.corbis.com; I is from www.enn.com; N is from www.foxglove.com. Missed positive examples are due to occlusions (A, F, O) or where a car is too close to the edge of the image (A). False positives (C, J, I, N) are due to insufficient training and can be eliminated with more negative training patterns. The car detection system processes approximately 125,000 patterns per image.

Chapter 3

Experimental Results

Our criteria for the representation we are using is that it be efficiently computable and identify local, oriented intensity difference features. Haar wavelets satisfy these criteria and are perhaps some of the simplest such features with finite support. Thusfar, we have focused on the Haar wavelet representation but could have considered other possible representations including pixels and principal components. This chapter empirically quantifies the added value in using the wavelet representation as compared with several alternate representations. We also present the result of experiments that compare the effect of using different types of classifiers including linear, polynomial, and radial basis function classifiers. Finally, we provide empirical evidence that shows that relatively few training examples may be needed to sufficiently train an object detection system.

3.1 Test Procedure

To accurately measure our detection system's performance, we use a receiver operating characteristic (ROC) curve which quantifies the tradeoff between detection accuracy and the rate of false positives. In all of our ROC curves, the y scale is the percentage of correct positive detections and the x scale is the rate of false positives measured as the number of false positives per negative pattern processed.

To test our detection system, we have developed an automated testing package that gives us a detailed view of how well our system is performing. The testing procedure measures positive and negative pattern performance separately over different sets of data. The positive test data consists of aligned, but not scaled, examples of the object class with a sufficiently large boundary around them such that the detection system can run at several different locations and scales. While these images can be of various sizes, the proportions at which the object occurs in the image is constant from test example to test example. This allows us to check whether or not each detection in the image falls on the actual object. We allow tolerances of a few pixels in the detections; exact tolerances are given in Table 3.1.

To generate the false positive rate, we have a set of 50 images of different natural and man-made scenes that do not contain any examples of the objects we are

	Faces	People	Cars
Tolerance	+/- 3	+/- 8	+/- 10

Table 3.1: Pixel tolerances for the automated testing procedure; given tolerances are for both x and y positions as normalized to the sizes of the objects in the training set, i.e. for faces, tolerances are scaled for 19×19 patterns, for people 128×64, and for cars 128×128.

detecting. The false positive rate is simply the number of detections in this set of data divided by the total number of patterns that are examined. Using the actual backgrounds of images containing people may give a more accurate false positive rate but we ignore this issue, opting for the most simple method of determining the false positive rate. Table 3.2 gives the number of positive examples and negative patterns for each of the object detection systems we develop in this thesis.

	Faces	People	Cars
Positive Examples	105	123	90
Negative Patterns	3,909,200	794,906	600,272

Table 3.2: Summary of the test set sizes for each of the detection systems.

This technique gives us a single detection/false positive point. To generate a full ROC curve, we shift the SVM decision surface and obtain detection/false positive points for various shifts. Shifting the decision surface has the effect of tuning the system to be more strict or more relaxed in its definition of what is and is not an element of the object class. The shifting is accomplished as

$$f(\mathbf{x}) = \theta\left(\sum_{i=1}^{\ell} \alpha_i y_i K(\mathbf{x}, \mathbf{x}_i) + b + s\right) \tag{3.1}$$

where s dictates the magnitude and direction of the shift as shown in Figure 3-1. For $s = 0$ we have the decision surface that SVM training providesl $s > 0$ moves the decision surface towards the positive class making classification more strict; and $s < 0$ moves the decision surface towards the negative class making classification more lenient.

3.2 Experiments

The dense Haar transform captures a rich set of features that allow the SVM classifier to obtain a powerful class model; the wavelets respond to significant visual features while smoothing away noise. This choice of features is *a priori*, however; this section presents the results of many tests comparing different features for object detection.

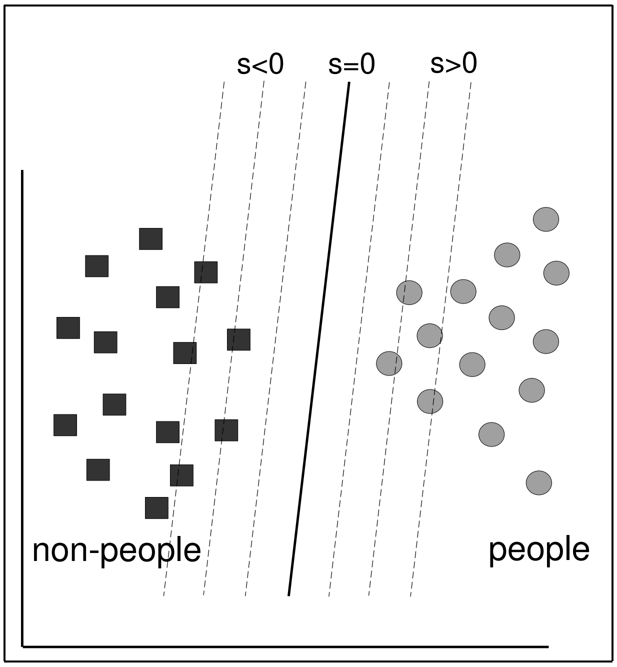


Figure 3-1: Shifting the original decision surface $s = 0$ by changing the bias term makes classification more strict if $s > 0$ or more lenient if $s < 0$. Each shifted decision boundary yields a detection system with different performance characteristics. We generate ROC curves by plotting the performance obtained for each shifted decision surface.

There are many possible alternate representations that have been used in the literature, including pixels and PCA, and these different representations are compared in our detection framework. Another decision we made was to ignore the sign of the wavelets and use their absolute value, tested against the signed values. In addition, for people detection, our training set is in color. We empirically quantify the improvement in performance using color data as opposed to gray level data.

In the results presented in this section, our people detection system is trained on 1,848 positive patterns (924 frontal and rear people images and their mirror images) and 11,361 non-people patterns, and is tested on 123 images containing people and 794,906 non-people patterns. The face detection system is trained on 2,429 face images and 13,229 non-face patterns, and is tested on 105 images containing faces and 3,909,200 non-face patterns. The car detection system is trained on 1,032 frontal and rear color images of cars (516 examples and their mirrors) and 5,166 non-car patterns, and is tested on 90 images containing cars and 600,272 non-car patterns.

Figure 3-2: Comparing histogram equalized pixels to histogram equalized scaling coefficients; (a) raw pixels (128×64), (b) histogram equalized pixels, and (c) histogram equalized overlapping 8×8 scaling coefficients (61×29).

3.2.1 Pixels, Wavelets, PCA

Our main premise for choosing a wavelet based representation is that intensity differences between local adjacent regions contain higher quality information for the purpose of object detection than other traditional representations. Pixel representations capture the "most local" features. These have been used extensively for face detection but due to the variability in the people patterns, we would expect pixel representations to fail for people detection. At the other end of the locality spectrum are global representations like PCA which encodes a class in terms of basis functions that account for the variance in the data set. We can change the class of features to see which yields the best performance. For people, we use the 1,769 overlapping 8×8 averages instead of pixels for a more fair comparison that uses similar numbers of features. Furthermore, these averages are histogram equalized in the same manner as the pixel representation (see Figure 3-2).

3.2.2 Signed vs. Unsigned Wavelets

The features our system uses do not contain information on the sign of the intensity gradient but are the absolute values of the wavelet responses. With these features, we are solely describing the strength of the intensity differences. For an object class like people, where a dark body on a light background has the same information as a light body on a dark background and there is little consistency in the intensities, the sign of the gradient should not matter. On the other hand, if we consider face patterns, there is consistent information in the sign of the gradient of the intensity differences. For instance, the eyes are darker than the cheeks and the forehead, and the mouth is darker than the cheeks and the chin. These types of relationships have been explored in [Sinha, 1994b]. We might expect that using the sign information ($+$ or $-$) would enhance results in this case.

3.2.3 Complete vs. Overcomplete

The motivation for using the overcomplete Haar wavelet representation is to provide a richer set of features over which the system will learn and, ultimately, a more accurate description of a person. We test this against the standard complete Haar representation[1].

3.2.4 Color vs. Gray Level

For color images in the case of people detection, we collapse information from the three color channels into a single pseudo-channel that maintains the strongest local intensity differences. It is intuitively obvious that color images contain much richer information than the corresponding gray-scale versions. We present experiments that quantify the inherent information content in using color images as opposed to gray level for object detection.

3.2.5 Faces, People, and Cars

Our ROC curves highlight the performance of the detection system as accuracy over out-of-sample data against the rate of false positives, measured as the number of false positives per pattern examined. The ROC curves that compare different representations for the face detection system are shown in Figure 3-3. The representations used for face detection are raw pixels (361 features); histogram equalized pixels (361 features); principal components of histogram equalized pixels (361 features); gray signed wavelets (1,740 features); and gray unsigned wavelets (1,740 features). Gray unsigned

[1] The wavelets that we use actually form an undercomplete basis for this space. A more correct characterization of the representation is that the features are the wavelets from the complete basis that are at the two scales (32×32 and 16×16) we are using.

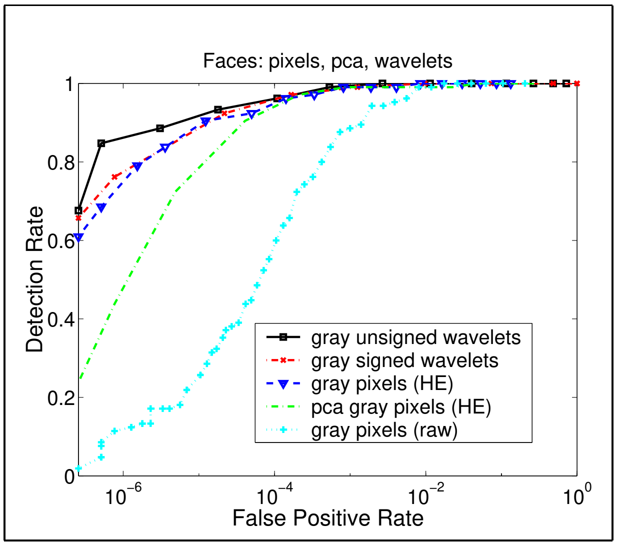


Figure 3-3: ROC curves for face detection comparing different features using pixel features as a benchmark. The graph shows gray unsigned and signed wavelets, raw and histogram equalized pixels, and PCA of histogram equalized pixels. The face detection system typically processes about 125,000 patterns per image.

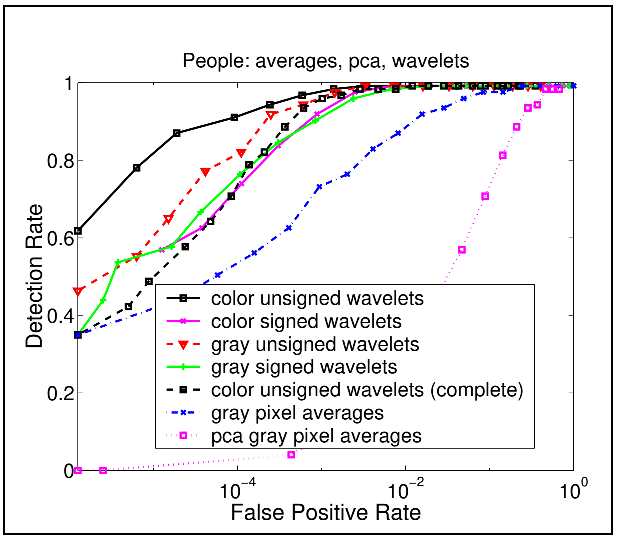


Figure 3-4: ROC curves for people detection comparing different features using pixel type features as a benchmark. The graph shows color and gray, signed and unsigned wavelets (overlapping), color unsigned wavelets (non-overlapping), 8×8 overlapping pixel averages, and PCA of the 8×8 overlapping pixel averages. We compare the 1,326 wavelets against the 1,769 overlapping 8×8 pixel averages instead of the 8,192 pixels themselves to use approximately the same number of features for a more fair comparison. The people detection system typically processes about 35,000 patterns per image.

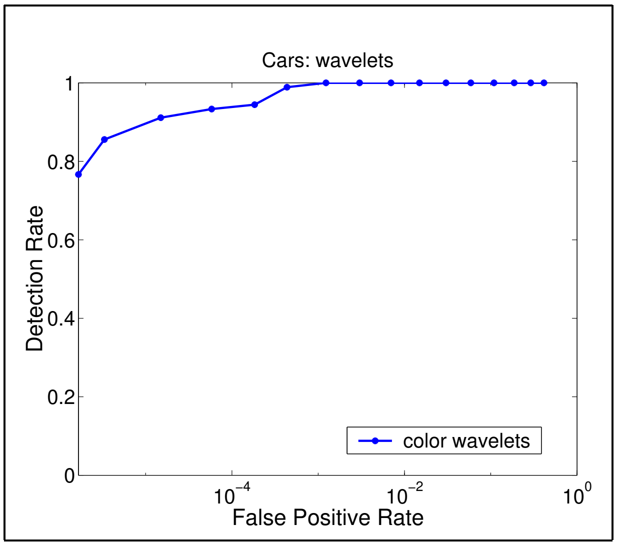


Figure 3-5: Preliminary ROC curve for car detection using wavelet features over color images. The car detection system typically processes about 125,000 patterns per image.

wavelets yield the best performance while gray signed wavelets and histogram equalized gray pixels lead to the same level of performance (slightly worse than the gray unsigned wavelets). The version using principal components is less accurate than the histogram equalized pixels. That the unsigned wavelets perform better than the signed wavelets is somewhat counterintuitive; we had postulated that the signs of the wavelets contain important information for face detection since human faces have consistent patterns. Using the absolute magnitude of the wavelets may result in a representation with less variability than the signed version while still encoding the important information for detection, allowing the classifier to find a better decision surface. To gauge the performance of the system, we can take a point on the ROC curve and translate the performance into real image terms. For instance, for a 90% detection rate, we must tolerate 1 false positive for every 100,000 patterns processed, or approximately 1 false positive per image.

The ROC curves for the people detection system are shown in Figure 3-4. Here, using all the color features performs the best, where, for instance, a 90% detection rate leads to one false positive for every 10,000 patterns that are processed (about three false positives per image). Gray level wavelets perform significantly better than the corresponding gray level averages. Here, unlike in the case of face detection, the raw pixel values do not characterize the object class well. When we use the 1,769 PCAs of the 8×8 averages the performance is significantly worse. Figure 3-4 also supports our hypothesis on the necessity of an overcomplete versus a complete representation. The system starting from a complete representation (120 color wavelets) underperforms all of the systems based on the overcomplete representation. The signed versions of both the color and gray level wavelets perform worse than their unsigned versions. We hypothesize that the reason is the same as the case for faces: that the unsigned versions result in more compact representations over which it is easier to learn (see the intuition given in Section 2.3.1).

The preliminary ROC curve for our car detection system using unsigned wavelet features on color images is shown in Figure 3-5.

3.2.6 Discussion

There is a simple relation between linear transformations of the original images and kernels that bears mentioning. An image $\mathbf{x}$ can be linearly decomposed into a set of features $\mathbf{c} = c_1, \ldots, c_m$ by $\mathbf{c} = A\mathbf{x}$, where A is a real matrix (we can think of the features $\mathbf{c}$ as the result of applying a set of linear filters to the image $\mathbf{x}$).

If the kernel used is a polynomial of degree m^2 as in the experiments, then $K(\mathbf{x}_i, \mathbf{x}_j) = (1 + \mathbf{x}_i^\top \cdot \mathbf{x}_j)^m$, while $K(\mathbf{c}_i, \mathbf{c}_j) = (1 + \mathbf{c}_i^\top \cdot \mathbf{c}_j)^m = (1 + \mathbf{x}_i^\top (A^\top A)\mathbf{x}_j)^m$. So using a polynomial kernel in the "$\mathbf{c}$" representation is the same as using a kernel $(1 + \mathbf{x}_i^\top (A^\top A)\mathbf{x}_j)^m$ in the original one. This implies that one can consider any linear transformation of the original images by choosing the appropriate square matrix $A^T A$ in the kernel K of the SVM.

As a consequence of this observation, we have a theoretical justification of why the pixel and eigenvector representations should lead to the same performance. In the case of using the PCA, the matrix A is orthonormal, therefore $A^T A = I$ which implies that the SVM should find the same solution in both cases. On the other hand, if we choose only some of the principal components, or if we project the images onto a non-orthonormal set of Haar wavelets, the matrix A is no longer orthonormal so the performance of the SVM may be different.

This theoretical justification is empirically validated, or at least not contradicted, in the case of faces, where the PCA and pixel representations perform at about the same level. However, for people, our results seem to contradict the theory: the PCA of the local averages perform much worse than the local averages themselves. Why might this be happening?

Closer analysis reveals that the pixel and principal component representations do not in fact lead to identical solutions. In the case of pixels, our polynomial kernel has the form:

$$K(\mathbf{x}_i, \mathbf{x}_j) = (1 + \mathbf{x}_i^\top \cdot \mathbf{x}_j)^m \tag{3.2}$$

In the case of PCA, we actually compute the eigenvectors over the set of *mean normalized* pixel images $(\mathbf{x} - \bar{\mathbf{x}})$, yielding as the kernel:

$$K(\mathbf{c}_i, \mathbf{c}_j) = (1 + \mathbf{c}_i^\top \cdot \mathbf{c}_j)^m = (1 + (\mathbf{x}_i - \bar{\mathbf{x}})^\top (A^\top A)(\mathbf{x}_j - \bar{\mathbf{x}}))^m \tag{3.3}$$

[2]Generally this holds for any kernel for which only dot products between input arguments are needed - i.e. also for Radial Basis Functions.

where now $\mathbf{c} = A(\mathbf{x} - \bar{\mathbf{x}})$ are computed from the mean normalized pixel features. Since A is orthonormal, in the case of PCA we are actually computing:

$$K(\mathbf{c}_i, \mathbf{c}_j) = (1 + \mathbf{c}_i^\top \cdot \mathbf{c}_j)^m = (1 + (\mathbf{x}_i - \bar{\mathbf{x}})^\top \cdot (\mathbf{x}_j - \bar{\mathbf{x}}))^m \tag{3.4}$$

which is strictly not equivalent to the kernel in the case of pixels. Further analysis of these issues is an area of future research.

The number of support vectors in the different solutions is one indicator of how difficult the individual problems are. Table 3.3 lists the number of support vectors for each of the different representations we have considered above. One of the most important conclusions we can draw from this table is that the signed representations do indeed result in more complex SVM decision surfaces than their unsigned counterparts, meaning that the signed representations are not as conducive to learning.

	Faces	People	Cars
pixels (raw)	521	-	-
pixels (histogram equalized)	239	853	-
pca	577	202	-
gray signed	1,006	1,639	-
gray unsigned	668	703	-
color signed	-	1,803	-
color unsigned	-	547	396

Table 3.3: Number of support vectors in each of the solutions to the classification problems using different representations.

3.3 Different Classifiers

As we stated in Section 2.4, our main motivation for using classifiers trained by the support vector machine algorithm is its ability to find separating hyperplanes in sparsely populated high dimensional feature spaces that do not overfit. Until now, we have ignored exactly what form the classifier has and have described the system in the context of using a polynomial classifier of degree two. In this section, we present further experiments where several different types of classifiers are used and we compare the results over an identical test set.

The general form of the decision surface found through SVM training is:

$$f(\mathbf{x}) = \theta\left(\sum_{i=1}^{\ell} \alpha_i y_i K(\mathbf{x}, \mathbf{x}_i) + b\right) \tag{3.5}$$

where ℓ is the number of training data points $(\mathbf{x}_i, y_i)$ (y_i being the label ± 1 of training point $\mathbf{x}_i$); α_i are nonnegative parameters learned from the data; and $K(\cdot, \cdot)$ is the

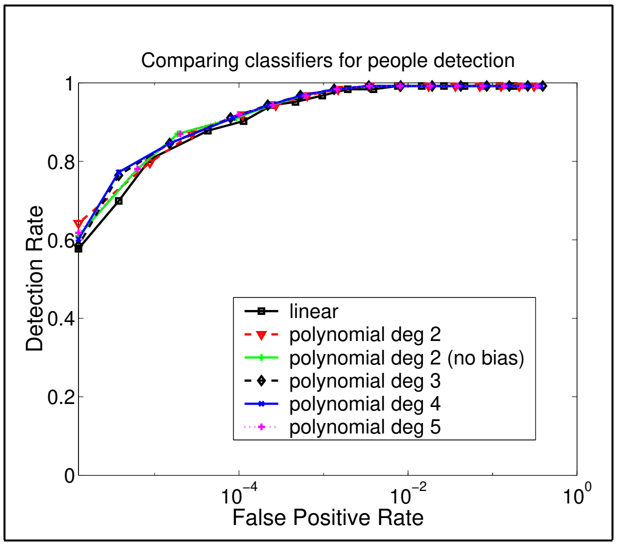


Figure 3-6: ROC curves for people detection with the full 1,326 feature set comparing different classifiers: linear and polynomial classifiers of degree 2 through 5, including a polynomial classifier of degree 2 with no bias term. There is no appreciable difference in the performance using these different classifiers.

kernel that defines a dot product between projections of the two arguments in some feature space. By choosing different kernels $K(\cdot,\cdot)$, we define qualitatively different decision surfaces. For instance, $K = (\mathbf{x}\cdot\mathbf{y}+1)^n$ defines an n^{th} degree polynomial and $K(\mathbf{x},\mathbf{y}) = \exp(-\|\mathbf{x}-\mathbf{y}\|^2)$ defines a Gaussian radial basis function.

Here, we consider polynomial classifiers of varying degree n, from a perceptron ($n = 1$) to a 5^{th} degree polynomial; several attempts at using Gaussian RBF classifier would not converge in SVM training. Figure 3-6 shows the performance of each of these systems. From the ROC curves, it is immediately evident that there is no appreciable difference in any of the classifiers. This has important practical ramifications; with a linear classifier, the decision surface can be described as:

$$f(\mathbf{x}) = \theta\left(\sum_{i=1}^{\ell} \alpha_i y_i (\mathbf{x}\cdot\mathbf{x}_i) + b\right) \tag{3.6}$$

but, since the α_i, y_i, and $\mathbf{x}_i$ are all predetermined at runtime, we can write the decision function in its more general form:

$$f(\mathbf{x}) = \theta\left(\mathbf{w}\cdot\mathbf{x} + b\right) \tag{3.7}$$

since $w = \sum_{i=1}^{\ell} \alpha_i y_i$. This means that we can classify a point with a single dot product.

