THE UNIVERSITY *of* EDINBURGH

Situated Face Detection

Arturo Espinosa-Romero

Ph.D.
University of Edinburgh
2001

Abstract

In the last twenty years, important advances have been made in the field of automatic face processing, given the importance of human faces for personal identification, emotional expression and verbal and non verbal communication.

The very first step in a face processing algorithm is the detection of faces; while this is a trivial problem in controlled environments, the detection of faces in real environments is still a challenging task. Until now, the most successful approaches for face detection represent the face as a grey-level pattern, and the problem itself is considered as the classification between "face" and "non-face" patterns. Satisfactory results have been achieved in this area. The main disadvantage is that an exhaustive search has to be done on each image in order to locate the faces. This search normally involves testing every single position on the image at different scales, and although this does not represent an important drawback in off-line face processing systems, in those cases where a real-time response is needed it is still a problem.

In the different proposed methods for face detection, the "observer" is a disembodied entity, which holds no relationship with the observed scene. This thesis presents a framework for an efficient location of faces in real scenes, in which, by considering both the observer to be situated in the world, and the relationships that hold between the two, a set of constraints in the search space can be defined. The constraints rely on two main assumptions: first, the observer can purposively interact with the world (*i.e.* change its position relative to the observed scene) and second, the camera is fully calibrated.

The first source of constraint is the structural information about the observer environment, represented as a depth map of the scene in front of the camera. From this representation the search space can be constrained in terms of the range of scales where a face might be found at different positions in the image. The second source of constraint is the geometrical relationship between the camera and the scene, which allows us to project a model of the subject into the scene in order to eliminate those areas where faces are unlikely to be found.

In order to test the proposed framework, a system based on the premises stated above was constructed. It is based on three different modules: a face/non-face classifier, a depth estimation module and a search module. The classifier is composed of a set of convolutional neural networks (CNN) that were trained to differentiate between face and non-face patterns, the depth estimation modules uses a multilevel algorithm to compute the scene depth map from a sequence of images captured while the camera is moved in a controlled manner, and the search module combines the depth information and the subject model into the image where the search will be performed in order to constrain the search space.

Finally, the proposed system was validated by running a set of experiments on the individual modules and then on the whole system.

Acknowledgements

There are many people with whom I am grateful, without whom the work presented in this thesis would have not been possible, and would have been a much more difficult enterprise.

First of all, I need to thank my family, for their support and encouragement.

I also need to thank my colleagues at the mobile robots group, for their useful comments and advice. Particularly Richard Reeve, Louise Matthews and Heba Lakany for their comments, advice, and help proof-reading this document. Simon Perkins who taught me how to use "Gillespie" the robot, and his friendship, and John Demiris who make me feel comfortable in the group and the department. I also would like to thank my supervisor, John Hallam for his support, guidance and time, Bob Fisher and Gillian Hayes who gave me useful comments on my work, whose advice I really appreciate. I would also like to thank Bridget Hallam for her support and very opportune advice in time of need.

It also has to be mentioned the encouragement, guidance and friendship I received from Jesus Figureoa, Esther Vargas, Walterio Mayol, and Alberto Muñoz, without whom the last five years of study in Edinburgh would not have been possible.

In the crucial last months of my studies, I received invaluable support from Louise Matthews, Richard Reeve, and Justin Roberts, who helped me in times of illness and despair; for this and many others things, I am particularly grateful. I also need to thank Charlotte Cabrero, who gave me advice and love, and made my life happier and more enjoyable in my last years in Britain.

I am as well indebted to many people I met in Edinburgh, who with their friendship, my stay in Britain would not have been as pleasurable as it was. In particular I would like to thank Monica Sanchez, Rocio Aguilar, Ingrid Van Lancker, Azhar Ahmad and Rikke Magnussen for the long discussions, lovely meals, and support in difficult times. I also I need to mention Paul Sandle, Mike Spratling, Alberto and Mayte Muñoz, Spratling, Marloes van Amerom, Lars Wilhelmsson, Anthony Ashbrook, Jose Carmena, Esther Dura and Joachim Huelsmann for their hospitality, friendship and help to adapt to different customs and different ways of seeing life; and especially to Francesca Serra, Ana Agusti, Emma Barriero, Terence Chan, William Chester, Enrique Corona, Heba Lakany, Tim Taylor, Abraham Escobar, Terence Chan, William Chester, Raul Monroy, Rafael y Paty Morales, Maria Navarrete, Herbert Peremans, John and Louise Taylor, Francisco and Myrna Villareal, and Rafael and Claudia Villegas for their friendship and for being at the right moment at the right time.