3.4 Feature Selection

The goal of feature selection is to identify which features are important for detection and subsequently use this (sub)set of features as the representation over which learning occurs. Until now, we have described systems with no feature selection; after deciding on a feature class (pixels or wavelets or PCA), we use all of the features of that class. Generally, we pay a price for large sets of features in the form of expensive computation. Moreover, it may be the case that certain features are not important for detection. For instance, the wavelets in the background portion of the people patterns are not good indicators of the presence of a person. It may be possible to reduce the dimensionality of the representation through a feature selection step while still preserving most of the system's representational power.

To find the optimal set of features, we would need to determine the performance of our system using every possible subset of features; for the wavelet representation, this would mean checking the $\sum_{s=1}^{1,326} \begin{pmatrix} 1,326 \\ s \end{pmatrix}$ unique subsets of features, an astronomical number. For a given size s, the selection of the best s features amounts to an integer programming problem which is NP-complete.

We have implemented a manual feature selection technique that results in much lower dimensional data, and therefore less expensive classification, while still maintaining a reasonable level of accuracy.

To do manual feature selection, we start with the tables of the average wavelet responses for our training set broken down by scale and orientation. From these tables we manually choose the strongest and weakest features, while at the same time, we ensure that the features we choose are not overly redundant, i.e. do not overlap. This process is fairly subjective but, at the same time, allows for the enforcement of criteria such as picking strong features that span the space in some greedy manner, rather than just picking the n strongest features, some of which may overlap significantly and therefore not provide improvement.

For the people detection case, the tables of raw averages are shown in Appendix C. The 29 manually chosen features are shown overlayed on an example person in Figure 3-7. The performance of this 29 feature system is shown in Figure 3-8. While it underperforms the versions using the full feature sets (1,326 wavelets), the system is able to capture much of the structure of the human form using just 29 local features. The resulting performance may be acceptable for certain applications.

3.5 Training with Small Data Sets

Most example-based systems for object detection have relied on large sets of positive and negative examples that are used to train a system to differentiate between the target class and the non-target class. Table 3.4 enumerates that number of positive and negative examples used to train different detection systems reported on in the

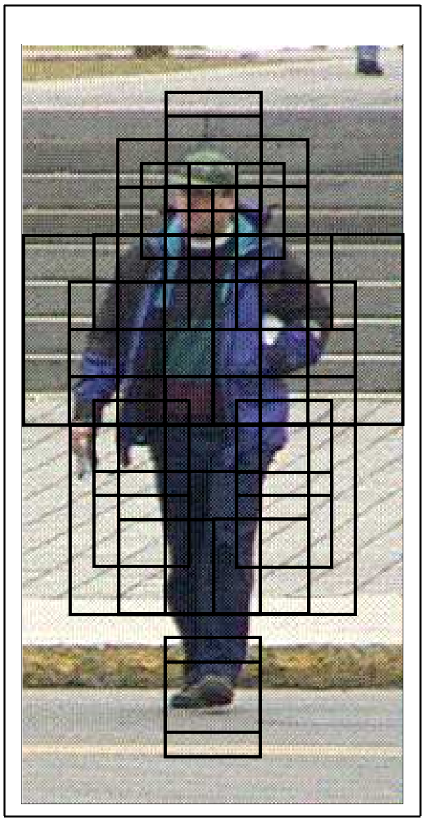


Figure 3-7: The reduced set of 29 manually chosen wavelet features for fast people detection overlayed on an example image of a person.

literature.

Gathering positive examples of an object class is an expensive, tedious task; all these systems require input data where the objects are aligned in the same position in the image and the images are scaled to the same size. Typically, we have cheap access to an unlimited number of negative examples, while obtaining a large number positive examples is relatively expensive. In our domain of people detection, we invested significant effort in gathering a large number of positive examples. Furthermore, the domains for object detection that people have tackled are well-defined and easily accessible. While it is time consuming to gather more positive examples, there are no inherent limitations in doing so; we know what a face or person looks like, so we simply find or take pictures of examples from these classes. We bring to bear much prior knowledge in engineering object detection systems for these domains.

What if our detection problem was such that we only had information about a small number of elements of the positive class? Let us also assume that we have *no* prior experience or information about this detection problem. This is not too unbelievable a situation; consider a hypothetical task of detecting certain rare anomalies in images taken with an electron microscope. Taking this one step further, what if

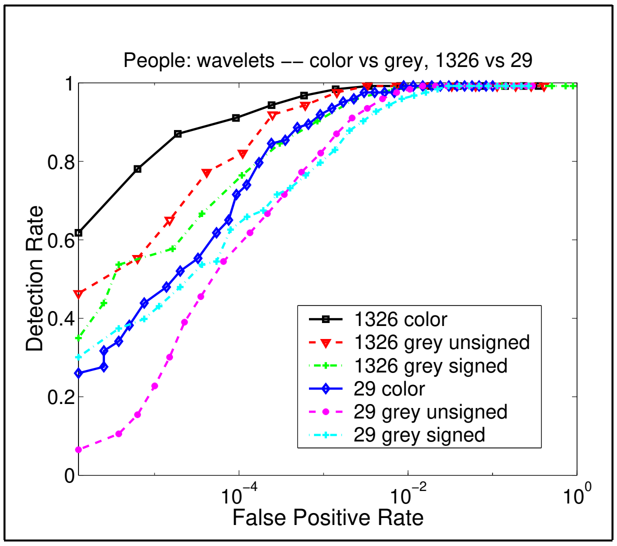


Figure 3-8: ROC curves for people detection comparing different wavelet features and different feature set sizes.

Researchers	Domain	Positive Examples	Negative Examples
Vaillant, Monrocq, Le Cun	Faces	1,792	1,792
Sung and Poggio	Faces	4,150 (1,067 base)	43,166
Rowley, Baluja, Kanade	Faces	15,750 (1,050 base)	9,000
Moghaddam and Pentland	Faces	7,562	NA
Schneiderman and Kanade	Faces	119,911 (991 base)	1,552
Papageorgiou	Faces	2,249	24,730
Papageorgiou	People	3,600 (1,800 base)	16,726

Table 3.4: Training set sizes for several object detection system reported on in the literature.

it is also the case that our knowledge of negative examples is severely restricted, as well?

One of the main attractions of the SVM framework is that it controls both the training error and the complexity of the decision classifier at the same time. This can be contrasted with other training techniques, like back propagation, that only minimize training error. Since there is no controlling of the classifier complexity, this type of system will tend to overfit the data and provide poor generalization.

In practical terms, this means that SVMs can find good solutions to classification problems in very high dimensions. In addition to being a theoretically sound property, this capability has been demonstrated empirically in the literature in face detection [Osuna *et al.*, 1997b], text categorization [Joachims, 1997], and people detection [Papageorgiou *et al.*, 1998]. All of these systems and other object detection systems [Sung and Poggio, 1998, Rowley *et al.*, 1998] use a large set of positive examples in addition to a large set of negative examples.

Figure 3-9 quantifies the performance of our 1,326 wavelet feature, color people detection system when the positive training set size is varied as 1, 10, 100, and the full set of 1,848 people (924 plus mirror images) and the negative training set size is varied as 10, 100, and the full set of 11,361 non-people patterns. The size 1, 10, and 100 training set experiments were each run 10 times with randomly chosen data points; the ROC curves report the average performance.

If we assume that we have unlimited access to negative examples as represented by Figure 3-9a, we can see that with just one positive example, the system performs extremely well. Using the fully trained system's benchmark of one false positive per 10,000 patterns with a 90% detection rate, the systems trained with one positive example correctly detect 50% of the people. Increasing the number of positive examples to 10 results in an 80% detection rate.

With limited access to negative examples, Figures 3-9b and c show that the system still performs very well. For as few as 10 negative training examples, the system reaches 75% accuracy in both the 10 and 100 positive training versions, for the rate of one false positive per 10,000 patterns.

This leads us to believe that, for the domain of people detection, the choice of the representation we use is more important than gathering large sets of training data. In our case, the transformation from pixels to wavelets compresses the image information into a model that seems to be quite compact, so that even a single positive training example characterizes the class well.

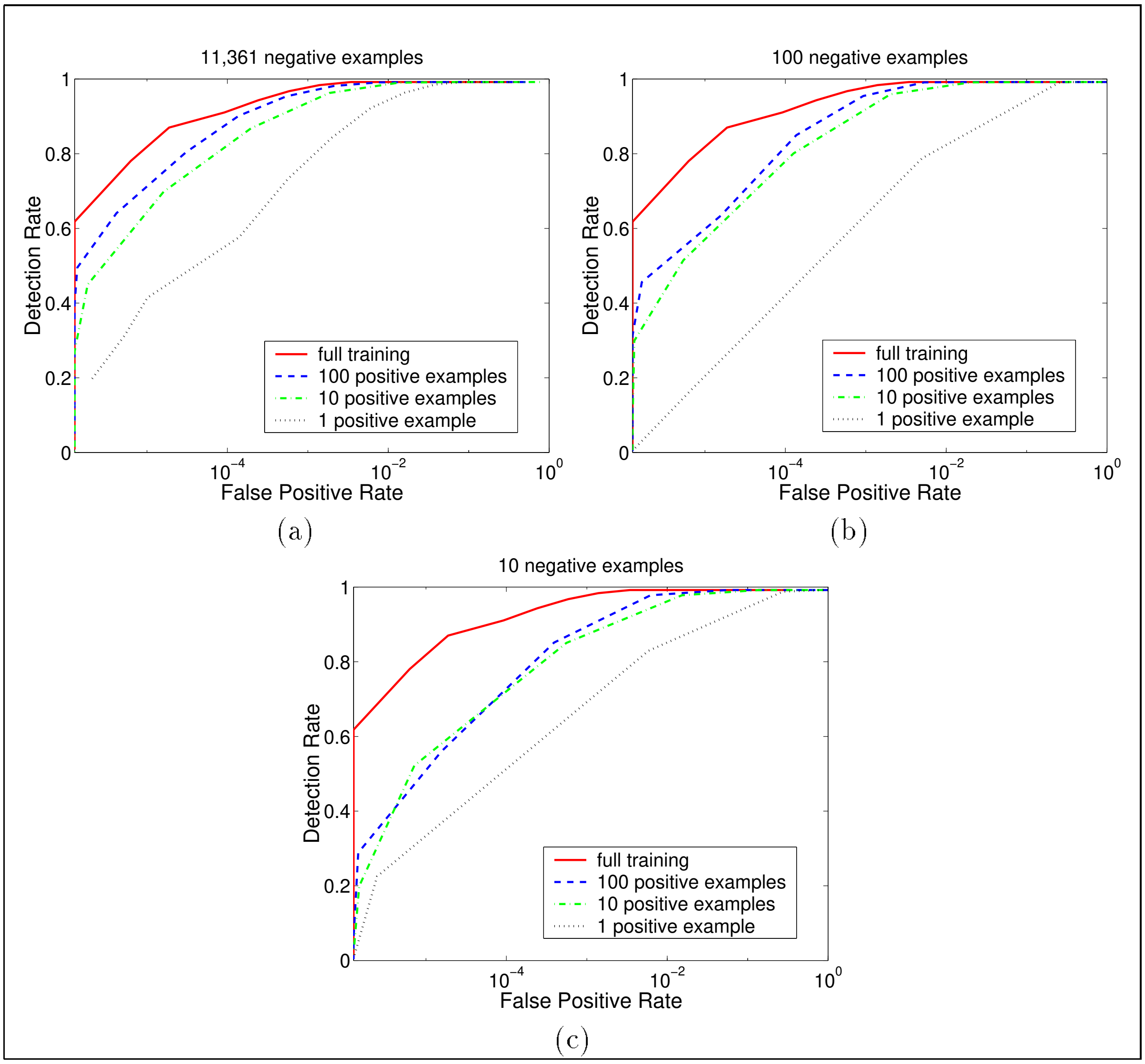


Figure 3-9: ROC curves comparing different sized positive training sets for people detection. The full 1,848 example positive training set is compared against 100, 10, and 1 positive examples each averaged over 10 iterations. We vary the size of the negative training set from 11,361, 100, and 10 negative examples in (a), (b), and (c), respectively. Even with 1 positive example, the system is able to learn a great deal of the structure of the people class.

Chapter 4

Detection by Components

One of the most prevalent problems with our static people detection system is its difficulty in detecting people when a portion of the body is occluded or there is little contrast between the background and part of the body. One would expect that if we knew there was a head, left arm, and legs in the proper configuration, but we could not see the right arm, that this may still be a person. In other words, if we look for the core building blocks, or *components* of an object, and allow some leniency in allowing one or two of the components that make up the object to be missing, this may result in a more robust detection system than our full pattern approach.

This chapter develops a component based object detection system for static images. Here, we apply it to the problem of people detection, but the architecture is quite general and could be applied to, among other objects, faces and cars. The description in this chapter closely follows the material published in [Mohan, 1999] and [Mohan *et al.*, 1999]. In Section 4.1, we introduce the *detection by components* framework and review some relevant work in component-based object detection. Section 4.2 describes the system development and architecture. Finally, we show results of the system and compare it to the original full-body people detection system in Section 4.3.

4.1 Introduction

In this type of system, geometric information concerning the physical structure of the human body supplements the visual information present in the image and should thereby improve the overall performance of the system. More specifically, the visual data in an image is used to detect body components and knowledge of the structure of the human body allows us to determine if the detected components are proportioned correctly and arranged in a permissible configuration. In contrast, a full-body person detector relies solely on visual information and does not take advantage of the known geometric properties of the human body.

Also, it is sometimes difficult to detect the human body pattern as a whole due to variations in lighting and orientation. The effect of uneven illumination and varying viewpoint on individual body components, like the head, arms, and legs, is less drastic and hence, they are comparatively easier to identify. Another reason to adopt a component based approach to people detection is that the framework directly addresses the issue of detecting people that are partially occluded or whose body parts have little contrast with the background. This is because the system may be designed, using an appropriate classifier combination algorithm, to detect people even if all of their components are not detected.

The fundamental design of the system is as a two-level hierarchical classifier: there are specialized detectors for finding the different components of a person at the base level, whose results are combined in a top level classifier. The component based detection system attempts to detect components of a person's body in an image, i.e. the head, the left and right arms, and the legs, instead of the full body. The system checks to ensure that the detected components are in the proper geometric configuration and then combines them using a classifier. We will show that this approach of integrating components using a classifier increases accuracy compared to the full-body version of our people detection system.

The system introduces a new hierarchical classification architecture to visual data classification. Specifically, it comprises *distinct* example based *component classifiers* trained to detect different objects at one level and a similar example based *combination classifier* at the next. This type of architecture, where example based learning is conducted at more than two levels, is called an Adaptive Combination of Classifiers (ACC). The component classifiers separately detect components of the person object, i.e. heads, legs, and arms. The combination classifier takes the output of the component classifiers as its input and classifies the entire pattern under examination as a "person" or a "non-person."

4.1.1 Classifier Combination Algorithms

Recently, a great deal of interest has been shown in hierarchical classification structures, i.e. pattern classification techniques that are combinations of several other classifiers. In particular, two methods have received considerable attention: *bagging* and *boosting*. Both of these algorithms have been shown to increase the performance of certain classifiers for a variety of datasets [Breiman, 1996, Freund and Schapire, 1996, Quinlan, 1996]. Despite the well documented practical success of these algorithms, the reason why bagging and boosting work is still open to debate. One theory proposed by Schapire [Schapire *et al.*, 1998] likens boosting to support vector machines in that both maximize the minimum margin over the training set. However, his definition of "margin" differs from [Vapnik, 1995]. Bauer and Kohavi, 1998, present a study of such structures including bagging and boosting, oriented towards determining the circumstances under which these algorithms are successful.

4.1.2 Previous Work

Most of the previous work in component based detection systems has focused on face detection. An overview of relevant systems [Yuille, 1991, Yuille *et al.*, 1992, Leung *et al.*, 1995, Burl *et al.*, 1995, Burl and Perona, 1996, Shams and Spoelstra, 1996, Yow and Cipolla, 1997, Forsyth and Fleck, 1997, Forsyth and Fleck, 1998, Lipson, 1996] is presented in the introduction (Section 1.3). We highlight several systems here.

In Yuille, 1991, and Yuille *et al.*, 1992, they describe systems that extract facial features in a framework where the detection problem is cast as an energy minimization problem; hand-crafted deformable templates are used for the individual features.

Leung *et al.*, 1995, Burl *et al.*, 1995, and Burl *et al.*, 1996, use local templates to match eye, nose, and mouth features on the human face and determine valid arrangements of these features using random graph matching. Computationally, this amounts to a constrained search through a very large set of candidate face configurations. This system has been shown to have some robustness against occlusions.

The system of [Shams and Spoelstra, 1996] uses a neural network to generate confidences for possible left and right eye regions that are paired together to form all possible combinations. The confidences of these pairings are weighted by their topographic suitability which are then thresholded to classify the pattern. These weights are defined by a 2D Gaussian.

Yow and Cipolla, 1997, have also developed a component based approach to detecting faces. In their system, potential features are categorized into candidate groups based on topographic evidence and probabilities that they are faces are assigned to these groups. The probabilities are updated using a Bayesian network. If the final probability measure of a group is above a certain threshold, then it is declared as a "detection." The features are initially identified using an image invariance scheme.

Where the component based systems described take different approaches to detecting objects in images by components, they have two similar features:

- they all have *component detectors* that identify candidate components in an image
- they all have a means to integrate these components and determine if together they define a face

4.2 System Details

In this section, we describe the structure and operation of the component based object detection system as applied to the domain of people detection.

4.2.1 Overview of System Architecture

The section explains the overall architecture of the system by tracing the detection process when the system is applied to an image. Figure 4-1 is a graphical representation of this procedure.

The process of classifying a pattern starts by taking a 128×64 window as an input. This input is then processed to determine where and at which scales the components of a person – head, legs, left arm, and right arm – may be found within the window using our prior knowledge of the geometry of bodies. All of these candidate regions are processed by the respective component detectors to find the "strongest" candidate components. There are four distinct component detectors in this system which operate independently of one another and are trained to find separately the four components of the human body: the head, the legs, and the left and right arms.

The component detectors process the candidate regions by applying the quadruple density Haar wavelet transform to them and then classifying the resultant data vector. The component classifiers are quadratic polynomials that are trained using the support vector machine algorithm. The training of the component and combination classifiers is described in detail in Section 4.2.2. The "strongest" candidate component is the one that produces the highest positive raw output, referred to in this thesis as the *component score*, when classified by the component classifiers. The raw output of a SVM is a rough measure of how well a classified data point fits in with its designated class and is defined in Section 4.2.2.

The highest component score for each component is fed into the combination classifier which is a linear classifier. If the highest component score for a particular component is negative, i.e. the component detector in question did not find a component in the geometrically permissible area, then a component score of zero is used instead. The combination classifier processes the set of scores received from the component classifier to determine if the pattern is a person.

Since our classifier is not shift and scale invariant, we follow the same brute force search procedure as used in the general detection system. The 128×64 window is shifted across and down the image. The image itself is processed at several sizes, ranging from 0.2 to 1.5 times its original size. These steps allow the system to detect various sizes of people at any location in an image.

4.2.2 Details of System Architecture

This section of the chapter outlines the details of the component detectors and the combination classifier.

Stage One - Identifying Components of People in an Image

When a 128×64 window is evaluated by the system, the component detectors are applied only to specific areas of the window at only particular scales. This is because

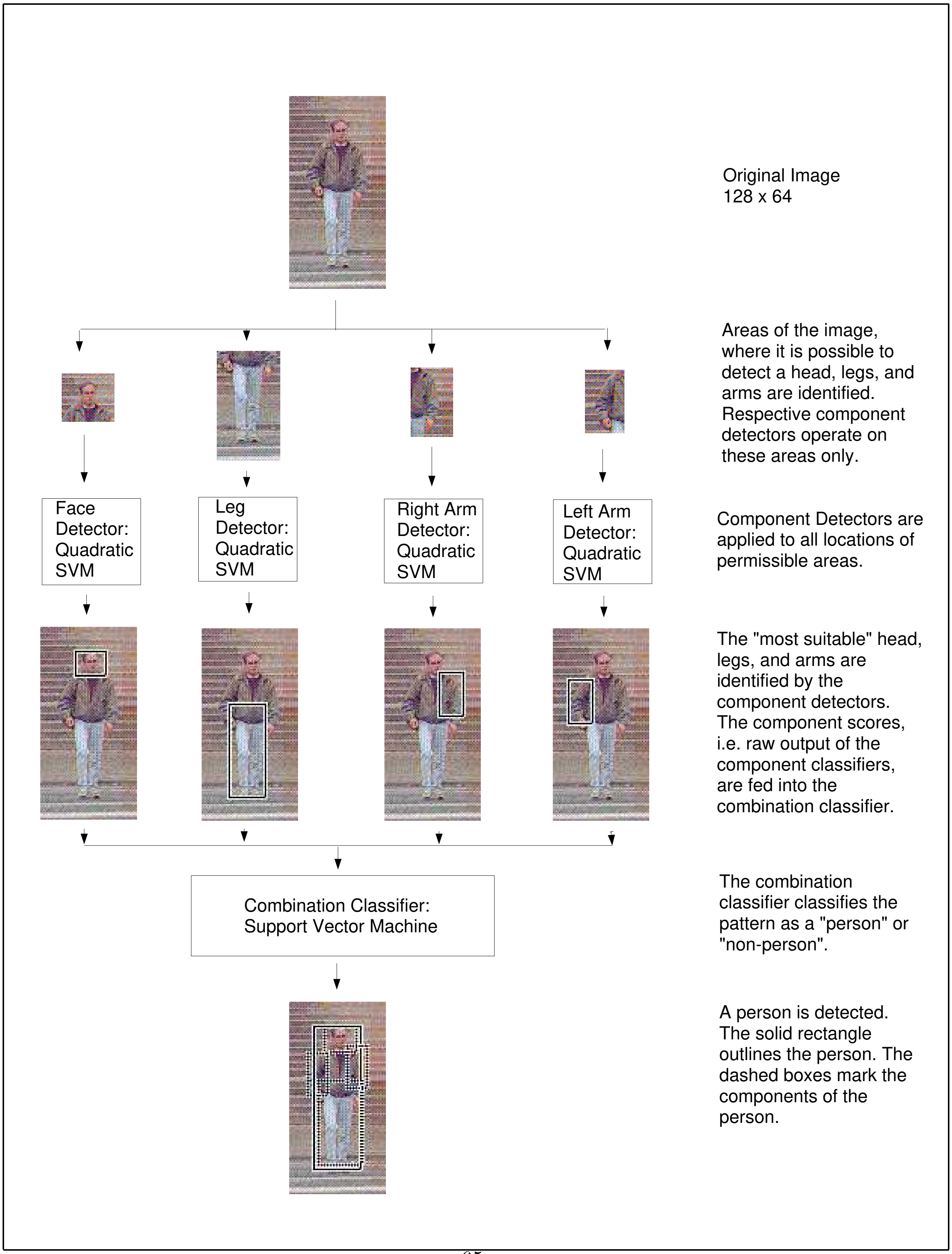


Figure 4-1: Diagrammatic description of the operation of the system.

Figure 4-2: It is very important to place geometric constraints on the location and scale of component detections. Even though a detection may be the "strongest" in a particular window examined, it might not be located properly. In this figure, the shadow of the person's head is detected with a higher score than the head itself. If we did not check for proper configuration and scale, component detections like these would lead to false alarms and/or missed detections of people.

the arms, legs, and head of a person have a defined relative configuration, i.e. the head is found above the legs, with left and right arms to either side and the components must also be proportioned correctly. This is, in effect, prior information that is being integrated into our detection system. By placing these geometric constraints on the location and scale of the components, we ensure that they are arranged in the form of a human body, and thus improve the performance of the object detection system. This is necessary, because even though a component detection is the "strongest" in a particular window under examination (i.e. it has the highest component score), it does not imply that it is in the correct position, as illustrated in Figure 4-2.

Since the component detectors operate on rectangular areas of the image, the constraints placed on the location and scale of component detections are expressed in terms of the properties of the rectangular region examined. For example, the centroid and boundary of the rectangular area determines the location of a component detection and the width of the rectangle is a measure of a component's scale. All coordinates are relative to the upper left hand corner of the 128×64 window.

We calculated the geometric constraints for each component from a sample of the training images. The constraints themselves, both in location and scale, are tabulated in Table 4.1 and shown in Figure 4-3. The values of quantities such as the location of the centroid, and top and bottom boundary edges of a component, were determined by taking the statistical mean of the quantities over positive detections in the training set. The tolerances were set to include all positive detections in the training set. Permissible scales were also estimated from the training images. There

Component	*Centroid*		*Scale*		*Other Criteria*
	Row	*Column*	*Minimum*	*Maximum*	
Head and Shoulders	23 ± 3	32 ± 2	28×28	42×42	
Lower Body		32 ± 3	42×28	69×46	*Bottom Edge:* Row: 124 ± 4
Right Arm Extended	54 ± 5	46 ± 3	31×25	47×31	
Right Arm Bent		46 ± 3	31×25	47×31	*Top Edge:* Row: 31 ± 3
Left Arm Extended	54 ± 5	17 ± 3	31×25	47×31	
Left Arm Bent		17 ± 3	31×25	47×31	*Top Edge:* Row: 31 ± 3

Table 4.1: Geometric constraints placed on each component. All coordinates are in pixels and relative to the upper left hand corner of a 128×64 rectangle.

are two sets of constraints for the arms, one intended for extended arms and the other for bent arms.

For each of the component detectors, the pixel images are processed with the quadruple density Haar wavelet transform described in Section 2.3.1 and Appendix A. We use the wavelets at the scales 16×16 and 8×8 to represent the patterns. As in the case of the full body detection system, we do the transform in each of the three color channels and for each scale, location, and orientation of wavelet, we use the one that is maximum in absolute value among the three color channels. In this way, the information in the three color channels is collapsed into a single virtual color image.

We use quadratic polynomial classifiers trained using support vector machines to classify the data vectors resulting from the Haar wavelet representation of the components. The optimal hyperplane is computed as a decision surface of the form:

$$f(\mathbf{x}) = sgn\left(g(\mathbf{x})\right) \tag{4.1}$$

where,

$$g(\mathbf{x}) = \left(\sum_{i=1}^{l^*} y_i \alpha_i K(\mathbf{x}, \mathbf{x}_\mathbf{i}^*) + b\right) \tag{4.2}$$

In Equation 4.2, K is one of many possible kernel functions; $y_i \in \{-1, 1\}$ is the class label of the data point $\mathbf{x}_i^*$; and $\{\mathbf{x}_i^*\}_{i=1}^{l^*}$ is a subset of the training data set. The $\mathbf{x}_i^*$ are called *support vectors* and are the points from the data set that fall closest to the separating hyperplane. Finally, the coefficients α_i and b are determined by solving a large-scale quadratic programming problem. The kernel function K that is used in the component classifiers is a quadratic polynomial and has the form shown below:

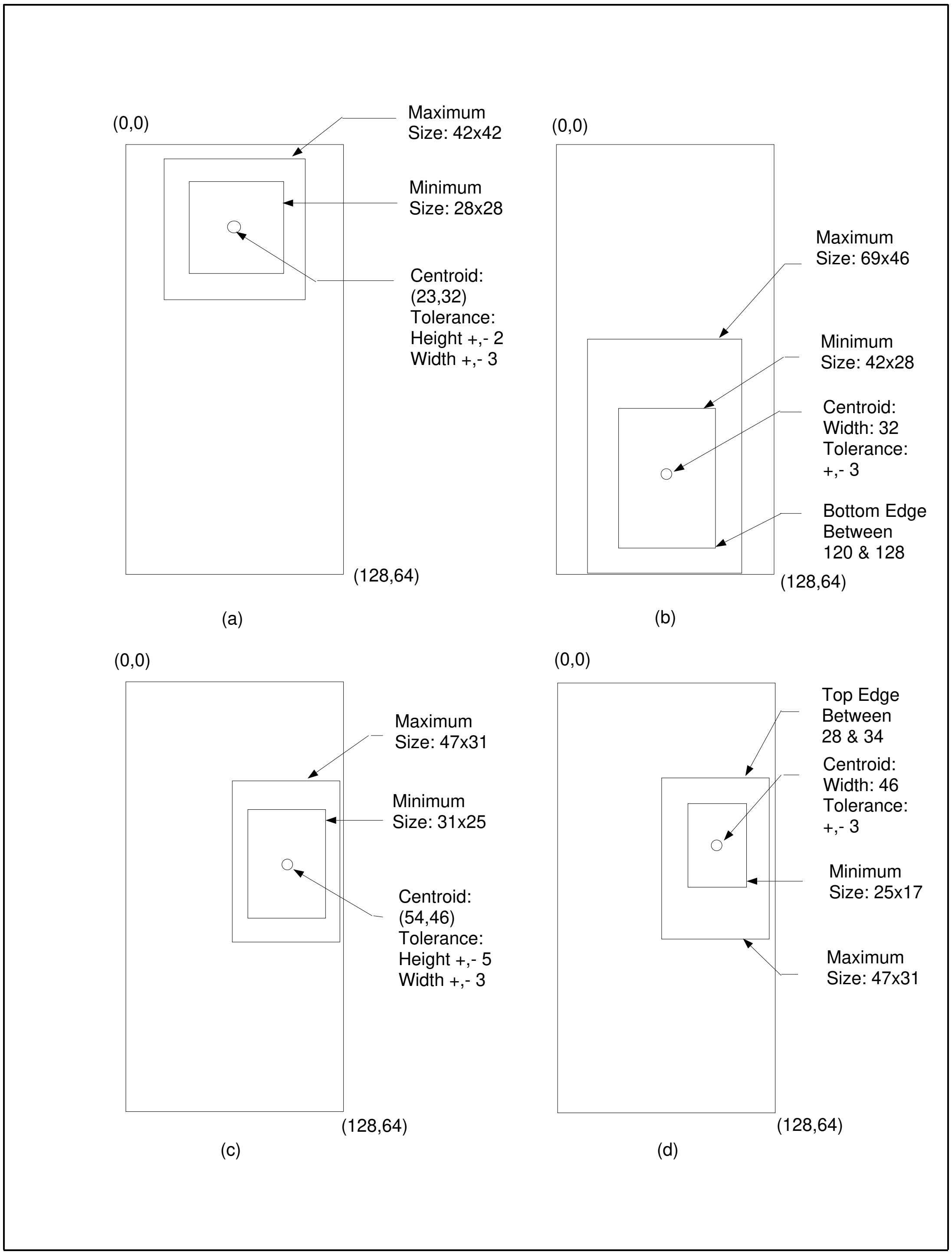


Figure 4-3: Geometric constraints that are placed on different components. All coordinates are in pixels and relative to the upper left hand corner of a 128×64 rectangle. Dimensions are also expressed in pixels. (a) illustrates the geometric constraints on the head, (b) the lower body, (c) an extended right arm, and (d) a bent right arm.

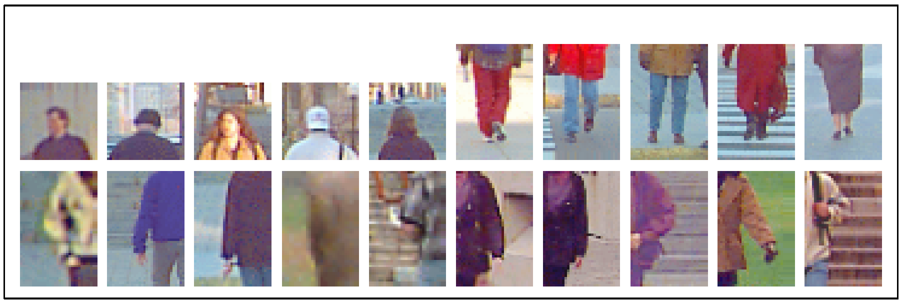


Figure 4-4: The top row shows examples of "heads and shoulders" and "lower bodies" of people that were used to train the respective component detectors. Similarly, the bottom row shows examples of "left arms" and "right arms" that were used for training purposes.

$$K(\mathbf{x}, \mathbf{x_i^*}) = (\mathbf{x} \cdot \mathbf{x_i^*} + 1)^2 \tag{4.3}$$

$f(\mathbf{x}) \in \{-1, 1\}$ in Equation 4.1 is referred to as the *binary class* of the data point $\mathbf{x}$ which is being classified by the SVM. Values of 1 and -1 refer to the classes of the positive and negative training examples, respectively. As Equation 4.1 shows, the binary class of a data point is the sign of the *raw output* $g(\mathbf{x})$ of the SVM classifier. The raw output of an SVM classifier is the distance of a data point from the decision hyperplane. In general, the greater the magnitude of the raw output, the farther the point is from the decision surface or the more likely the point belongs to the binary class it is grouped into by the SVM classifier.

The component classifiers are trained on positive and negative images for their respective classes. The positive examples are of arms, legs, and heads of people in various environments, both indoors and outdoors, and under various lighting conditions. The negative examples are taken from scenes that do not contain any people. Examples of positive images used to train the component classifiers are shown in Figure 4-4.

Stage Two - Combining the Component Classifiers

Once the component detectors have been applied to all geometrically permissible areas within the 128×64 window, the highest component score for each component type is entered into a four dimensional vector that serves as the input to the combination classifier. The component score is the raw output of the component classifier and is the distance of the test point from the decision hyperplane. This distance is a rough measure of how "well" a test point fits into its designated class. If the component detector does not find a component in the designated area of the 128×64 window, then zero is placed in the data vector. A component score of zero refers to a test point that is classified as neither a "component" nor a "non-component" because it lies on the hyperplane.

Head and Shoulder Scores	*Lower Body Scores*	*Right Arm Scores*	*Left Arm Scores*
Positive Examples			
2.415	3.152	3.233	3.145
1.861	1.855	2.339	2.280
4.184	2.332	3.258	3.994
2.871	1.691	2.311	1.221
Negative Examples			
0.677	0.694	0.817	1.020
4.530	0.231	0.252	0.824
0.105	0.021	0.002	0.560
1.869	0.010	0.718	1.746

Table 4.2: Examples of positive and negative data points used to train the combination classifier. The entries are component scores. The component scores of the positive examples are generally higher.

The combination classifier is trained with a linear kernel and has the following form:

$$K(\mathbf{x}, \mathbf{x_i^*}) = (\mathbf{x} \cdot \mathbf{x_i^*} + 1) \tag{4.4}$$

This type of hierarchical classification architecture where learning occurs at multiple stages is termed Adaptive Combination of Classifiers (ACC).

Positive examples were generated by processing 128×64 images of people at one scale and taking the highest component score from detections that are geometrically allowed for each component type. Table 4.2 shows examples of data vectors that were used to train the combination classifier.

4.3 Results

The performance of this system is compared to that of other component based person detection systems that combine the component classifiers in different ways, as well as the full body person detection system. This framework allows us to determine the strengths of the component based approach to detecting objects in images and the performance of various classifier combination algorithms.

4.3.1 Experimental Setup

All of the component based detection systems that were tested in this experiment are two tiered systems. Specifically, they detect heads, legs, and arms at one level and

Component Classifier	*Number of Positive Examples*	*Number of Negative Examples*
Head and Shoulders	856	9315
Lower Body	866	9260
Left Arm	835	9260
Right Arm	838	9260

Table 4.3: Number of Positive and Negative Examples Used to Train the Different Component Classifiers.

at the next they combine the results of the component detectors to determine if the pattern in question is a person or not. The component detectors that were used in all of the component based people detection systems are identical and are described in section 4.2.2. The positive examples for training these detectors were obtained from the same database that is used in the full body system. The images of people were taken in Boston and Cambridge, Massachusetts, with different cameras, under different lighting conditions, and in different seasons. This database includes images of people who are rotated in depth and who are walking, in addition to frontal and rear views of stationary people. The positive examples of the lower body include images of women in skirts and people wearing full length overcoats as well as people dressed in pants. Similarly, the database of positive examples for the arms was varied in content, and included arms at various positions in relation to the body. The negative examples were obtained from images of natural scenery and buildings that did not contain any people. The number of positive and negative examples that were used to train the different component classifiers is presented in Table 4.3.

Adaptive Combination of Classifiers Based Systems

Once the component classifiers were trained, the next step in evaluating the Adaptive Combination of Classifiers (ACC) based systems was to train the combination classifier. Positive and negative examples for the combination classifier were collected from the same databases that were used to train the component classifiers. A positive example was obtained by processing each image of a person at a single appropriate scale. The four component detectors were applied to the geometrically permissible areas of the image and at the allowable scales. The greatest positive classifier output for each component were assembled as a vector to form a positive training example. If all of the component scores were not positive then no vector was formed and the window examined did not yield an example. The negative examples were computed in a similar manner except that this process was repeated over the entire image and at various scales. The images for the negative examples did not contain people.

We used 889 positive examples and 3,106 negative examples to train the combination classifiers. First, second, third, and fourth degree polynomial classifiers were trained and tested.

The trained system was run over our positive test database of 123 images of people to determine the positive detection rate. There was no overlap between these images and the ones that were used to train the system. The out-of-sample false alarm rate was obtained by running the system over the negative test database of 50 images that do not contain any people. These images are pictures of natural scenery and buildings. By running the system over these 50 images, 796,904 windows were examined and classified, as in our full-body detection system. We generate ROC curves to measure their performance by shifting the decision surface.

Voting Combination of Classifiers Based System

The other method of combining the results of the component detectors that was tested is a Voting Combination of Classifiers (VCC). VCC systems combine classifiers by implementing a voting structure amongst them. One way of viewing this arrangement is that the component classifiers are "weak experts" in the matter of detecting people. VCC systems poll the weak experts and then, based on the results, decide if the pattern is a person. For example, in a possible implementation of VCC, if a majority of the weak experts classify a pattern as a person, then the system declares the pattern to be a person.

One motivating reason for trying VCC as an approach to combining the component classifiers is that since VCC is one of the simplest classes of classifier combination algorithms, it affords the best opportunity to judge the strengths of a component based object detection system that is not augmented with a powerful classifier combination method. Similarly, when compared to an ACC based system, one can determine the benefits of more sophisticated classifier combination methods. Since the computational complexity of these methods is known and the experiment described in this section determines their performance, this framework characterizes the tradeoff involved between enhanced performance and greater computational complexity for these systems. The person detection systems that are evaluated here are: the ACC based system, the VCC based system, and the baseline full-body system.

In the incarnation of VCC that is implemented and tested in this experiment, a positive detection of the person class results only when all four component classes are detected in the proper configuration. The geometric constraints placed on the components are the same in the ACC and VCC based systems and are described in Section 4.2.2. For each pattern that the system classifies, the system must evaluate the logic presented below:

$$person \;=\; Head \;\&\; Legs \;\&\; Left\ arm \;\&\; Right\ arm \tag{4.5}$$

where a state of *true* indicates that a pattern belonging to the person class has been detected.

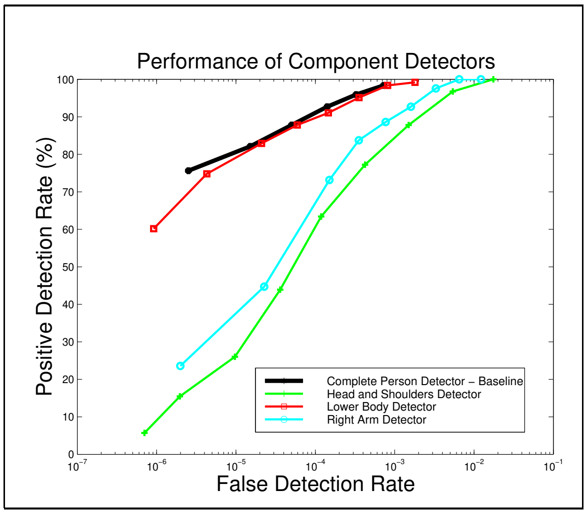


Figure 4-5: ROC curves illustrating the ability of the component detectors to correctly detect a person in an image. The positive detection rate is plotted as a percentage against the false alarm rate which is measured on a logarithmic scale. The false alarm rate is the number of false positive detections per window inspected.

The detection threshold of the VCC based system is determined by selecting appropriate thresholds for the component detectors. The thresholds for the component detectors are chosen such that they all correspond to approximately the same positive detection rate. This information was estimated from the ROC curves of each of the component detectors that are shown in Figure 4-5. For example, if one wished to run the VCC based system at a threshold that corresponded to a positive detection rate of 92%, then they would choose thresholds of 0.77, 0.69, and 0.80 for the head, legs, and arm classifiers, respectively. These ROC curves were calculated in a manner similar to the procedure described earlier in Section 4.3.1. A point of interest is that these ROC curves indicate how discriminating the individual components of a person are in the process of detecting the full body. The legs perform the best, followed by the arms and the head. The superior performance of the legs may be due to the fact that the background of the lower body in images is usually either the street, pavement, or grass and hence is relatively clutter free compared to the background of the head and arms.

4.3.2 Experimental Results

An analysis of the ROC curves suggests that a component based person detection system performs very well, and significantly better than the baseline full body system at all thresholds. This is noteworthy because the baseline system has produced very accurate results. It should be emphasized that the baseline system uses the same image representation scheme (Haar wavelets) and classifier (SVM) that the component

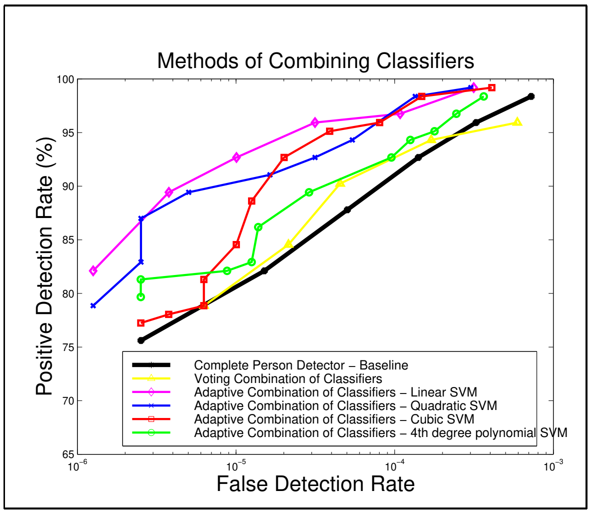


Figure 4-6: ROC curves comparing the performance of various component based people detection systems. The systems differ in the method used to combine the classifiers that detect the various components of a person's body. The positive detection rate is plotted as a percentage against the false alarm rate which is measured on a logarithmic scale. The false alarm rate is the number of false positives detections per window inspected. The curves indicate that a system in which a linear SVM combines the results of the component classifiers performs best. The baseline system is a full body person detector similar to the component detectors used in the component based system.

detectors used in the component based systems. All of the component based systems' performance were comparable to, or better than, the baseline system.

For the component based systems, the ACC approach produces better results than VCC. In particular, the ACC based system that uses a linear classifier to combine the components is the most accurate. During the course of the experiment, the linear SVM system displayed a superior ability to detect people even when one of the components was not detected, in comparison to the higher degree polynomial SVM systems. A possible explanation for this observation may be that the higher degree polynomial classifiers place a stronger emphasis on the presence of combinations of components due to the structure of their kernels [Burges, 1998]. The second, third, and fourth degree polynomial kernels include terms that are products of up to two, three, and four elements, which are component scores. This suggests that all of those elements must be "person like" for the pattern to be classified as a person. The emphasis placed on the presence of combinations of components increases with the degree of the polynomial classifier. The results show that the performance of the ACC based systems decreases with an increase in the degree of the polynomial classifier. In fact, the ROC curve for the ACC based system that employs a fourth degree polynomial classifier is very similar to the VCC based system. Interestingly, both of the above systems look for all four components in a pattern. The VCC based system explicitly requires the presence of all four components where as the ACC based system that uses the fourth degree polynomial classifier makes it an implicit requisite due to the design of its kernel. Two other possible reasons for the decrease in performance with higher degree polynomials is that higher degree classifiers may need more training data, or they could be overfitting.