This research has been done with the support of the scholarship 71357 from CONACyT (Consejo Nacional de Ciencia y Tecnología, México.).

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Human faces are complex multidimensional visual patterns which convey important information, not only for identification purposes, but also for emotional expression and verbal and non-verbal communication (Ellis and Young, 1989). The extraction and interpretation of this information requires elaborate and refined perceptual skills which represent the peak of our recognition capacity. This ability plays a very important role in our social intercourse as the identification of individuals relies heavily on face recognition. Furthermore, this skill is quite robust and we can recognise faces with different expressions, orientation and variations in the appearance of the subject (*e.g.* lenses, beard, hair style and age).

In the last 20 years the study of the processes underlying the perception and recognition of faces has attracted the attention of researchers from a variety of disciplines. In particular, the development of computational models of face recognition ("automatic face recognition") has received special attention, mainly because of their important practical applications. Some examples of those applications are: forensic science, image retrieval from a database, surveillance, person identification and human-machine interaction.

In this thesis we deal with the problem of detection and location of faces in real images, which is the first stage of any automatic face processing system. In this introductory chapter some preliminary issues concerning the work presented in this thesis are given, together with the main objective and scope. The chapter is organised as follows: first, a brief review of the psychological basis of face processing is given,

followed by a small introduction to automatic face processing systems. Next, the objective of the thesis is presented, with a brief description of achievements. Finally a description of the chapters to follow is given.

1.1 Face Processing

It is important to distinguish between four main face processing tasks that may be confused. These tasks are recognition, identification, classification and detection of faces: face recognition implies the classification of a face as familiar or unfamiliar, face identification establishes a relationship between a face and semantic information (*e.g.* name), face classification differentiates faces according to common physical characteristics (sex, race, age) and finally, face detection is the classification of a stimulus as face or not face, and implies the detection of the face features and their spatial relationship (configuration). The information needed to accomplish these tasks is not the same for each case; for example we can recognise a face as a familiar one, but if we can not recall her/his name we say that the face is recognised but not identified. Face identification relies on visual and semantic information, while detection, recognition and classification rely only on the visual information.

Bruce and Young (1986) proposed a model for face processing which decomposes the face processing system into a set of functional components. In the model, shown in figure 1.1, each box represents an independent functional unit. The arrows between boxes denote access to information and activation of one component by another. Each component plays a distinct functional role in the model, whose operation can be isolated or manipulated by means of experimentation or as a consequence of brain damage.

The first stage in this model[1] is structural encoding, which is particularly important for us because, as Ellis (1986) pointed out, this is the process responsible for the detection of faces. Bruce and Young gave a more significant role to this component, making it responsible for the extraction of a set of descriptors — view-centred descriptors used by the components responsible for expression and facial speech analysis and expression independent descriptors used by the face recognition units.

[1] Which is also the first stage in model proposed by Ellis (1986).

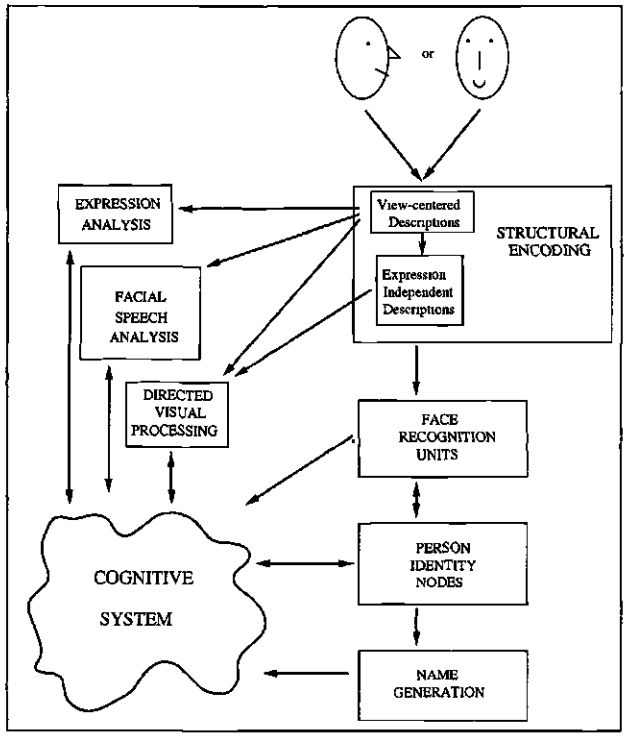


Figure 1.1: Bruce&Young face processing model.