It is also worth mentioning that the database of test images that was used to generate the ROC curves did not just include frontal views of people, but also contained a variety of challenging images. Some of these classes of images portray exactly the situations in which we would expect a component based approach to improve results. Included are pictures of people walking and running. In some of the images, the person is partially occluded or a part of their body has little contrast with the background. A few of the images depict people who are slightly rotated in depth. Figure 4-7 is a selection of these images.

Figure 4-8 shows the results obtained when the system was applied to images of people who are partially occluded or whose body parts blend in with the background. In these examples, the system detects the person while running at a threshold that, according to the ROC curve shown in Figure 4-6, corresponds to a false detection rate of less than one false alarm for every 796,904 patterns inspected.

Figure 4-9 shows the result of applying the system to sample images with clutter in the background. Even under such circumstances the system performs very well. The lower four images were taken with different cameras than the instruments used for the training set images. The conditions and surroundings for these pictures are different, as well.

Figure 4-7: Samples from the test image database. These images demonstrate the capability of the system. It can detect running people, people who are slightly rotated, people whose body parts blend into the background (bottom row, second from right - person detected even though the legs are not), and people under varying lighting conditions (top row, second from left - one side of the face is light and the other dark).

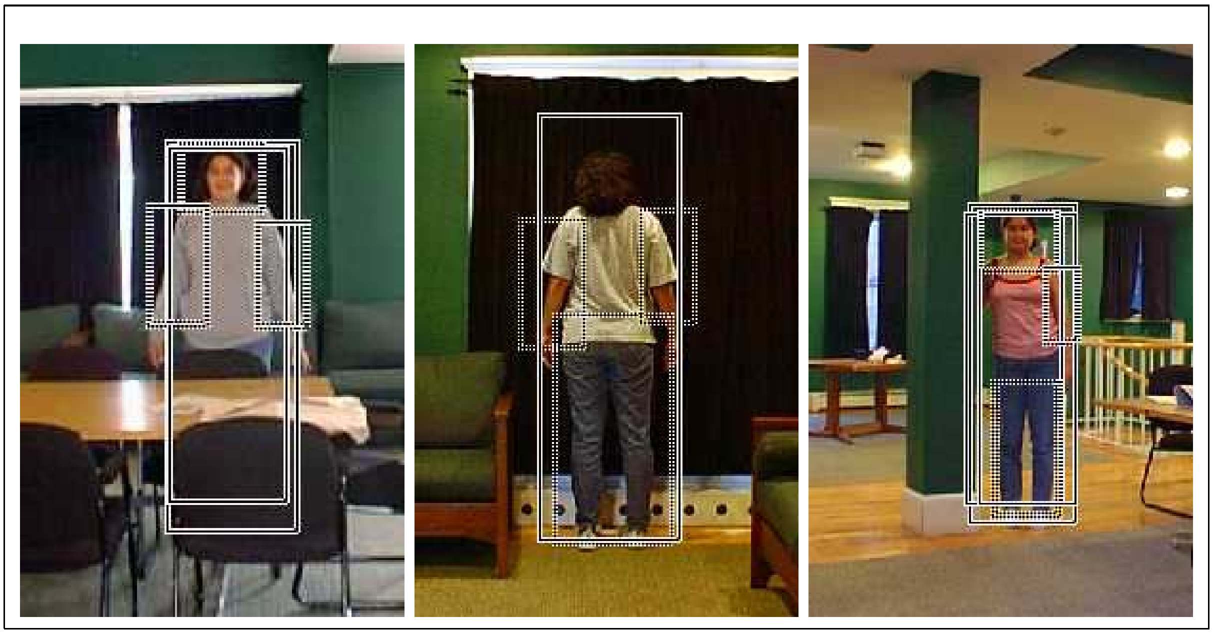


Figure 4-8: Results of the system's application to images of partially occluded people and people whose body parts have little contrast with the background. In the first image, the person's legs are not visible; in the second image, her hair blends in with the curtain in the background; and in the last image, her right arm is hidden behind the column.

Figure 4-9: Results from the component based person detection system. The solid boxes outline the complete pedestrian, where the dashed rectangles are the components.

Chapter 5

A Real-Time System and Focus of Attention

While the core system is quite effective, the processing time (among other characteristics of the system) limits any practical applications. Clearly, considering every pattern in an image is wasteful and there may be ways of limiting the number of patterns that are processed. Taking people detection as our domain, we view the proper use of our detection system for practical applications as one where some *focus of attention* mechanism identifies areas in the image where there may be a person and then our detection system processes patterns in only those regions.

This chapter presents:

- A real-time implementation of our people detection system as part of a driver assistance system that uses an obstacle detection module to provide a bounding box for the detection system.

- An integration of our people detection system with a biologically inspired focus of attention module that identifies salient areas of an image using several feature maps.

5.1 A Real-Time Application

As alluded to in the introduction, there are many possible applications of our object detection technology, ranging from automotive assistance systems to surveillance. The only factor that is inhibiting our system from being used right now in such systems is the relatively slow processing speed. It is important to note that our full system is, for the most part, an unoptimized research tool as we have not invested significant amounts of effort in improving the core speed.

As an alternative to dynamic detection strategies, we can use a modified version of our static detection system to achieve real-time performance. This section describes a real-time application of our technology as part of a larger system for driver assistance. The combined system, including our people detection module, is currently

Figure 5-1: Side view of the DaimlerChrysler S Class vehicle with the Urban Traffic Assistant (UTA); our people detection system has been integrated into the system.

deployed "live" in a DaimlerChrysler S Class demonstration vehicle (Figure 5-1). The remainder of this section describes the integrated system.

5.1.1 Speed Optimizations

Our original unoptimized static detection system for people detection in color images processes sequences at a rate of 1 frame per 20 minutes which is clearly inadequate for any real-time automotive application. We have implemented optimizations that have yielded several orders of magnitude worth of speedups.

subset of 29 features: Instead of using the entire set of 1,326 wavelet features, we use just 29 of the more important features (manually chosen) that encode the outline of the body. This changes the 1,326 dimensional inner product in Equation 2.5 into a 29 dimensional dot product. The 29 features are shown overlayed on an example person in Figure 3-7.

reduced set vectors: From Equation B.14, we can see that the computation time is also dependent on the number of support vectors, N_s. In our system, this is typically on the order of 1,000. We use results from [Burges, 1996] to obtain an equivalent decision surface in terms of a small number of synthetic vectors. This method yields a new decision surface that is equivalent to the original one but uses just 29 vectors.

gray level images: Our use of color images is predicated on the fact that the three different color channels (RGB) contain a significant amount of information that

Figure 5-2: A view of the cameras in UTA.

gets washed out in gray level images of the same scene. This use of color information results in significant computational cost; the resizing and Haar transform operations are performed on each color channel separately. In order to improve system speed, we modify the system to process intensity images.

The reduction in performance resulting from using gray level images and 29 features is shown in Figure 5-3. Taking a false positive rate of 10^{-4}, whereas the full 1,326 color feature system achieves 90% accuracy, the 29 gray feature system achieves 50% accuracy. While this is a significant reduction in performance, this level of accuracy may still be adequate for certain applications.

5.1.2 Focus of Attention

To further enhance the processing speed of the system, we can use a focus of attention module that concentrates processing only on areas of an image that are likely to contain people. This focus of attention can key off of different characteristics, including motion, distance, local image complexity, shape, color, etc.

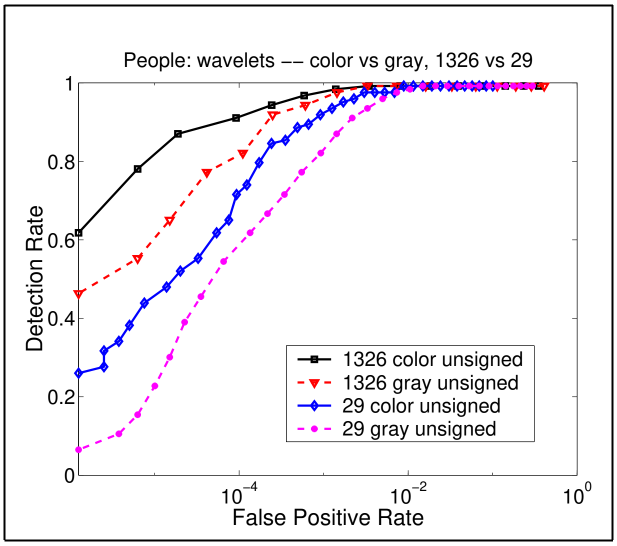


Figure 5-3: ROC curves for people detection gray level and color, 1,326 and 29 features. While going from 1,326 color features to 29 gray level features results in a significant decrease in accuracy, the performance may still be adequate for some applications.

5.1.3 Integration With The DaimlerChrysler Urban Traffic Assistant

To this end, we have integrated our people detection system with a stereo-based obstacle detection system in collaboration with DaimlerChrysler AG. DaimlerChrysler has obviously motivated interests in obstacle detection algorithms for automotive applications, as a means to aid driving and, ultimately, to allow for autonomous driving. One of the important requirements of the system is that it is able to deal with both highway and urban scenes, the latter being much more complex than the former.

The DaimlerChrysler Urban Traffic Assistant (UTA) is a real-time vision system for obstacle detection, recognition, and tracking [Franke and Kutbach, 1996, Franke *et al.*, 1997, Franke *et al.*, 1998]. The car has a binocular stereo vision system (Figure 5-2) mounted on the rear-view mirror, a 200 MHz PowerPC 604 in the rear trunk, and a flat panel display between the driver and passenger seats on which system processing and output is visualized. Figure 5-4 shows the view of the system inside the car. UTA currently has several processing modules implemented including pedestrian motion recognition, lane detection, car detection, sign detection, and an automatic car following system.

Their system relies on 3D position and depth information using the stereo cameras. To overcome the expensive correspondence problem, they have developed a feature based approach to stereo analysis that runs at 25 Hz. The system clusters feature points that correspond to the same object, thereby providing a rectangular bounding box around each obstacle in the scene.

Using this bounding box which closely outlines the shape of the obstacle, we expand this area to provide a larger region of interest in which we will run our people detection system. This is done to alleviate possible misalignments in the bounding box provided by the stereo system. Furthermore, the stereo system provides an accurate estimate of the distance to each object. Using this information we can constrain the number of sizes at which we look for people to a small number, typically under three scales.

Within these regions of interest, we use our 29 gray level feature system with the reduced set method that lowers the number of support vectors to 29. In real-world test sequences processed while driving through Esslingen/Stuttgart, Germany, we are able to achieve rates of more than 10 Hz. DaimlerChrysler expects to soon upgrade their onboard system to 400+ MHz computers, so further gains in speed will follow.

5.1.4 Future Work

The portion of the total system time that is spent in our people detection module is 15 ms per obstacle. An analysis of how much time is taken by each portion of the people detection module shows that the smallest amount of time is being spent in the SVM classification. This bodes well for improving the performance. We should be able to use a much richer set of features than the 29 that are currently used, perhaps on the order of a few hundred features, without significantly degrading the speed of the system.

5.2 Focus of Attention Experiments

Our current metaphor for detection is one where we do a brute force search in the entire image; this is clearly inefficient. The integrated system presented in Section 5.1 showcases one method for focusing processing only on *important* areas of an images which are, in this case, defined by the results of an obstacle detection system. In this section, we present another version of our system that uses a *focus of attention* module to direct the processing in only certain areas of the image.

To detect objects in cluttered scenes, the human visual system rapidly focuses its attention to different areas where there is compelling visual information that may indicate the presence of an interesting object. The information that our system keys off of could include features such as intensity, color, and orientation discontinuities.

This type of model is developed in [Itti *et al.*, 1998, Itti and Koch, 1999]. Their system decomposes an input image into several feature maps, each of which key off of different visual information. The visual features the method uses are intensity, color, and orientation (via Gabor filters). These feature maps are integrated into a single saliency map where, at a given point in the image, the individual feature responses are additively combined. The mechanism that changes the focus of attention from

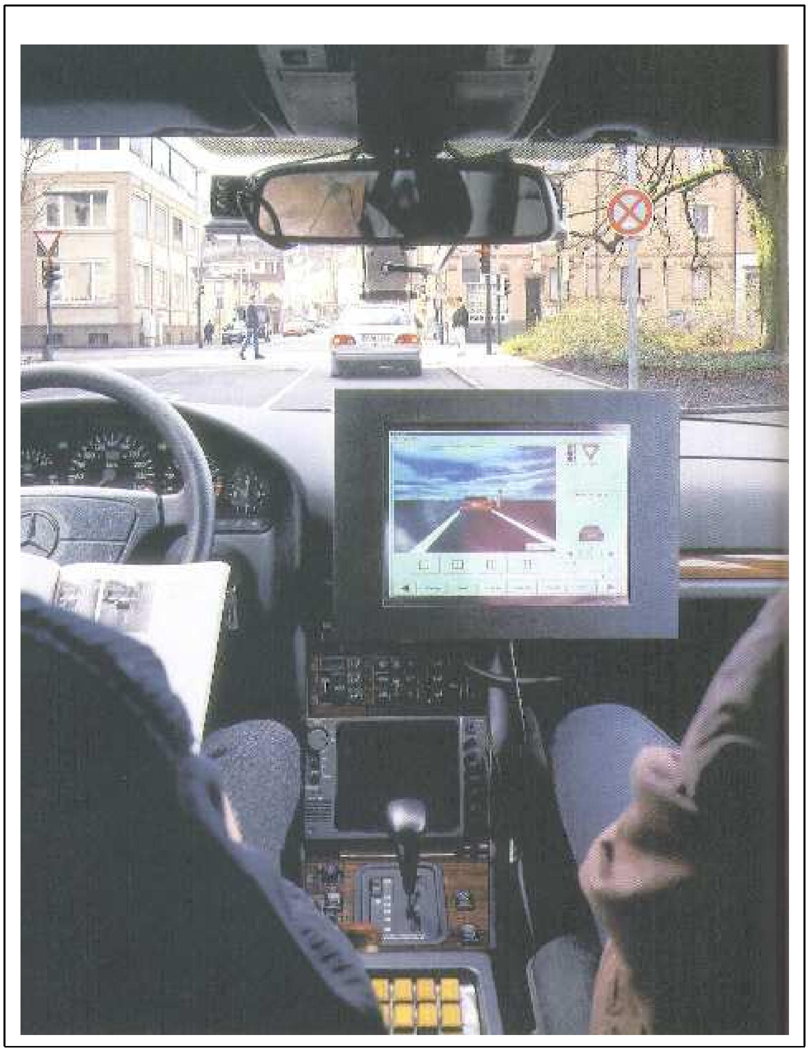


Figure 5-4: Looking out from UTA; processing results are shown on the flat panel display between the driver and passenger seats.

point to another is modeled as a dynamical neural network. The network outputs the ordered salient locations in simulated time as (x, y) positions in the image.

As in the case of our DaimlerChrysler integration, we can use this information to focus processing on only the important areas of an image. The saliency based attention system does not provide information on the size of the salient regions, so for our test, we define a 150×60 region centered on each salient point in which we try to find people – note that this is the maximum body size of people that we will look for. Furthermore, we limit the number of regions processed to the top N salient regions per image and test $N = \{5, 6, 7\}$.

The ROC curves for the system compared to the base system are shown in Figure 5-6. The performance of the versions using the focus of attention mechanism is slightly

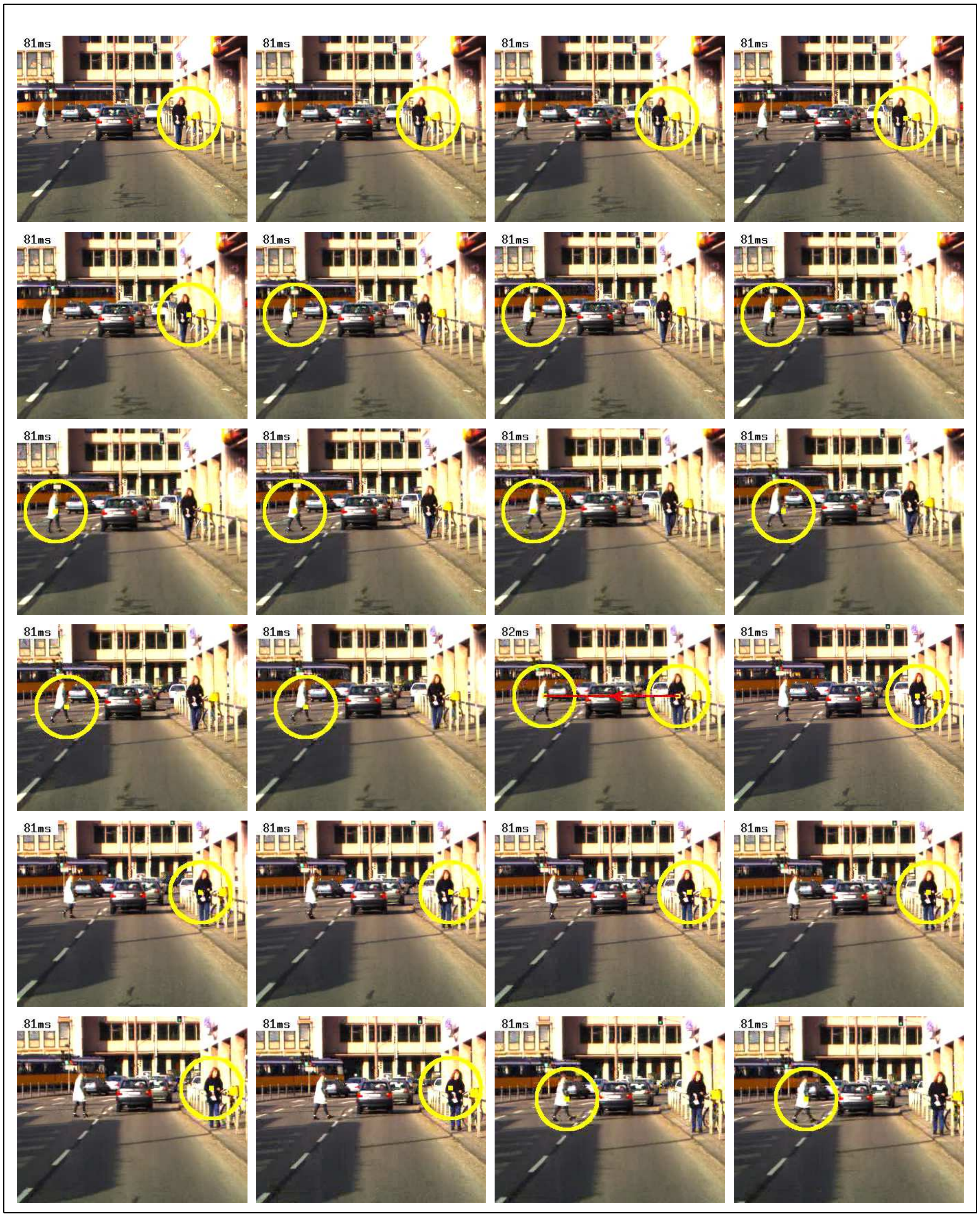


Figure 5-5: A sequence showing the single most salient location for each frame as processed by the system of [Itti *et al.*, 1998, Itti and Koch, 1999]. Our people detection system can use this *focus of attention* module as a preprocessor to target specific areas of an image.

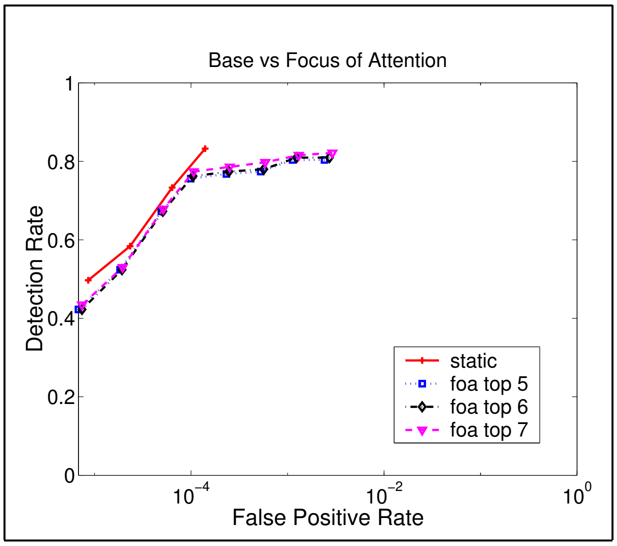


Figure 5-6: ROC curves for people detection comparing the base system to one that preprocess an image by focusing attention on certain salient regions in the image.

worse than the base system. Fewer patterns are considered resulting in a significantly faster system, however. Also, the ROC curves seem to indicate that after the first 5 salient regions, considering more of these regions may not impact performance.

Integrating this saliency information in our detection framework is not a perfect fit on account of the following:

- The center of the salient region is often not at the center of the person and there is no scale information so, by default, a large area around the salient spot is examined.

- While it usually finds one of the people in the sequence as the most salient, the other person may be the 8th or 9th most salient spot. This, combined with the previous point, means that a large portion of the image still ends up being examined

Irrespective of these issues, the integration of a focus of attention mechanism results in significant improvement in processing time. Table 5.1 lists the percentage of total patterns that are processed in the versions of our system using this focus of attention module as compared to the base system that processes all regions. These results indicate that approximately the same performance can be achieved while processing only one third of the patterns.

	Percentage of patterns processed
Base	100.00%
FOA (top 7)	38.97%
FOA (top 6)	35.14%
FOA (top 5)	29.22%

Table 5.1: The percentage of total possible patterns that are examined by each version of the system; the use of a focus of attention mechanism such as the one we consider here can result in significant increase in processing speed.

Chapter 6

Integration Through Time

This thesis has focused mainly on the static detection problem but with the increasing amount of video information available, it is of great interest to see how our system could be applied to video sequences. The straightforward method of applying our static system to video sequences is to analyze each frame as a static image. However, this brute force technique ignores all dynamical information available, information that could be quite useful for the purposes of detection. The brute force version will be used as our benchmark in this chapter.

In our extensions of the system to handle the time domain, we will not drastically alter the core technology used in the static version. Rather, we will either slightly modify it or will augment it with various modules that use the static detection system in different ways. In this manner, each of the techniques we present in this chapter can be seen as more general means of converting a static detection system into one with improved processing capabilities for video sequences.

Our first system is a pure pattern classification approach to dynamical object detection. Here, we seek to circumvent the need for 1) the extensive engineering that is quite typical in current dynamical detection systems, and 2) assuming particular underlying dynamics. We will modify our base static detection approach to represent dynamic information by extending the static representation into the time domain. With this new representation, the system will be able to learn limited dynamics of people, with or without motion. The system will learn what a person looks like and what constitutes valid dynamics over short time sequences, without the need for explicit models of either shape or dynamics.

The second system to take advantage of dynamical information is a rule based module that integrates information through time as an approximation to a Kalman filter. Kalman filtering theory assumes an underlying linear dynamical model and, given measurements of the location of a person in one image, yields a prediction of the location of the person in the next image and the uncertainty in this prediction. Our heuristic smooths the information in an image sequence over time by taking advantage of this fundamental *a priori* knowledge that a person in one image will appear in a similar position in the next image. We can smooth the results through time by automatically eliminating false positives, or detections that do not persevere over small subsequences.

Figure 6-1: Example image sequences that are used to train the pure pattern classification version of our dynamic detection system.

The third system we will develop uses a new approach to propagating general, multi-modal densities through time, based on the so called Condensation technique [Isard and Blake, 1998]. This technique has a significant advantage over the Kalman filter, namely, that it is not constrained to model a single Gaussian density but can effectively model arbitrarily complex densities. Condensation is also able to gracefully handle changes in scene structure, i.e. objects entering or exiting the scene, whereas, if we were to use a Kalman filter, we would have to run our detection system every few frames and initialize a new Kalman filter on any newly detected objects.

6.1 Pure Pattern Classification Approach

In this section, our goal is to develop a detection system for video sequences that makes as few assumptions as possible. We do not want to develop an explicit model of the shape of people or outwardly model their possible motions in any way. Instead of a dynamical model, we will use very high dimensional feature vectors to describe patterns through time. These will be used to train a support vector machine classifier, which we expect will implicitly model certain dynamics.

We would like a technique that implicitly generates a model of both the shape and valid dynamical characteristics of people at the same time from a set of training data. This should be accomplished without assuming that human motion can be approximated by a linear Kalman filter or that it can be described by a hidden Markov model or any other dynamical model. The only assumption we will make is that five consecutive frames of an image sequence contain characteristic information regarding the dynamics of how people appear in video sequences. From a set of training data, the system will learn exactly what constitutes a person and how people typically appear in these short time sequences.

Instead of using a single 128×64 pattern from one image as a training example, our new approach takes the 128×64 patterns at a fixed location in five consecutive frames, computes the 1,326 features for each of these patterns, and concatenates them into a single 6,630 dimensional feature vector for use in the support vector training. We use images $t-4, t-3, t-2, t-1, t$ where the person is aligned in the center of the image at t. Figure 6-1 shows several example sequences from the training set. The full training set is composed of 1,379 positive examples and 3,822 negative examples.

The extension to detecting people in new images is straightforward; for each candidate pattern in a new image, we concatenate the wavelet features computed for that pattern to the wavelet features computed at that location in the previous four frames. The full feature vector is subsequently classified by the SVM.

We emphasize that it is the implicit ability of the support vector machine classification technique to handle small sets of data that sparsely populate a very high-dimensional feature space that allows us to tackle this problem.

In developing this type of representation, we expect that the following dynamical information will be evident in the training data and therefore encapsulated in the classifier:

- people usually display smooth motion or are stationary
- people do not spontaneously appear or disappear from one frame to another
- camera motion is usually smooth or stationary

One of the primary benefits derived from this technique is that it extends this rich feature set into the time dimension and should be able to detect people at high accuracy, while eliminating transient false positives that would normally appear when using the static detection system.

This is purely a data-driven pattern classification approach to dynamical detection. We compare this approach to the static detection system, trained with the individual images corresponding to frame t in each of the sequences, so there are 1,379 positive and 3,822 negative 1,326 dimensional feature vectors as training for the static detection system. Both the static and dynamic systems are tested on the

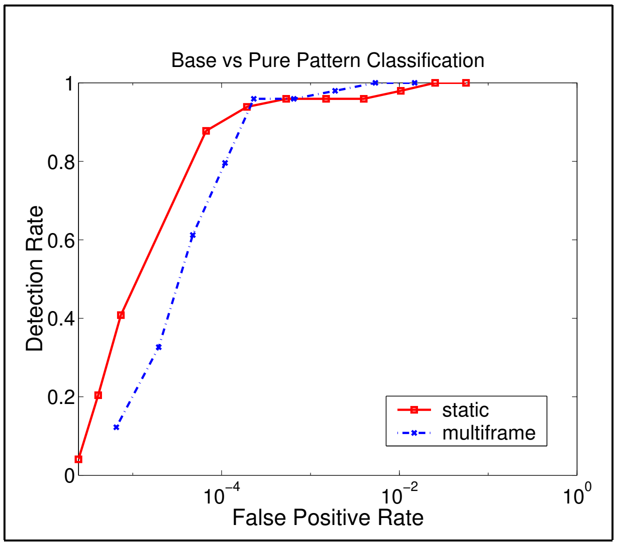


Figure 6-2: ROC curves for the static and pure pattern classification detection systems. The detection rate is plotted against the false positive rate, measured on a logarithmic scale. The false positive rate is defined as the number of false detections per inspected window.

same out-of-sample sequence. Figure 6-2 shows the ROC curves for the two systems. From the ROC curves, we see that the system that has incorporated dynamical information performs significantly worse than the static system at most points on the ROC curve. If we look at our training data, we see that right to left moving people are significantly underrepresented, but the people in the test sequence are moving from left to right. The bias in the training data that is not reflected in the test data may be causing this worse performance.

Regardless, this experiment illustrated that the dynamic system is capable of doing classification in a 6,630 dimensional space with only 5,201 training examples.

It is important to note that our features are not the 3D wavelets in space and time. What we have done is taken a set of 2D wavelet features spread through time and used these to develop our model. One extension of our system that would be interesting to pursue is to use 3D wavelets as features. Such a system would learn the dynamics as a set of displacements and therefore may generalize better.

6.2 Unimodal Density Tracking

This section describes the extension to the static system that incorporates a Kalman filter-like heuristic to track and predict detections in video sequences using unimodal Gaussian densities.

6.2.1 Kalman Filter

Kalman filters are recursive algorithms that provide optimal estimates of a system's parameters by incorporating a linear system model prediction with actual measurements. The basic Kalman filter emerged from classical optimal estimation theory and has been heavily used in many fields. For background and derivations of the Kalman filter see [Maybeck, 1982].

The goal of the Kalman filter algorithm is to, at time t_{k-1}, estimate or predict the state of the process one time step in the future, t_k, based on the underlying dynamics of the model captured in the state transition matrix, and then, upon receiving the sensory measurements at the next time step t_k, correct the prediction to yield the optimal estimate (in a linear sense) of the state at time t_k. In the context of tracking and prediction in computer vision, Kalman filters have been widely used ([Broida and Chellappa, 1986, Dickmanns and Graefe, 1988, Deriche and Faugeras, 1990, Harris, 1992]). In a Kalman filter tracking situation, a Kalman filter is typically initialized over each object of interest. Over time, the Gaussian density function that is over each object shifts, spreads, and peaks depending on the reinforcement it receives from the measurement model.

6.2.2 Poor Man's Kalman (PMK)

As a zeroth order approximation to Kalman prediction and tracking for our person detection/tracking task, we approximate a Kalman filter by using a simple rule-based heuristic, called the Poor Man's Kalman (PMK).

Our heuristic smooths the information in an image sequence over time by taking advantage of the fundamental *a priori* knowledge that a person in one image will appear in a similar position in the next image. We smooth the results through time by automatically eliminating false positives, detections that do not persevere over a small subsequence.

The rule is extremely simple and assumes we are looking at sequences of 3 consecutive frames ($k-1$, k, $k+1$):

> `If a pattern is labeled as a person in fewer than` $\frac{n}{2}$ `times in an` n`-frame subsequence, then we expect that this pattern is a false positive since true detections persevere through time, so eliminate that detection.`

If a pattern is labeled as a person less than $\frac{n}{2}$ times in any n-frame subsequence, than we relabel that pattern as a non-person, thereby eliminating that presumed false positive. For our tests, we use $n = 3$. Since the people and camera may be moving, we allow some tolerance in the overlap of patterns. For a given tolerance level t and assuming a pattern that has been rescaled to the base 128×64 size (containing people of size 96×32), we allow a $32t$ pixel tolerance in the x positions of the top-left and bottom-right locations and a $96t$ pixel tolerance in the y positions. The tolerances we

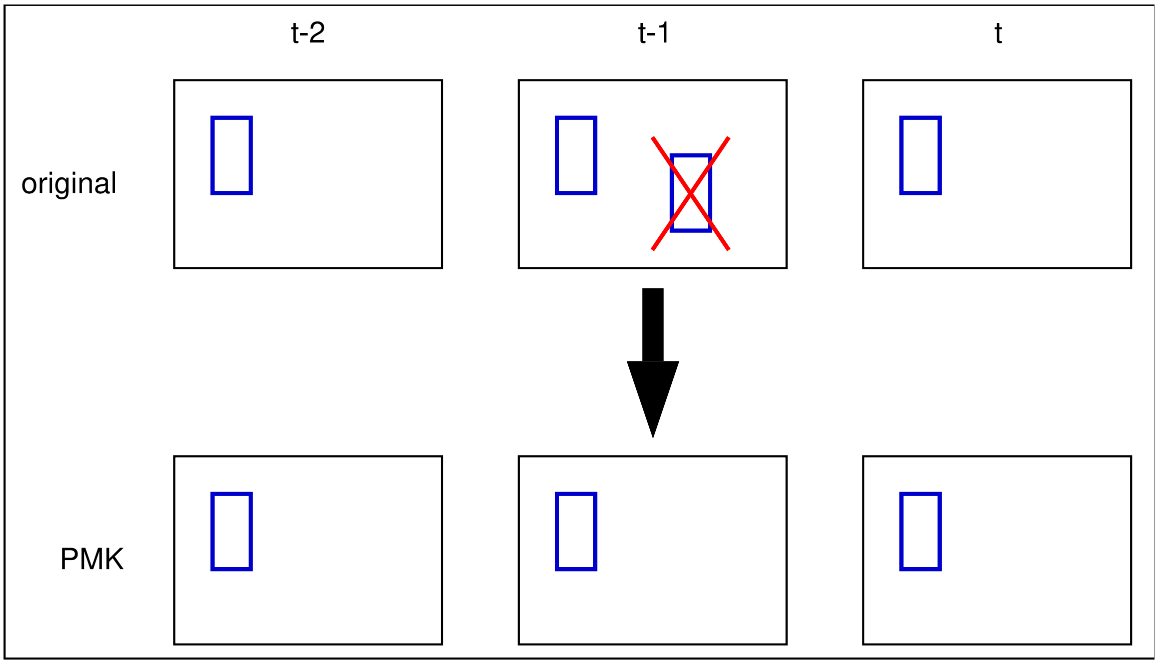


Figure 6-3: An illustration of how the Poor Man's Kalman heuristic would eliminate a detection that only appears at a certain location for one frame in a three frame subsequence. The top figures indicate the raw hypothesized detections. After processing using the PMK technique, the detection that does not persist is eliminated from the final output.

test are $t = \{0.1, 0.2, 0.3\}$. The effect that this technique has is illustrated in Figure 6-3.

Note that a rule as simple as this will fail when there are large motions of either the camera or people. This problem could be solved by, instead of just looking at a single location in consecutive images, trying to predict linear motion in local regions. We will not address this problem, however.

To compare this system against the static version and facilitate comparisons with the other time integration approaches, we train on the 1,379 positive and 3,822 negative subsequences $(t-4, t-3, t-2, t-1, t)$ used to train the pure pattern classification approach. For a fair comparison with the other techniques that are more directly based on static detection, we train a static detection SVM with the 1,379 positive and 3,822 negative *individual images* corresponding to frame t in each of the subsequences. The resulting ROC curve for our PMK technique is shown in Figure 6-4. The ROC curve shows that, while at higher thresholds PMK is the same as or slightly outperformed by the static system, at lower thresholds PMK performs better. The reason behind this is due to the fact that when the core detection is strict, people patterns that lie close to the decision boundary may not be correctly classified. This could yield a subsequence where a person appears and then disappears, exactly the

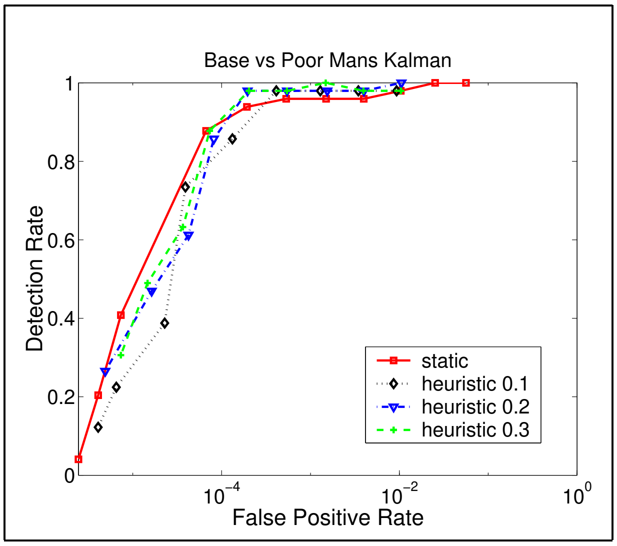


Figure 6-4: ROC curves for the static and Poor Man's Kalman detection systems. The numbers associated with each PMK curve indicates the tolerance for persevering detections.

conditions under which the PMK technique dictates the removal of the detection, which, in this case, is a correct person and not a false positive.

6.3 Multimodal Density Tracking

Kalman filtering has been used extensively to track objects in video sequences through time. The key assumption of the Kalman filter is that the state density is Gaussian. This means that a Kalman filter can only track a single object through time. One straightforward way of achieving multi-object tracking with Kalman filters is to instantiate a Kalman filter for each object that is being tracked. Though this would certainly work (as is shown in our PMK experiments), this strategy is not as appealing as a single unified framework that could track multiple objects. Such an algorithm, called Condensation, has been proposed by [Isard and Blake, 1998].

6.3.1 Condensation

The two components of Condensation are a model of the state dynamics and a measurement model. We denote the time history of the density function as $\mathbf{X}_t = \{\mathbf{x}_t, \mathbf{x}_{t-1}, \mathbf{x}_{t-2}, \ldots, \mathbf{x}_1\}$ and the time history of the measurement output as $\mathbf{Z}_t = \{\mathbf{z}_t, \mathbf{z}_{t-1}, \mathbf{z}_{t-2}, \ldots, \mathbf{z}_1\}$. We assume that the state dynamics can be modeled as a first order Markov chain,

$$P(\mathbf{x}_t|\mathbf{X}_{t-1}) = P(\mathbf{x}_t|\mathbf{x}_{t-1}) \tag{6.1}$$

The measurement component of the condensation model assumes that observations $\mathbf{z}_t$ are mutually independent and independent of the state dynamics. The observation process is therefore $P(\mathbf{z}|\mathbf{x})$. It is important to note that no other assumptions are being made about the observation process, so in the general case it could be multimodal.

The prediction or propagation of the densities to the next time step is done by applying Bayes rule using, as the prior for the current step, the posterior state density that has been transformed by one step of the underlying dynamical model:

$$P(\mathbf{x}_t|\mathbf{Z}_t) = k_t P(\mathbf{z}_t|\mathbf{x}_t) P(\mathbf{x}_t|\mathbf{Z}_{t-1}) \tag{6.2}$$

where

$$P(\mathbf{x}_t|\mathbf{Z}_{t-1}) = \int_{\mathbf{x}_{t-1}} P(\mathbf{x}_t|\mathbf{x}_{t-1}) P(\mathbf{x}_{t-1}|\mathbf{Z}_{t-1}) d\mathbf{x}_{t-1} \tag{6.3}$$

and k_t is a normalization constant. In our case, $\mathbf{x}$ indicates the position and scale of a person in the image and $\mathbf{z}$ is the support vector machine output at given positions and scales.

The main problem here is to somehow estimate and rebalance the multimodal state density through time as we obtain new measurements. Isard and Blake, 1998, propose a factored sampling algorithm to efficiently estimate these densities. They first sample points from the prior and apply the dynamical model to these points. The new points are then weighted in proportion to the measured features, i.e. in proportion to $P(\mathbf{z}|\mathbf{x})$. This weighted point set serves as an efficient approximation to the posterior $P(\mathbf{x}|\mathbf{z})$. Since we are concerned with accuracy in detection and tracking and not as concerned with efficiency, we will use a formulation of the Condensation algorithm that does not rely on a factored sampling approach to provide estimates of the state densities. Our version represents the density and takes measurements at each point in space.

Our version of this density propagation approach is as follows. Using the posterior density from the previous image in the sequence, we directly transfer this density to be the prior for the current image. This imposes an assumption of a trivial underlying dynamical model for people, $P(\mathbf{x}_t) = P(\mathbf{x}_{t-1})$. We run our person detection system over the current image and reinforce or inhibit the density in the locations where the support vector machine output is high and low respectively. Formally, we use estimates of the density $P(\mathbf{z}|\mathbf{x})$, the probability relating support vector machine outputs to confidence that there is a person at the location, that have been generated from a set of training data.

The result of this processing is that for each frame we have a posterior density reflecting the likelihood of the presence or absence of people at each location. To accomplish actual detection, we threshold these densities.

The derivation of the density in Equation 6.2 is summarized here:

$$
\begin{aligned}
P(\mathbf{x}_t|\mathbf{Z}_t) &= P(\mathbf{x}_t|\mathbf{z}_t, \mathbf{Z}_{t-1}) && (6.4)\\
&= \frac{P(\mathbf{z}_t|\mathbf{x}_t, \mathbf{Z}_{t-1})P(\mathbf{x}_t|\mathbf{Z}_{t-1})}{P(\mathbf{z}_t|\mathbf{Z}_{t-1})} \quad \text{(by Bayes Rule)} && (6.5)\\
&= \frac{P(\mathbf{z}_t|\mathbf{x}_t, \mathbf{Z}_{t-1})P(\mathbf{x}_t|\mathbf{Z}_{t-1})}{P(\mathbf{z}_t)} \quad \text{(by independence of } \mathbf{z}_i\text{)} && (6.6)\\
&= \frac{P(\mathbf{z}_t|\mathbf{x}_t)P(\mathbf{x}_t|\mathbf{Z}_{t-1})}{P(\mathbf{z}_t)} \quad \text{(since } \mathbf{x}_t \text{ already encapsulates all information about } \mathbf{Z}_{t-1}\text{)} && (6.7)\\
&= k_t P(\mathbf{z}_t|\mathbf{x}_t)P(\mathbf{x}_t|\mathbf{Z}_{t-1}) \quad \text{(since } P(\mathbf{x}_t|\mathbf{Z}_t) \text{ is a distribution that integrates to 1, we can normalize by } k_t\text{)} && (6.8)
\end{aligned}
$$

Furthermore, we are assuming a trivial dynamical model in which the prediction of a person's position in the next frame is simply the location in the current frame. This means:

$$P(\mathbf{x}_t|\mathbf{Z}_{t-1}) = P(\mathbf{x}_{t-1}|\mathbf{Z}_{t-1}) \tag{6.9}$$

so:

$$P(\mathbf{x}_t|\mathbf{Z}_t) = k_t P(\mathbf{z}_t|\mathbf{x}_t)P(\mathbf{x}_{t-1}|\mathbf{Z}_{t-1}) \tag{6.10}$$

The effect of this type of processing is illustrated in Figure 6-5 where the density is visualized in two dimensions. Unlike [Isard and Blake, 1998] who in effect obtain measurements from a fixed number of locations in the image, we obtain and measure candidate "features" at every point in the new image.

Since we are assuming a trivial dynamical model, the only quantity in Equation 6.10 that is of any real interest is the measurement process $P(\mathbf{z}_t|\mathbf{x}_t)$. This function will propagate the density based on the output of the support vector machine classifier at each location. The problem here is that the SVM output is in the form of the distance of the pattern from the separating hyperplane and not a probability on $\mathbf{x}$. The next section addresses this issue.

6.3.2 Estimating $P(\mathbf{z}|\mathbf{x})$

In words, the measurement process $P(\mathbf{z}_t|\mathbf{x}_t)$ denotes the probability of a certain SVM output conditional on the distribution of $\mathbf{x}$, the people at each location. We base our measurement process on the SVM output at each location. This value is a distance from the hyperplane that separates the people from the non-people. Assuming we have already estimated $P(\mathbf{z}|\mathbf{x})$, at each location in the image we take the output of

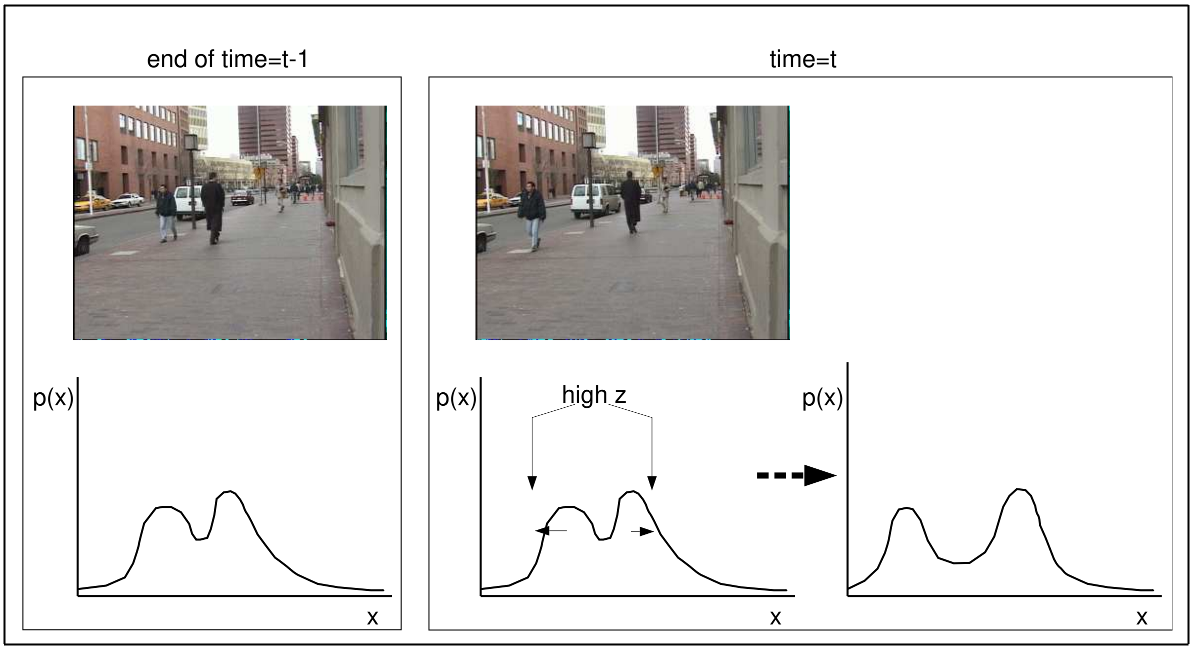


Figure 6-5: An illustration of Condensation in action. The posterior density from $t-1$ is used as the prior a time t. This prior is then acted on by the effect of the SVM measurements at time t; strong SVM output z reinforces the prior and shifts the density. The result of this propagation is the rightmost density, the posterior for time t.

the SVM and compute this likelihood. To counter the effects of possibly non-smooth SVM outputs in our results, we average $P(\mathbf{z}|\mathbf{x})$ over local regions.

One of the key assumptions is that the output of the SVM peaks over a person and then relatively *smoothly* decays as we move away from the pattern both in (x, y) and in scale. Figure 6-6 shows that the support vector machine raw output does indeed decay fairly smoothly. We have not developed our system to be shift invariant but in fact there is a small amount of shift invariance that the system seems to learn from the training set, possibly due to small misalignments in the data.

To transform this raw distance value into a distribution conditional on $\mathbf{x}$, we use a set of out-of-sample images that have been manually annotated with the locations of the people $\mathbf{x}^{(i)}$. At each location in the image $\mathbf{x}^{(j)}$, we obtain the SVM output for that pattern $\mathbf{z}^{(j)}$. This gives us a version of the density that reflects distances to people, i.e. $P(z^{(j)}|d(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}))$.

When running over a new image, we use this as follows. Having computed the SVM output for a given pattern, we can compute $P(z^{(i)}|\mathbf{x})$ by smoothing over a local neighborhood indexed as A_N ($|A_N| = N$) by:

$$P(z^{(i)}|\mathbf{x}) = \frac{1}{N} \sum_{j \epsilon A_N} P\left(z^{(j)}|d(\mathbf{x}^{(i)}, \mathbf{x}^{(j)})\right) \tag{6.11}$$

This gives us the likelihood that serves to reinforce the prior in areas where there are people.

As our function $d(\mathbf{a}, \mathbf{b})$, we need to use a measure that takes into account the differences in both position and scale between $\mathbf{a}$ and $\mathbf{b}$. We use:

$$d(\mathbf{a}, \mathbf{b}) = (\mathbf{a}_x^{tl} - \mathbf{b}_x^{tl}) + (\mathbf{a}_y^{tl} - \mathbf{b}_y^{tl}) + (\mathbf{a}_x^{br} - \mathbf{b}_x^{br}) + (\mathbf{a}_y^{br} - \mathbf{b}_y^{br}) \tag{6.12}$$

where the superscripts tl and br indicate top-left and bottom-right coordinates respectively and the subscripts x and y denote the x and y components of the coordinates. This is a very simple measure that fits our criteria of encapsulating differences in both distance and scale. Other more complex measures of distance could be used but we find that this one works effectively.

The motivation for the Condensation algorithm is to propagate *densities* through time, not identify distinct locations where an object lies. In other words, the goal is to track densities, not to do pure pattern classification. This leads to a problem when applying it to a detection task in dealing with objects with unequal probability masses.

If there are two people in the scene and each is detected "equally well" (same number of detections with the same strength) by the underlying support vector machine, then the Condensation framework will assign equally strong probability masses to the two people. However, it is extremely unlikely that all the people in the scene will be detected with equal strength. This leads the Condensation algorithm to assign probability masses of varying size to different people. In turn, this means that thresholding the densities to localize the objects of interest may miss weaker peaks in the density.

Our solution to this is to first do a hill climbing search in the density to find the distinct peaks. Then, we individually threshold the peaks by specifying that a constant mass around each peak indicates a detection, which amounts to labeling the closest points to each peak as a detection. Figure 6-7 illustrates this concept.

6.3.3 Results

We train and test the system in the same manner as the other time integration systems. The performance of the Condensation-based approach as compared to the static system is shown in Figure 6-8. The curves show that the Condensation-based technique significantly outperforms the static version in the "low false positive" portion of the ROC curve. This is not a surprising result, since the effect of false positives are magnified by our taking a constant volume as the detected patterns. When there are few false positives, the detected patterns are more likely to be people and these detections will persevere on account of the propagation of the density.

6.4 Discussion of Time Integration Techniques

It is of interest to compare the different time integration techniques and discuss their relative merits and drawbacks.

6.4.1 Pattern Classification Approach

The multiframe pattern classification approach is the one most directly related to the core static system and is the system that makes the least number of assumptions about dynamical models. Here, we assume that any of the dynamics are captured in the five frame subsequences. The SVM learning engine is able to learn in the 6,630 dimensional input space and is able to generalize quite well as seen in the ROC curve. The main problem with our particular version is that the training data seems to be heavily biased to left moving people.

At run time, this system is quite expensive. The core classification step for 1 window uses $5ns$ multiplications where n is the number of features in a single window and s is the number of support vectors. For an average image size of 240×360 with approximately 30,000 windows, this leads to $150,000ns$ multiplications per image with 1 access per pattern.

6.4.2 Poor Man's Kalman Heuristic

The PMK heuristic approximation to a Kalman filter is the simplest addition to the core static approach to take advantage of information over time. This module assumes no underlying dynamics but takes advantage of the fact that the video sequences we typically use are at a high enough rate that people do not exhibit significant changes in position and scale from one frame to another. In fact, this is why our simple distance measure, essentially using the L_1 norm in position and scale, works effectively. In domains where the assumption of smooth motion is not valid, this approach would not work. Cases like these are better served by an approach that is able to more directly capture direction and velocity in the model.

The run time complexity of this system is ns multiplications per window with 1 access per pattern, assuming the proper data structures. In all, there are $30,000ns$ multiplications per image for the core SVM computation.

6.4.3 Condensation-Based Approach

The Condensation-based extension to the static system is the most elegant, but involves significantly more development and changes to the system. One of the main issues is the estimation of the likelihood, $P(\mathbf{z}|\mathbf{x})$, which is non-trivial. In our case, this likelihood can be estimated from a set of data.

At run time, the Condensation-based approach is quite a bit more complex than the other techniques. For classification, each pattern needs ns multiplications with an additional 1 multiplication for propagating the prior (here, too, there are $O(10)$ additions per window since to compute the likelihood we average over a small neighborhood) and 1 division to normalize the density. This, combined with the necessary peak finding step, means $O(10)$ visits per pattern. Though more complex than the other approaches, Condensation seems to be the method of choice due to its flexibility in modeling multimodal densities. In addition to the computational complexity, another minor drawback is that the framework is developed for tracking and not pure detection; our peak finding heuristic is one simple way to circumvent this.

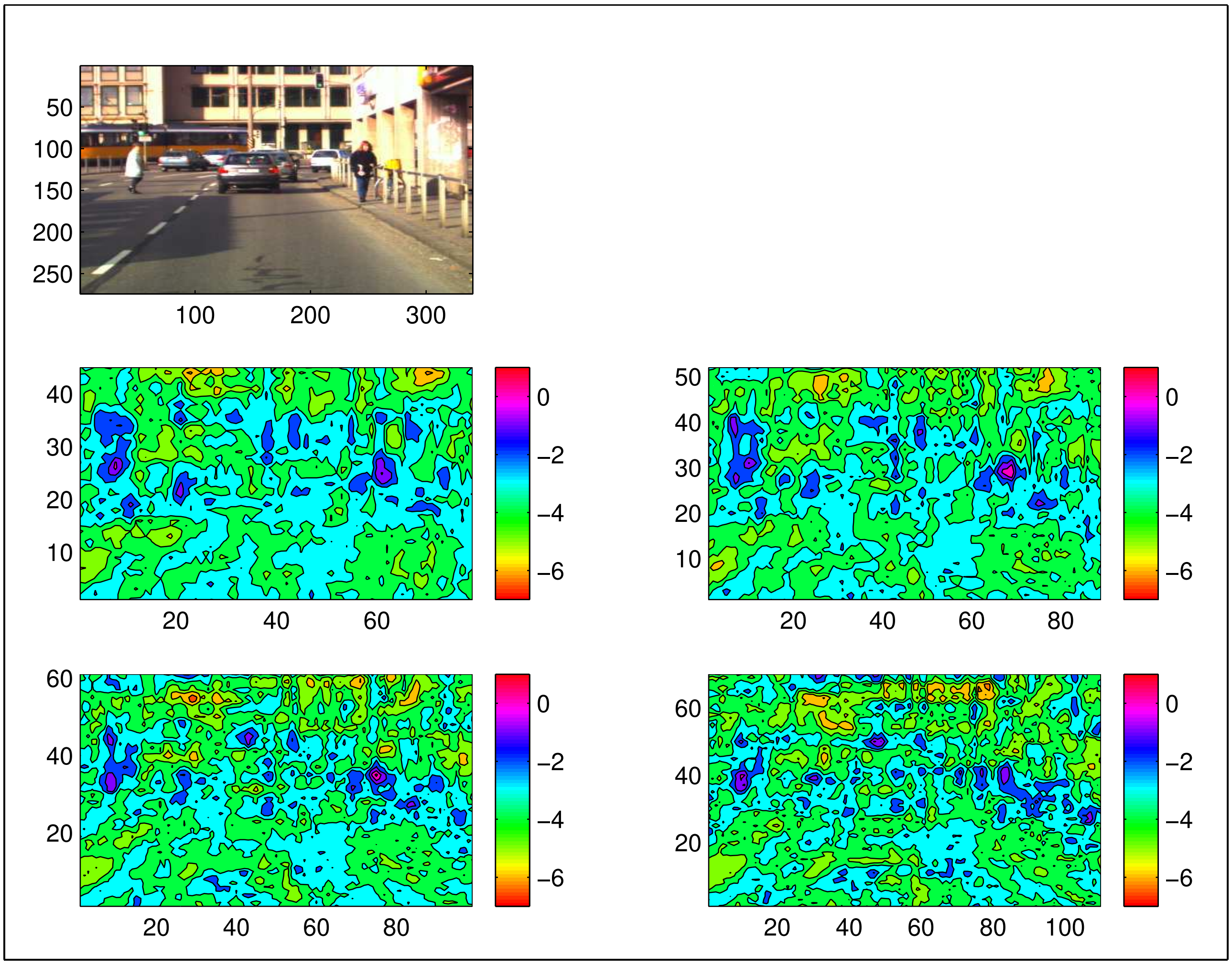


Figure 6-6: The top image is the original and the bottom images are the color coded raw support vector machine outputs from the image when processed for four different scales of people. The output of the SVM peaks over the people patterns and decays smoothly as we shift the location and scale of the pattern.

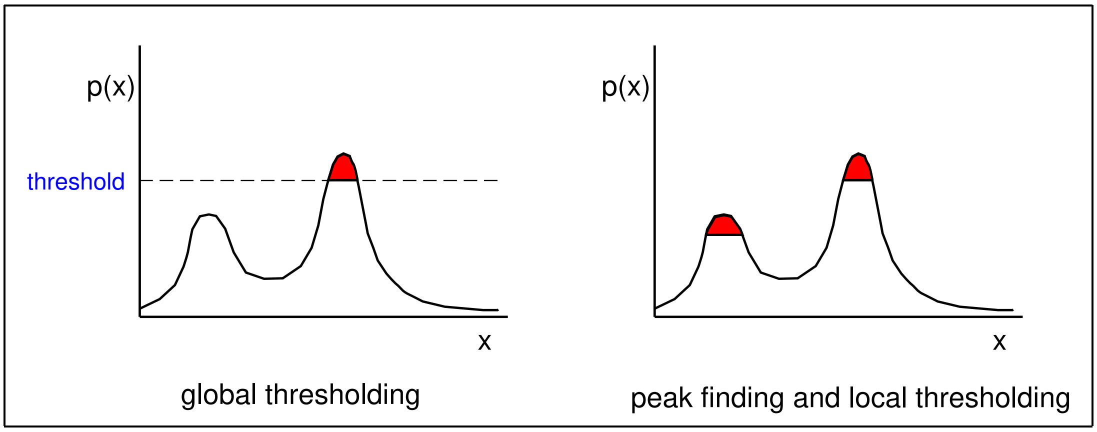


Figure 6-7: Global thresholding may miss a peak due to unequally weighted portions of the density. To correct for this, we do a hill climbing search to find the distinct peaks and then locally threshold the portions of the density around the peaks.

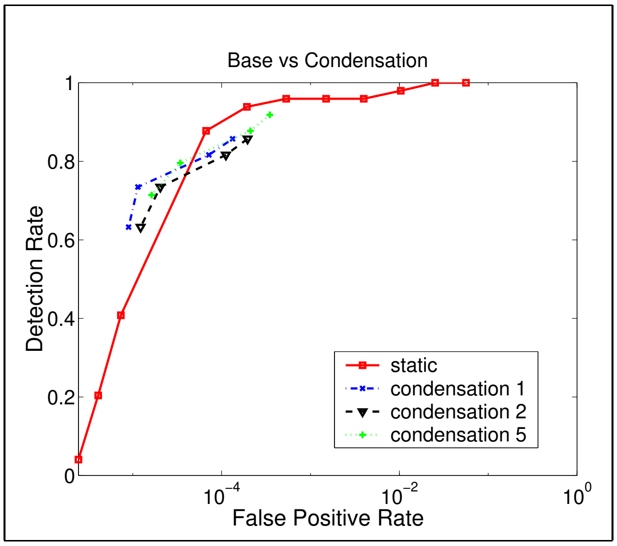


Figure 6-8: ROC curves for the static and Condensation-based detection systems. The numbers associated with the Condensation curves are the number of points that the algorithm uses at the peaks of the density.

Chapter 7

Conclusion

This thesis has presented a general trainable framework for object detection in static images that is successfully applied to face, people, and car detection in static images. Along with some extensions to video sequences, we have investigated different representations, a component-based approach, and implemented a real-time version of the system that is running in a DaimlerChrysler experimental vehicle. The system is robust, portable, and can be made efficient. While we have not pushed this system to be the "best" detection system in a particular domain, though this certainly may be possible, we have shown its applicability to a wide range of object classes. We feel that in practical uses of this system, the proper architecture should combine a focus of attention module with our static detection system. We have presented preliminary results here as well.

Future work around this system should focus on the following:

- component-based techniques: In the component-based system we present, the parts and geometries are manually specified. An important next step is to develop a method for automatically selecting the components from a large set of possible components.

- time integration: This thesis has presented some promising first steps in augmenting the core static detection system with different modules that take advantage of dynamical information. These techniques and others should be investigated further and extended. One interesting direction would be to develop a system that used wavelets in space **and time**.

- representations: One of the big leaps we have taken in this thesis is our choice of representation, Haar wavelets. While we have shown that this wavelet representation achieves excellent results when compared to other representations, there are many possible other feature sets that could be used. Subsequent research in this area is imperative.

- support vector machines: Though support vector machines are well-founded in statistics and heavily studied these days, there are a couple of open questions whose solutions would benefit our work as well. First, how can we quickly train support vector machines with very large data sets? Second, are there ways of using support vector machines to do feature selection in a principled manner? We present one idea here, but this is largely a heuristic first step.

Appendix A

Wavelets

A.1 The Haar Wavelet

Wavelets provide a natural mathematical structure for describing our patterns; a more detailed treatment can be found in [Mallat, 1989]. These vector spaces form the foundations of the concept of a multiresolution analysis. We formalize the notion of a multiresolution analysis as the sequence of approximating subspaces $V^0 \subset V^1 \subset V^2 \subset \ldots V^j \subset V^{j+1} \ldots$; the vector space V^{j+1} can describe finer details than the space V^j, but every element of V^j is also an element of V^{j+1}. A multiresolution analysis also postulates that a function approximated in V^j is characterized as its orthogonal projection on the vector space V^j.

As a basis for the vector space V^j, we use the *scaling functions*,

$$\phi_i^j = \sqrt{2^j}\phi(2^j x - i), i = 0, \ldots, 2^j - 1, \tag{A.1}$$

where, for our case of the Haar wavelet,

$$\phi(x) = \begin{cases} 1 & for\ 0 \leq x < 1 \\ 0 & otherwise \end{cases} \tag{A.2}$$

Next we define the vector space W^j that is the orthogonal complement of two consecutive approximating subspaces, $V^{j+1} = V^j \bigoplus W^j$. The W^j are known as *wavelet subspaces* and can be interpreted as the subspace of "details" in increasing refinements. The wavelet space W^j is spanned by a basis of functions,

$$\psi_i^j = \sqrt{2^j}\psi(2^j x - i), i = 0, \ldots, 2^j, \tag{A.3}$$

where for Haar wavelets,

$$\psi(x) = \begin{cases} 1 & for\ 0 \leq x < \frac{1}{2} \\ -1 & for\ \frac{1}{2} \leq x < 1 \\ 0 & otherwise \end{cases} \tag{A.4}$$

The sum of the wavelet functions form an orthonormal basis for $L_2(R)$. It can be shown (under the standard conditions of multiresolution analysis) that all the scaling functions can be generated from dilations and translations of one scaling function. Similarly, all the wavelet functions are dilations and translations of the mother wavelet function. Figure 2-5a shows the scaling and wavelet functions. The approximation of some function $f(x)$ in the space V^j is found to be:

$$A_j f = \sum_{k \in Z} \overbrace{< f(u), \phi_k^j(u) >}^{\lambda_{j,k}} \phi_k^j(x) \tag{A.5}$$

and, similarly, the projection of $f(x)$ on W^j is:

$$D_j f = \sum_{k \in Z} \overbrace{< f(u), \psi_k^j(u) >}^{\gamma_{j,k}} \psi_k^j(x) \tag{A.6}$$

The structure of the approximating and wavelet subspaces leads to an efficient cascade algorithm for the computation of the scaling coefficients, $\lambda_{j,k}$, and the wavelet coefficients, $\gamma_{j,k}$:

$$\lambda_{j,k} = \sum_{n \in Z} h_{n-2k} \lambda_{j+1,n} \tag{A.7}$$

$$\gamma_{j,k} = \sum_{n \in Z} g_{n-2k} \lambda_{j+1,n} \tag{A.8}$$

where $\{h_i\}$ and $\{g_i\}$ are the filter coefficients corresponding to the scaling and wavelet functions. Using this construction, the approximation of a function $f(x)$ in the space V^j is:

$$A_j f = \sum_{n \in Z} \lambda_{j,k} \sqrt{2^j} \phi(2^j x - k) \tag{A.9}$$

Similarly, the approximation of $f(x)$ in the space W^j is:

$$D_j f = \sum_{n \in Z} \gamma_{j,k} \sqrt{2^j} \psi(2^j x - k) \tag{A.10}$$

Since we use the Haar wavelet, the corresponding filters are: $h = \{\ldots, 0, \frac{1}{2}, \frac{1}{2}, 0, 0, \ldots\}$ and $g = \{\ldots, 0, -\frac{1}{2}, \frac{1}{2}, 0, 0, \ldots\}$ The scaling coefficients are simply the averages of pairs of adjacent coefficients in the coarser level while the wavelet coefficients are the differences.

It is important to observe that the discrete wavelet transform (DWT) performs *downsampling* or *decimation* of the coefficients at the finer scales since the filters h and g are moved in a step size of 2 for each increment of k.

A.2 2-Dimensional Wavelet Transform

The natural extension of wavelets to 2D signals is obtained by taking the tensor product of two 1D wavelet transforms. The result is the three types of wavelet basis functions shown in Figure 2-5. The first type of wavelet is the tensor product of a wavelet by a scaling function, $\psi(x,y) = \psi(x) \otimes \phi(y)$; this wavelet encodes a difference in the average intensity along a vertical border and we will refer to its value as a *vertical* coefficient. Similarly, a tensor product of a scaling function by a wavelet, $\psi(x,y) = \phi(x) \otimes \psi(y)$, is a *horizontal* coefficient, and a wavelet by a wavelet, $\psi(x,y) = \psi(x) \otimes \psi(y)$, is a *diagonal* coefficient since this wavelet responds strongly to diagonal boundaries.

Since the wavelets that the standard transform generates have irregular support, we use the non-standard 2D DWT where, at a given scale, the transform is applied to each dimension sequentially before proceeding to the next scale (Stollnitz *et al.*, 1994[Stollnitz *et al.*, 1994]). The results are Haar wavelets with square support at all scales, shown in Figure 2-5b.

A.3 Quadruple Density Transform

For the 1D Haar transform, the distance between two neighboring wavelets at level n (with support of size 2^n) is 2^n. To obtain a denser set of basis functions that provide better spatial resolution, we need a set of redundant basis functions, or an overcomplete *dictionary*, where the distance between the wavelets at scale n is $\frac{1}{4}2^n$ (Figure 2-5c). The straightforward approach of shifting the signal and recomputing the DWT will not generate the desired dense sampling. Instead, this can be achieved by modifying the DWT. To generate wavelets with *double density*, where wavelets of level n are located every $\frac{1}{2}2^n$ pixels, we simply do not downsample in Equation A.8. To generate the *quadruple density* dictionary, first, we do not downsample in Equation A.7, giving us double density scaling coefficients. Next, we calculate double density wavelet coefficients on the two sets of scaling coefficients — even and odd — separately. By interleaving the results of the two transforms we get quadruple density wavelet coefficients. For the next scale $(n+1)$, we keep only the even scaling coefficients of the previous level and repeat the quadruple transform on this set only; the odd scaling coefficients are dropped off. Since only the even coefficients are carried along at all the scales, we avoid an "explosion" in the number of coefficients, yet obtain a dense and uniform sampling of the wavelet coefficients at all the scales. As with the regular DWT, the time complexity is $O(n)$ in the number of pixels n. The extension of the quadruple density transform to 2D is straightforward.

Appendix B

Support Vector Machines

The second key component of our system is the use of a trainable pattern classifier that learns to differentiate between patterns in our object class and all other patterns. In general terms, these supervised learning techniques rely on having a set of labeled example patterns from which they derive an implicit model of the domain of interest. The particular learning engine we use is a support vector machine (SVM) classifier.

The support vector machine algorithm is a technique to train classifiers that is well-founded in statistical learning theory [Vapnik, 1995, Burges, 1998, Vapnik, 1998]. Here, we provide some of the mathematical and practical aspects of support vector machines that are relevant to our work.

B.1 Theory and Mathematics

For a given learning task, we have a set of ℓ N-dimensional labeled training examples:

$$(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_\ell, y_\ell)\ \mathbf{x}_i \in R^N, y_i \in \{-1, +1\} \tag{B.1}$$

where the examples have been generated from some unknown pdf, $P(\mathbf{x}, y)$. We would like the system to learn a decision function $f_a : \mathbf{x} \to y$ that minimizes the expected risk,

$$R(\alpha) = \int |f_\alpha(\mathbf{x}) - y| dP(\mathbf{x}, y) \tag{B.2}$$

In most cases, we will not know $P(\mathbf{x}, y)$; we simply see the data points that the distribution has generated. Thus, direct minimization of Equation B.2 is not possible. What we are able to directly minimize is the empirical risk, the actual error over the training set,

$$R_{emp}(\alpha) = \frac{1}{\ell} \sum_{i=1}^{\ell} |f_\alpha(\mathbf{x}_i) - y_i| \tag{B.3}$$

This is exactly the functional that many training techniques for classifiers minimize, but can lead to overfitting the training data and poor generalization. For these reasons, we introduce the theory of uniform convergence in probability:

$$R(\alpha) \leq R_{emp}(\alpha) + \Phi\left(\frac{h}{\ell}, \frac{-log(\eta)}{\ell}\right) \tag{B.4}$$

with probability $1 - \eta$. Here, $R(\alpha)$ is the expected risk; $R_{emp}(\alpha)$ is the empirical risk; ℓ is the number of training examples; h is the VC dimension of the classifier that is being used; and $\Phi(\cdot)$ is the VC confidence of the classifier. Intuitively, what this means is that the uniform deviation between the expected risk and empirical risk decreases with larger amounts of training data ℓ and increases with the VC dimension h. This leads us directly to the principle of structural risk minimization, where we can minimize both the actual error over the training set and the complexity of the classifier at the same time; this will bound the generalization error as in Equation B.4. It is exactly this technique that support vector machines approximate.

In the simple case of finding the optimal linear hyperplane $(\mathbf{w}, b)$ that separates two separable classes, this problem is equivalent to solving:

$$\underset{\mathbf{w}, b}{\text{minimize}} \quad \frac{1}{2} \parallel \mathbf{w} \parallel^2 \tag{B.5}$$

subject to the constraints:

$$y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \quad \geq 1 \quad i = 1 \ldots \ell \tag{B.6}$$

Typically, the dual formulation of this quadratic programming problem is solved, leading to a decision surface of the form:

$$f(\mathbf{x}) = \theta\left(\sum_{i=1}^{\ell} \alpha_i y_i (\mathbf{x} \cdot \mathbf{x}_i) + b\right) \tag{B.7}$$

where α_i are Lagrange variables.

To extend this to the more general case of linearly non-separable data, we add in slack variables ξ and a cost C that penalizes misclassifications:

$$\underset{\mathbf{w}, b}{\text{minimize}} \quad \frac{1}{2} \parallel \mathbf{w} \parallel^2 + C\left(\sum_{i=1}^{\ell} \xi_i\right)^k \tag{B.8}$$

subject to the constraints:

$$\begin{array}{lll} y_i(\mathbf{w} \cdot \mathbf{x}_i + b) & \geq 1 - \xi_i & i = 1 \ldots \ell \\ \xi_i & \geq 0 & i = 1 \ldots \ell \end{array} \tag{B.9}$$

Linear hyperplanes are a fairly restrictive set of decision surfaces, so ultimately we would like to use nonlinear decision surfaces. In this case, support vector machines work by projecting the input data into a higher dimensional feature space and finding the optimal linear separating hyperplane in this space. The data is mapped according to some function ϕ, as:

$$\mathbf{x} \rightarrow \phi(\mathbf{x}) = (\phi_1(\mathbf{x}), \phi_2(\mathbf{x}), \ldots, \phi_n(\mathbf{x})) \tag{B.10}$$

This leads to decision surfaces of the form:

$$f(\mathbf{x}) = \theta\left(\sum_{i=1}^{\ell} \alpha_i y_i (\phi(\mathbf{x}) \cdot \phi(\mathbf{x}_i)) + b\right) \tag{B.11}$$

A more compact representation is possible by introducing the nonlinear kernel function K:

$$K(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{x}) \cdot \phi(\mathbf{y}) \tag{B.12}$$

With this formulation, we obtain decision surfaces of the form:

$$f(\mathbf{x}) = \theta\left(\sum_{i=1}^{\ell} \alpha_i y_i K(\mathbf{x}, \mathbf{x}_i) + b\right) \tag{B.13}$$

One of the important characteristics of the solution is that there are typically only a small number of nonzero α_i. The separating hyperplane is therefore a linear combination of a small set of data points, called support vectors. Removing the non-support vectors and training again would yield exactly the same solution.

Using the SVM formulation, the classification rule for a pattern $\mathbf{x}$ using a polynomial of degree two, the classifier we use in our detection system, is as follows:

$$f(\mathbf{x}) = \theta\left(\sum_{i=1}^{N_s} \alpha_i y_i (\mathbf{x} \cdot \mathbf{x}_i + 1)^2 + b\right) \tag{B.14}$$

where N_s is the number of support vectors.

This controlling of both the training set error *and* the classifier's complexity has allowed support vector machines to be successfully applied to very high dimensional learning tasks; [Joachims, 1997] presents results on SVMs applied to a 10,000 dimensional text categorization problem and [Osuna *et al.*, 1997b] show a 283 dimensional face detection system.

B.2 Practical Aspects

The SVM package we use is that of Osuna et al., 1997 described in [Osuna *et al.*, 1997a, Osuna *et al.*, 1997b] that uses the MINOS quadratic programming solver [Murtagh and Saunders, 1995]. To train a support vector machine, we first need a set of training data. The process by which we gathered our training set is described in Section 2.2. We transform each training image into a feature vector of Haar wavelet coefficients as in Section 2.3.2. The data is then ready to be used for training.

In the SVM framework, there are essentially only 2 tunable parameters: the type of classifier (e.g. linear, polynomial, etc.) and C, the penalty for misclassifications.

Most of our experiments are run using a polynomial of degree 2 as in Equation B.14. In penalizing misclassifications, we use $C_{pos} = 100$ for the positive examples and $C_{neg} = 10$ for the negative examples to reflect the relative importance of having a low false negative rate: false negatives are penalized 10 times more than false positives.

Training time with the above settings is under 2 hours for our detection systems on either an SGI Reality Engine or a 450MHz PC running Linux. The output of the SVM training is a binary file containing the data for the decision surface: α_i, $\mathbf{x}_i$, y_i, for data points i that have non-zero α_i, and b. This data file is subsequently used by the detection system.

Appendix C

Raw Wavelet Data

Here we present the raw wavelet responses, averaged over the set of 1,848 people images and presented in their proper spatial location. The meaningful coefficients are those with values much larger or smaller than 1. Average values close to 1 indicate neither the presence nor the absence of an intensity difference in the average, i.e. these are inconsistent features.

0.7	0.7	0.7	0.7	0.7	0.7	0.7	0.7	0.7	0.7	0.7	0.7	0.7
0.7	0.7	0.7	0.7	0.8	0.8	0.7	0.8	0.8	0.7	0.7	0.7	0.7
0.7	0.7	0.7	0.8	0.9	0.9	0.9	0.9	0.9	0.8	0.7	0.7	0.7
0.7	0.7	0.7	0.8	1.1	1.2	1.0	1.2	1.1	0.8	0.7	0.7	0.7
0.7	0.7	0.8	1.0	1.4	1.5	1.2	1.5	1.4	1.0	0.8	0.7	0.7
0.7	0.7	0.8	1.1	1.5	1.4	1.1	1.4	1.5	1.1	0.8	0.7	0.7
0.7	0.8	1.0	1.3	1.4	1.2	0.9	1.2	1.4	1.3	1.0	0.8	0.7
0.7	0.9	1.3	1.5	1.4	1.0	0.8	1.0	1.4	1.5	1.3	0.9	0.7
0.8	1.1	1.6	1.7	1.3	0.9	0.7	0.9	1.3	1.7	1.6	1.1	0.8
0.8	1.3	1.8	1.7	1.2	0.9	0.7	0.9	1.2	1.7	1.8	1.3	0.8
0.9	1.4	1.9	1.6	1.2	0.8	0.7	0.8	1.2	1.6	1.9	1.4	0.9
0.9	1.5	1.9	1.6	1.2	0.8	0.6	0.8	1.2	1.6	1.9	1.5	0.9
0.9	1.4	1.8	1.7	1.2	0.8	0.6	0.8	1.2	1.7	1.8	1.4	0.9
0.9	1.3	1.7	1.7	1.3	0.8	0.6	0.8	1.3	1.7	1.7	1.3	0.9
0.8	1.1	1.5	1.8	1.3	0.8	0.6	0.8	1.3	1.8	1.5	1.1	0.8
0.7	0.9	1.4	1.9	1.5	0.8	0.6	0.8	1.5	1.9	1.4	0.9	0.7
0.6	0.8	1.3	1.9	1.6	0.9	0.7	0.9	1.6	1.9	1.3	0.8	0.6
0.6	0.7	1.2	1.9	1.8	1.0	0.8	1.0	1.8	1.9	1.2	0.7	0.6
0.5	0.6	1.1	1.9	1.8	1.1	0.8	1.1	1.8	1.9	1.1	0.6	0.5
0.5	0.6	1.0	1.8	1.8	1.2	0.9	1.2	1.8	1.8	1.0	0.6	0.5
0.5	0.6	1.0	1.6	1.8	1.2	0.9	1.2	1.8	1.6	1.0	0.6	0.5
0.5	0.6	0.9	1.5	1.8	1.3	0.9	1.3	1.8	1.5	0.9	0.6	0.5
0.5	0.6	0.8	1.4	1.8	1.3	0.9	1.3	1.8	1.4	0.8	0.6	0.5
0.5	0.6	0.7	1.3	1.7	1.2	0.9	1.2	1.7	1.3	0.7	0.6	0.5
0.5	0.5	0.7	1.2	1.5	1.2	0.8	1.2	1.5	1.2	0.7	0.5	0.5
0.4	0.5	0.6	1.0	1.3	1.0	0.7	1.0	1.3	1.0	0.6	0.5	0.4
0.4	0.4	0.5	0.7	1.0	0.8	0.6	0.8	1.0	0.7	0.5	0.4	0.4
0.3	0.3	0.4	0.5	0.7	0.6	0.4	0.6	0.7	0.5	0.4	0.3	0.3
0.2	0.2	0.3	0.3	0.4	0.4	0.3	0.4	0.4	0.3	0.3	0.2	0.2

Table C.1: Vertical 16×16 wavelets.

0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8
0.8	0.8	0.8	0.8	0.9	1.0	1.0	1.0	0.9	0.8	0.8	0.8	0.8
0.8	0.8	0.8	0.9	1.2	1.5	1.6	1.5	1.2	0.9	0.8	0.8	0.8
0.8	0.8	0.8	1.0	1.2	1.5	1.7	1.5	1.2	1.0	0.8	0.8	0.8
0.9	0.8	0.9	0.9	1.0	1.2	1.2	1.2	1.0	0.9	0.9	0.8	0.9
0.9	0.9	1.0	1.2	1.3	1.6	1.6	1.6	1.3	1.2	1.0	0.9	0.9
1.0	1.0	1.3	1.6	1.9	2.1	2.2	2.1	1.9	1.6	1.3	1.0	1.0
1.0	1.1	1.4	1.5	1.6	1.6	1.6	1.6	1.6	1.5	1.4	1.1	1.0
1.1	1.2	1.2	1.1	1.1	1.0	1.0	1.0	1.1	1.1	1.2	1.2	1.1
1.1	1.1	1.0	0.9	0.8	0.8	0.8	0.8	0.8	0.9	1.0	1.1	1.1
1.0	1.0	0.9	0.8	0.7	0.6	0.6	0.6	0.7	0.8	0.9	1.0	1.0
1.0	1.0	0.9	0.8	0.7	0.6	0.6	0.6	0.7	0.8	0.9	1.0	1.0
1.0	1.0	0.9	0.8	0.8	0.7	0.7	0.7	0.8	0.8	0.9	1.0	1.0
1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
1.0	1.1	1.1	1.1	1.2	1.2	1.3	1.2	1.2	1.1	1.1	1.1	1.0
1.0	1.1	1.2	1.2	1.3	1.4	1.4	1.4	1.3	1.2	1.2	1.1	1.0
0.9	1.0	1.1	1.2	1.3	1.3	1.4	1.3	1.3	1.2	1.1	1.0	0.9
0.9	1.0	1.1	1.1	1.1	1.2	1.2	1.2	1.1	1.1	1.1	1.0	0.9
0.9	0.9	1.0	0.9	0.9	1.0	1.0	1.0	0.9	0.9	1.0	0.9	0.9
0.9	0.9	0.9	0.8	0.8	0.9	0.9	0.9	0.8	0.8	0.9	0.9	0.9
0.9	0.9	0.9	0.8	0.8	0.9	0.9	0.9	0.8	0.8	0.9	0.9	0.9
0.9	0.9	0.9	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.9	0.9	0.9
0.9	0.9	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.9	0.9
0.9	0.9	0.9	0.8	0.9	0.9	0.9	0.9	0.9	0.8	0.9	0.9	0.9
0.9	0.9	0.9	0.9	0.9	1.0	1.1	1.0	0.9	0.9	0.9	0.9	0.9
0.8	0.9	0.9	0.9	1.0	1.2	1.3	1.2	1.0	0.9	0.9	0.9	0.8
0.9	1.0	1.0	1.0	1.1	1.3	1.4	1.3	1.1	1.0	1.0	1.0	0.9
0.8	0.8	0.9	1.0	1.1	1.3	1.4	1.3	1.1	1.0	0.9	0.8	0.8
0.7	0.7	0.8	0.9	1.0	1.1	1.2	1.1	1.0	0.9	0.8	0.7	0.7

Table C.2: Horizontal 16×16 wavelets.

0.7	0.7	0.7	0.7	0.7	0.7	0.7	0.7	0.7	0.7	0.7	0.7	0.7
0.7	0.7	0.7	0.8	0.9	1.0	0.9	1.0	0.9	0.8	0.7	0.7	0.7
0.7	0.7	0.8	1.0	1.4	1.6	1.4	1.6	1.4	1.0	0.8	0.7	0.7
0.7	0.7	0.8	1.1	1.5	1.7	1.5	1.7	1.5	1.1	0.8	0.7	0.7
0.8	0.8	0.8	1.0	1.3	1.4	1.3	1.4	1.3	1.0	0.8	0.8	0.8
0.8	0.8	1.0	1.4	1.6	1.7	1.5	1.7	1.6	1.4	1.0	0.8	0.8
0.8	1.0	1.6	1.9	1.8	1.9	1.6	1.9	1.8	1.9	1.6	1.0	0.8
0.9	1.2	1.7	1.6	1.7	1.6	1.2	1.6	1.7	1.6	1.7	1.2	0.9
0.9	1.3	1.4	1.5	1.5	1.1	0.9	1.1	1.5	1.5	1.4	1.3	0.9
0.9	1.2	1.3	1.3	1.1	0.8	0.7	0.8	1.1	1.3	1.3	1.2	0.9
0.9	1.1	1.2	1.1	0.9	0.7	0.6	0.7	0.9	1.1	1.2	1.1	0.9
0.9	1.1	1.2	1.1	1.0	0.7	0.6	0.7	1.0	1.1	1.2	1.1	0.9
0.9	1.1	1.2	1.2	1.1	0.8	0.6	0.8	1.1	1.2	1.2	1.1	0.9
0.8	1.1	1.2	1.3	1.2	0.9	0.7	0.9	1.2	1.3	1.2	1.1	0.8
0.8	1.1	1.3	1.5	1.4	0.9	0.8	0.9	1.4	1.5	1.3	1.1	0.8
0.8	1.1	1.3	1.6	1.5	1.0	0.8	1.0	1.5	1.6	1.3	1.1	0.8
0.8	1.0	1.2	1.5	1.4	1.0	0.9	1.0	1.4	1.5	1.2	1.0	0.8
0.7	0.8	1.1	1.3	1.3	1.0	0.9	1.0	1.3	1.3	1.1	0.8	0.7
0.6	0.7	0.9	1.1	1.2	1.1	0.8	1.1	1.2	1.1	0.9	0.7	0.6
0.5	0.6	0.8	1.0	1.2	1.0	0.8	1.0	1.2	1.0	0.8	0.6	0.5
0.6	0.7	0.8	1.1	1.2	1.0	0.8	1.0	1.2	1.1	0.8	0.7	0.6
0.6	0.6	0.8	1.1	1.1	1.0	0.8	1.0	1.1	1.1	0.8	0.6	0.6
0.6	0.7	0.9	1.1	1.1	1.0	0.8	1.0	1.1	1.1	0.9	0.7	0.6
0.6	0.7	0.9	1.1	1.1	1.0	0.9	1.0	1.1	1.1	0.9	0.7	0.6
0.6	0.7	0.8	1.1	1.2	1.0	0.9	1.0	1.2	1.1	0.8	0.7	0.6
0.6	0.6	0.8	1.0	1.2	1.1	1.0	1.1	1.2	1.0	0.8	0.6	0.6
0.6	0.6	0.7	1.0	1.3	1.2	1.0	1.2	1.3	1.0	0.7	0.6	0.6
0.5	0.5	0.6	0.9	1.1	1.0	0.8	1.0	1.1	0.9	0.6	0.5	0.5
0.3	0.3	0.4	0.6	0.7	0.7	0.6	0.7	0.7	0.6	0.4	0.3	0.3

Table C.3: Diagonal 16×16 wavelets.

0.7	0.7	0.6	0.7	0.7
0.8	0.9	0.6	0.9	0.8
1.0	1.1	0.6	1.1	1.0
1.3	1.2	0.6	1.2	1.3
1.6	1.3	0.5	1.3	1.6
1.6	1.3	0.5	1.3	1.6
1.5	1.3	0.5	1.3	1.5
1.3	1.4	0.5	1.4	1.3
1.3	1.4	0.5	1.4	1.3
1.1	1.3	0.5	1.3	1.1
0.9	1.2	0.4	1.2	0.9
0.7	1.0	0.4	1.0	0.7
0.5	0.7	0.3	0.7	0.5

Table C.4: Vertical 32×32 wavelets.

0.9	1.1	1.1	1.1	0.9
1.0	1.2	1.3	1.2	1.0
1.2	1.5	1.7	1.5	1.2
1.0	1.1	1.3	1.1	1.0
0.8	0.7	0.8	0.7	0.8
0.8	0.8	0.8	0.8	0.8
1.0	1.1	1.3	1.1	1.0
1.1	1.2	1.4	1.2	1.1
0.9	1.0	1.1	1.0	0.9
0.8	0.8	0.9	0.8	0.8
0.8	0.8	0.8	0.8	0.8
0.8	0.9	1.0	0.9	0.8
0.9	1.0	1.1	1.0	0.9

Table C.5: Horizontal 32×32 wavelets.

0.9	1.2	0.9	1.2	0.9
1.1	1.2	0.8	1.2	1.1
1.5	1.3	0.9	1.3	1.5
1.3	1.1	0.7	1.1	1.3
1.0	0.9	0.6	0.9	1.0
1.0	1.0	0.6	1.0	1.0
1.2	1.3	0.7	1.3	1.2
1.2	1.3	0.7	1.3	1.2
0.9	1.1	0.6	1.1	0.9
0.8	1.1	0.6	1.1	0.8
0.8	1.0	0.6	1.0	0.8
0.8	1.0	0.5	1.0	0.8
0.8	1.1	0.5	1.1	0.8

Table C.6: Diagonal 32×32 wavelets.

Bibliography

[Ali and Dagless, 1990] A. Ali and E. Dagless. Alternative proctical methods for moving object detection. In *IEE Colloquium on Electronic Images and Image Processing in Security and Forensic Science*, pages 6/1–6/7, 1990.

[Bauer and Kohavi, 1998] Eric Bauer and Ron Kohavi. An empirical comparison of voting classification algorithms: Bagging, boosting and variants. *Machine Learning*, 1998.

[Betke and Nguyen, 1998] M. Betke and H. Nguyen. Highway scene analysis form a moving vehicle under reduced visibility conditions. In *Proceedings of Intelligent Vehicles*, pages 131–136, October 1998.

[Betke *et al.*, 1997] M. Betke, E. Haritaoglu, and L. Davis. Highway scene analysis in hard real-time. In *Proceedings of Intelligent Transportation Systems*. IEEE, July 1997.

[Beymer *et al.*, 1997] D. Beymer, P. McLauchlan, B. Coifman, and J. Malik. A Real-time Computer Vision System for Measuring Traffic Parameters. In *Proceedings of Computer Vision and Pattern Recognition*, pages 495–501. IEEE Computer Society Press, 1997.

[Bishop, 1995] C. Bishop. *Neural Networks for Pattern Recognition*. Clarendon Press, 1995.

[Braithwaite and Bhanu, 1994] R. Braithwaite and B. Bhanu. Hierarchical gabor filters for object detection in infrared images. In *Proceedings of Computer Vision and Pattern Recognition*, pages 628–631, 1994.

[Breiman, 1996] L. Breiman. Bagging predictors. *Machine Learning*, 24:123–140, 1996.

[Broida and Chellappa, 1986] T. Broida and R. Chellappa. Estimation of Object Motion Parameters from Noisy Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(1):90–99, January 1986.

[Burges, 1996] C. Burges. Simplified Support Vector decision rules. In *Proceedings of 13th International Conference on Machine Learning*, 1996.

[Burges, 1998] C. Burges. A Tutorial on Support Vector Machines for Pattern Recognition. In Usama Fayyad, editor, *Proceedings of Data Mining and Knowledge Discovery*, pages 1–43, 1998.

[Burl and Perona, 1996] M. Burl and P. Perona. Recognition of planar object classes. In *Proceedings of Computer Vision and Pattern Recognition*, June 1996.

[Burl *et al.*, 1995] M. Burl, T. Leung, and P. Perona. Face localization via shape statistics. In *International Workshop on Automatic Face and Gesture Recognition*, June 1995.

[Campbell and Bobick, 1995] L. Campbell and A. Bobick. Recognition of human body motion using phase space constraints. Technical Report 309, MIT Media Laboratory, 1995.

[Chapa and Raghuveer, 1996] J. Chapa and M. Raghuveer. Matched wavelets - their construction, and application to object detection. In *Proceedings of International Conference on Acoustics, Speech, and Signal Processing*, pages 2987–2989, 1996.

[Colmenarez and Huang, 1996] A. Colmenarez and T. Huang. Maximum likelihood face detection. In *Proceedings of International Conference on Automatic Face and Gesture Recognition*, pages 307–311, 1996.

[Colmenarez and Huang, 1997] A. Colmenarez and T. Huang. Face detection with information-based maximum discrimination. In *Proceedings of Computer Vision and Pattern Recognition*, pages 782–787, 1997.

[Colmenarez and Huang, 1998] A. Colmenarez and T. Huang. Pattern detection with information-based maximum discrimination and error bootstrapping. In *Proceedings of International Conference on Pattern Recognition*, pages 222–224, 1998.

[Compaq Computer Corp., 1995] Compaq Computer Corp. www.altavista.com, 1995.

[Cortes and Vapnik, 1995] C. Cortes and V. Vapnik. Support Vector Networks. *Machine Learning*, 20:273–297, 1995.

[Crowley and Berard, 1997] J. Crowley and F. Berard. Multi-modal tracking of faces for vbideo communications. In *Proceedings of Computer Vision and Pattern Recognition*, pages 640–645, 1997.

[Davis and Bobick, 1997] J.W. Davis and A.F. Bobick. The Representation and Recognition of Human Movement Using Temporal Templates. In *Proceedings of Computer Vision and Pattern Recognition*, pages 928–934. IEEE Computer Society Press, 1997.

[Deriche and Faugeras, 1990] R. Deriche and O. Faugeras. Tracking line segments. *Image and Vision Computing*, 8(4):261–270, 1990.

[Dickmanns and Graefe, 1988] E. Dickmanns and V. Graefe. Applications of Dynamic Monocular Machine Vision. *Machine Vision and Applications*, 1:241–261, 1988.

[Dumais *et al.*, 1998] S. Dumais, J. Platt, D. Heckerman, and M. Sahami. Inductive learning algorithms and representations for text categorization. In *Proceedings of 7th International Conference on Information and Knowledge Management*, November 1998.

[Duta and Jain, 1998] N. Duta and A. Jain. Learning the human face concept in black and white images. In *Proceedings of International Conference on Pattern Recognition*, pages 1365–1367, 1998.

[Ebbecke *et al.*, 1997] M. Ebbecke, M. Ali, and A. Dengel. Real time object detection, tracking and classification in monocular image sequences of road traffic scenes. In *Proceedings of International Conference on Image Processing*, volume 2, pages 402–405, 1997.

[Excite, Inc., 1995] Excite, Inc. www.excite.com, 1995.

[Forsyth and Fleck, 1997] D. Forsyth and M. Fleck. Body plans. In *Proceedings of Computer Vision and Pattern Recognition*, pages 678–683, 1997.

[Forsyth and Fleck, 1998] D. Forsyth and M. Fleck. Finding naked people. *International Journal of Computer Vision*, 1998. accepted for publication.

[Franke and Kutbach, 1996] U. Franke and I. Kutbach. Fast stero based object detection for stop&go traffic. In *Proceedings of Intelligent Vehicles*, pages 339–344, 1996.

[Franke *et al.*, 1997] U. Franke, S. Goerzig, F. Lindner, D. Nehren, and F. Paetzold. Steps towards an intelligent vision system for driver assistance in urban traffic. In *Intelligent Transportation Systems*, 1997.

[Franke *et al.*, 1998] U. Franke, D. Gavrila, S. Goerzig, F. Lindner, F. Paetzold, and C. Woehler. Autonomous driving goes downtown. *IEEE Intelligent Systems*, pages 32–40, November/December 1998.

[Freund and Schapire, 1996] Y. Freund and R. E. Schapire. *Machine Learning: Proceedings of the Thirteenth National Conference*, chapter Experiments with a new boosting algorithm,. Morgan Kaufmann, 1996.

[Garcia *et al.*, 1999] C. Garcia, G. Zikos, and G. Tziritas. Face detection in color images using wavelet packet analysis. In *International Conference on Multimedia Computing and Systems*, pages 703–708, 1999.

[Gavrila and Philomin, 1999] D. Gavrila and V. Philomin. Real-time object detection for "smart" vehicles. In *Proceedings of 7th International Conference on Computer Vision*, volume 1, pages 87–93, 1999.

[Girosi *et al.*, 1995] F. Girosi, M. Jones, and T. Poggio. Regularization theory and neural networks architectures. *Neural Computation*, 7(2):219–269, 1995.

[Guarda *et al.*, 1998] A. Guarda, C. le Gal, and A. Lux. Evolving visual features and detectors. In *Proceedings of International Symposium on Computer Graphics, Image Processing, and Vision*, pages 246–253, 1998.

[Haritaoglu *et al.*, 1998] I. Haritaoglu, D. Harwood, and L. Davis. W4: Who? when? where? what? a real time system for detecting and tracking people. In *Face and Gesture Recognition*, pages 222–227, 1998.

[Harris, 1992] C. Harris. Tracking with Rigid Models. In A. Blake and A. Yuille, editors, *Active Vision*, pages 59–74. MIT Press, 1992.

[Heisele and Wohler, 1998] B. Heisele and C. Wohler. Motion-Based Recognition of Pedestrians. In *Proceedings of International Conference on Pattern Recognition*, 1998. (in press).

[Heisele *et al.*, 1997] B. Heisele, U. Kressel, and W. Ritter. Tracking Non-rigid, Moving Objects Based on Color Cluster Flow. In *Proceedings of Computer Vision and Pattern Recognition*, 1997.

[Hogg, 1983] D. Hogg. Model-based vision: a program to see a walking person. *Image and Vision Computing*, 1(1):5–20, 1983.

[Hoogenboom and Lew, 1996] R. Hoogenboom and M. Lew. Face detection using local maxima. In *Proceedings of International Conference on Automatic Face and Gesture Recognition*, pages 334–339, 1996.

[IBM, 1993] IBM. wwwqbic.almaden.ibm.com, 1993.

[Isard and Blake, 1998] M. Isard and A. Blake. Condensation - conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29(1):5–28, 1998.

[Itti and Koch, 1999] L. Itti and C. Koch. A Comparison of Feature Combination Strategies for Saliency-Based Visual Attention Systems. In *Human Vision and Electronic Imaging*. SPIE, January 1999.

[Itti *et al.*, 1998] L. Itti, C. Koch, and E. Niebur. A Model of Saliency-based Visual Attention for Rapid Scene Analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(11):1254–59, 1998.

[Joachims, 1997] T. Joachims. Text Categorization with Support Vector Machines. Technical Report LS-8 Report 23, University of Dortmund, November 1997.

[Kahn and Swain, 1995] R. Kahn and M. Swain. Understanding people pointing: The perseus system. In *International Symposium on Computer Vision*, pages 569–574, 1995.

[Kalinke *et al.*, 1998] T. Kalinke, C. Tzomakas, and W. van Seelen. A texture-based object detection and an adaptive model-based classification. In *Proceedings of Intelligent Vehicles*, pages 143–147, 1998.

[Kober *et al.*, 1994] R. Kober, J. Schiffers, and K. Schmidt. Model-based versus knowledge-guided representation of non-rigid objects: A case study. In *Proceedings of International Conference onn Image Processing*, pages 973–977, 1994.

[Kotropoulos and Pitas, 1997] C. Kotropoulos and I. Pitas. Rule-based face detection in frontal views. In *Proceedings of International Conference on Acoustics, Speech, and Signal Processing*, 1997.

[Kruger *et al.*, 1997] N. Kruger, M. Potzsch, and C. von der Malsburg. Determination of face position and pose with a learned representation based on labelled graphs. *Image and Vision Computing*, 15:665–673, 1997.

[Kwon and Lobo, 1994] Y. Kwon and N. Lobo. Face detection using templates. In *Proceedings of International Conference on Image Processing*, pages 764–767, 1994.

[Lakany and Hayes, 1997] H. Lakany and G. Hayes. An algorithm for recognising walkers. In J. Bigun, G. Chollet, and G. Borgefors, editors, *Audio- and Video-based Biometric Person Authentication*, pages 112–118. IAPR, Springer, 1997.

[Leung and Yang, 1987a] M.K. Leung and Y-H. Yang. Human body motion segmentation in a complex scene. *Pattern Recognition*, 20(1):55–64, 1987.

[Leung and Yang, 1987b] M.K. Leung and Y-H. Yang. A region based approach for human body analysis. *Pattern Recognition*, 20(3):321–39, 1987.

[Leung *et al.*, 1995] T.K. Leung, M.C. Burl, and P. Perona. Finding faces in cluttered scenes using random labeled graph matching. In *Fifth International Conference on Computer Vision*, June 1995. pages?

[Lew and Huang, 1996] M. Lew and T. Huang. Optimal supports for image matching. In *Digital Signal Processing Workshop Proceedings*, pages 251–254, 1996.

[Lipson *et al.*, 1997] P. Lipson, W. Grimson, and P. Sinha. Configuration Based Scene Classification and Image Indexing. In *Proceedings of Computer Vision and Pattern Recognition*, pages 1007–1013, 1997.

[Lipson, 1996] P. Lipson. *Context and Configuration Based Scene Classification.* PhD thesis, Massachusetts Institute of Technology, 1996.

[Lycos, Inc., 1995] Lycos, Inc. www.lycos.com, 1995.

[MacKay, 1992] D. MacKay. A practical bayesian framework for backpropagation networks. *Neural Computation*, 4(3):448–472, 1992.

[Mallat, 1989] S.G. Mallat. A theory for multiresolution signal decomposition: The wavelet representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(7):674–93, July 1989.

[Maybeck, 1982] P. Maybeck. *Stochastic Models, Estimation, and Control*, volume 141-1 of *Mathematics in Science and Engineering.* Academic Press, Inc., 1982.

[McKenna and Gong, 1997] S. McKenna and S. Gong. Non-intrusive person authentication for access control by visual tracking and face recognition. In J. Bigun, G. Chollet, and G Borgefors, editors, *Audio- and Video-based Biometric Person Authentication*, pages 177–183. IAPR, Springer, 1997.

[Micheli-Tzanakou and Marsic, 1991] E. Micheli-Tzanakou and I. Marsic. Object detection and recognition in a multiresolution representation. In *International Joint Conference on Neural Networks*, pages 2564–2569, 1991.

[Mirhosseini and Yan, 1996] A. Mirhosseini and H. Yan. Neural networks for learning human facial features from labeled graph models. In *Conference on Intelligent Information Systems*, pages 170–173, 1996.

[Moghaddam and Pentland, 1995] B. Moghaddam and A. Pentland. Probabilistic visual learning for object detection. In *Proceedings of 6th International Conference on Computer Vision*, 1995.

[Mohan *et al.*, 1999] A. Mohan, C. Papageorgiou, and T. Poggio. Example based object detection in images by components. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1999. in preparation.

[Mohan, 1999] A. Mohan. Robust Object Detection in Images by Components. Master's thesis, Massachusetts Institute of Technology, May 1999.

[Murase and Nayar, 1995] H. Murase and S. Nayar. Visual Learning and Recognition of 3-D Objects from Appearance. *International Journal of Computer Vision*, 14(1):5–24, 1995.

[Murtagh and Saunders, 1995] B. Murtagh and M. Saunders. *MINOS 5.4 User's Guide*. System Optimization Laboratory,Stanford University, Feb 1995.

[Nelson and Illingworth, 1991] M. Nelson and W. Illingworth. *A Practical Guide to Neural Nets*. Addison-Wesley, 1991.

[Niyogi *et al.*, 1999] P. Niyogi, C. Burges, and P. Ramesh. Capacity control in support vector machines and their application to feature detection. *ieeespeaudproc*, 1999. to be submitted, verify journal name.

[Osuna *et al.*, 1997a] E. Osuna, R. Freund, and F. Girosi. Support Vector Machines: Training and Applications. A.I. Memo 1602, MIT Artificial Intelligence Laboratory, 1997.

[Osuna *et al.*, 1997b] E. Osuna, R. Freund, and F. Girosi. Training support vector machines: An application to face detection. In *Proceedings of Computer Vision and Pattern Recognition*, pages 130–36, 1997.

[Papageorgiou *et al.*, 1998] C. Papageorgiou, T. Evgeniou, and T. Poggio. A trainable pedestrian detection system. In *Proceeding of Intelligent Vehicles*, pages 241–246, October 1998.

[Philips, 1994] P. Philips. Matching pursuit filter design. In *Proceedings of International Conference on Pattern Recognition*, pages 57–61, 1994.

[Platt, 1999] J. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In A. Smola, P. Bartlett, B. Scholkopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*. MIT Press, 1999.

[Qian and Huang, 1997] R. Qian and T. Huang. Object detection using hierarchical MRF and MAP estimation. In *Proceedings of Computer Vision and Pattern Recognition*, pages 186–192, 1997.

[Quinlan, 1996] J.R. Quinlan. Bagging, boosting and c4.5. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, 1996.

[Rajagopalan *et al.*, 1999] A. Rajagopalan, P. Burlina, and R. Chellappa. Higher order statistical learning for vehicle detection in images. In *Proceedings of 7th International Conference on Computer Vision*, pages 1204–1209, 1999.

[Ratan and Grimson, 1997] A. Lakshmi Ratan and W.E.L. Grimson. Training templates for scene classification using a few examples. In *Content Based Access of Image and Video Libraries*, 1997.

[Ratan *et al.*, 1998] A. Lakshmi Ratan, W.E.L. Grimson, and W. Wells III. Object detection and localization by dynamic template warping. In *Proceedings of Computer Vision and Pattern Recognition*, June 1998.

[Reiter and Matas, 1998] M. Reiter and J. Matas. Object-detection with a varying number of eigenspace projections. In *Proceedings of International Conference on Pattern Recognition*, pages 759–761, 1998.

[Rikert *et al.*, 1999] T. Rikert, M. Jones, and P. Viola. A cluster-based statistical model for object detection. In *Proceedings of 7th International Conference on Computer Vision*, pages 1046–1053, 1999.

[Rohr, 1993] K. Rohr. Incremental recognition of pedestrians from image sequences. *Proceedings of Computer Vision and Pattern Recognition*, pages 8–13, 1993.

[Rowley *et al.*, 1995] H.A. Rowley, S. Baluja, and T. Kanade. Human face detection in visual scenes. Technical Report CMU-CS-95-158, School of Computer Science, Carnegie Mellon University, July/November 1995.

[Rowley *et al.*, 1997] H. Rowley, S. Baluja, and T. Kanade. Rotation invariant neural network-based face detection. Technical Report CMU-CS-97-201, School of Computer Science, Carnegie Mellon University, December 1997.

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

[Saber and Tekalp, 1996] E. Saber and A. Tekalp. Face detection and facial feature extraction using color, shape and symmetry-based cost functions. In *Proceedings of International Conference on Pattern Recognition*, pages 654–658, 1996.

[Schapire *et al.*, 1998] R.E. Schapire, Y. Freund, P. Bartlett, and W.S. Lee. Boosting the margin: a new explanation for the effectiveness of voting methods. The Annals of Statistics, 1998. To appear.

[Schneiderman and Kanade, 1998] H. Schneiderman and T. Kanade. Probabilistic modeling of local appearance and spatial relationships for object recognition. In *Proceedings of Computer Vision and Pattern Recognition*, pages 45–51, 1998.

[Shams and Spoelstra, 1996] Ladan Shams and Jacob Spoelstra. Learning gabor-based features for face detection. In *Proceedings of World Congress in Neural Networks. International Neural Network Society.*, pages 15–20, Sep 1996.

[Sinha, 1994a] P. Sinha. Object Recognition via Image Invariants: A Case Study. In *Investigative Ophthalmology and Visual Science*, volume 35, pages 1735–1740. Sarasota, Florida, May 1994.

[Sinha, 1994b] P. Sinha. Qualitative image-based representations for object recognition. A.I. Memo 1505, MIT Artificial Intelligence Laboratory, 1994.

[Stollnitz *et al.*, 1994] E.J. Stollnitz, T.D. DeRose, and D.H. Salesin. Wavelets for computer graphics: A primer. Technical Report 94-09-11, Department of Computer Science and Engineering, University of Washington, September 1994.

[Sullivan *et al.*, 1995] M. Sullivan, C. Richards, C. Smith, O. Masoud, and N. Papanikolopoulos. Pedestrian tracking from a stationary camera using active deformable models. In *Proceedings of Intelligent Vehicles*, pages 90–95, 1995.

[Sung and Poggio, 1994] K-K. Sung and T. Poggio. Example-based learning for view-based human face detection. A.I. Memo 1521, MIT Artificial Intelligence Laboratory, December 1994.

[Sung and Poggio, 1998] K-K. Sung and T. Poggio. Example-based learning for view-based human face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(1):39–51, January 1998.

[Sung, 1995] K-K. Sung. *Learning and Example Selection for Object and Pattern Detection.* PhD thesis, MIT Artificial Intelligence Laboratory, December 1995.

[Takatoo *et al.*, 1996] M. Takatoo, C. Onuma, and Y. Kobayashi. Detection of objects including persons using image processing. In *Proceedings of International Conference on Pattern Recognition*, pages 466–472, 1996.

[Tsukiyama and Shirai, 1985] T. Tsukiyama and Y. Shirai. Detection of the movements of persons from a sparse sequence of tv images. *Pattern Recognition*, 18(3/4):207–13, 1985.

[Vaillant *et al.*, 1994] R. Vaillant, C. Monrocq, and Y. Le Cun. Original approach for the localisation of objects in images. *IEE Proceedings Vision Image Signal Processing*, 141(4):245–50, August 1994.

[Vapnik, 1995] V. Vapnik. *The Nature of Statistical Learning Theory.* Springer Verlag, 1995.

[Vapnik, 1998] V. Vapnik. *Statistical Learning Theory.* John Wiley and Sons, New York, 1998.

[Venkatraman and Govindaraju, 1995] M. Venkatraman and V. Govindaraju. Zero crossings of a non-orthogonal wavelet transform for object location. In *Proceedings of International Conference on Image Processing*, pages 57–60, 1995.

[Virage, Inc., 1994] Virage, Inc. www.virage.com, 1994.

[Wahba, 1990] G. Wahba. *Spline Models for Observational Data.* Series in Applied Mathematics, Vol. 59, SIAM, Philadelphia, 1990.

[Weber and Casasent, 1997] D. Weber and D. Casasent. Quadratic gabor filters for object detection. In *Proceedings of Symposium on Communications and Signal Processing*, pages 69–74, 1997.

[Wolpert, 1995] D. Wolpert, editor. *The Mathematics of Generalization: The Proceedings of the SFI/CNLS Workshop on Formal Approaches to Supervised Learning*, Santa Fe Institute Studies in the Sciences of Complexity. Addison-Wesley, 1995.

[Wren *et al.*, 1995] C. Wren, A. Azarbayejani, T. Darrell, and A. Pentland. Pfinder: Real-time tracking of the human body. Technical Report 353, MIT Media Laboratory, 1995.

[Wu *et al.*, 1999] H. Wu, Q. Chen, and M. Yachida. Face detection from color images using a fuzzy pattern matching method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(6):557–563, 1999.

[Yahoo!, Inc., 1994] Yahoo!, Inc. www.yahoo.com, 1994.

[Yang and Huang, 1993] G. Yang and T. Huang. Human face detection in a scene. In *Proceedings of Computer Vision and Pattern Recognition*, pages 453–458, 1993.

[Yang and Huang, 1994] G. Yang and T. Huang. Human face detection in a complex background. *Pattern Recognition*, 27(1):53–63, 1994.

[Yow and Cipolla, 1996a] K. Yow and R. Cipolla. A probabilistic framework for perceptual grouping of features for human face detection. In *Proceedings of International Conference on Automatic Face and Gesture Recognition*, pages 16–21, 1996.

[Yow and Cipolla, 1996b] Kin Choong Yow and R. Cipolla. Detection of human faces under scale, orientation and viewpoint variations. In *Proceedings of the Second International Conference on Automatic Face and Gesture Recognition.*, pages 295–300, Oct 1996.

[Yow and Cipolla, 1997] Kin Choong Yow and R. Cipolla. Feature-based human face detection. *Image and Vision Computing*, 15(9):713–35, Sep 1997.

[Yuille *et al.*, 1992] A. Yuille, P. Hallinan, and D. Cohen. Feature Extraction from Faces using Deformable Templates. *International Journal of Computer Vision*, 8(2):99–111, 1992.

[Yuille, 1991] A. Yuille. Deformable Templates for Face Recognition. *Journal of Cognitive Neuroscience*, 3(1):59–70, 1991.