The issue of whether a face needs to be first classified as a face in order to be processed as such has not been resolved. Ellis (1981) has argued that given the importance of facial configurations as biological stimuli for primates it is likely that at least the early stages of face processing (*i.e.* face detection) are hard-wired to some degree. He also mentions that the strong bias humans have for seeing faces in ambiguous configurations, such as clouds and flames, (Ellis, 1986) is an indication of a special face detection stage, *i.e.* while we can sometimes fail to correctly classify an object, failures to classify a face as being one are unlikely.

On the other hand, Bruce and Young (1986) argue that a prior face detection stage would imply a special type of analysis for face processing which, although being very probable given our outstanding ability to process facial information, is not yet known. Also, even if such specialisation exists, an early classification of a stimulus as a face may not be necessary, since such classification happens simultaneously with a more specific classification, *i.e.* face detection relying on global structural description may occur, while face recognition/identification occurs based on local information.

For automatic face processing systems the face detection stage is a necessity, since it enormously reduces the complexity of any face processing task by bounding the area and scale where the face features are located in the image.

Bruce and Young (1986) proposed that faces are represented via an interlinked set of expression-independent structural codes for distinct head angles, with some codes reflecting the global configuration at each angle and others representing particular distinctive features. They argue that the range of viewpoints across which face recognition is performed is considerably smaller than the range of viewpoints involved in object recognition. Normally the position of the face is more or less upright and there are geometric transformations that only apply for face recognition such as facial expression. Their thinking has been influenced by the work of Perrett et al. (1985), who demonstrate that some visual cells in the temporal cortex of macaque monkeys are sensitive to specific views of the face. This suggests that recognition may proceed via the independent analysis of several restricted view-centred descriptions of the object.

1.2 Automatic face detection

The different approaches proposed to solve the face detection problem have focused mainly on the development of face classifiers that are able to discriminate between faces and the image background. The problem that has not been completely solved, and has been somehow neglected, is the location of faces in the image. A face can be found at different positions in the image and at different scales; the localisation involves an exhaustive search, and as will be seen later in chapter 2, this problem is not trivial and is computationally expensive.

The different applications for face processing can be roughly classified in terms of the response time needed: those applications that need a real-time response, and those where that is not necessary (on-line and off-line face processing systems respectively).

For off-line face processing systems, processing time is not an issue, and they also have the advantage that there are fewer restrictions in terms of computer resources available (*e.g.* computing power and memory). Examples of such systems are database image retrieval systems, where a face image has to be found in a image database,

or photographic image processing systems, where it would be useful to identify the location of the faces in the image in order, for example to use selective processing for different parts of the photograph.

On-line face processing systems involve interaction with people, where the system reaction time should be transparent to the person using it, or as fast as the person being observed by it. As examples of on-line face processing systems we can mention human-machine interaction systems, where analysis of facial-expression, person identification and tracking are common issues, and automatic surveillance systems where persons have to be identified while still in the camera field of view. In contrast with off-line face processing systems, the computing resources normally available in this kind of system are more restricted, *e.g.* for a human-machine interactive system to be really useful it is necessary that it be affordable, thus requiring cheap components and limited resources.

1.3 Situated face detection

The objective of this thesis is the development of a framework for efficient detection of faces in real scenes. Such a framework should allow us to implement a system able to detect human faces with an acceptable range of orientations in real time.

The main problem is the complexity of the search, where every position of the image has to be tested at different scales. In order to simplify this problem, I decided to exploit the relationship between the observer, scene and subject, and the fact that the observer is situated in the world to constrain the search space.

Two main constraints can be derived from the fact of having the observer situated in the world. The first source of constraint is the structural information of the robot's environment, represented as a depth map of the scene in front of it. Using the depth map, the search space can be constrained in terms of the range of scales where a face might be found at different positions in the image.

The second source of constraint emerges from the geometrical relationship between the observer, the scene and the subject. This relationship allows us to determine those

areas of the image where it is unlikely that a face might be found. The relationship between the observer and the scene is expressed in terms of the camera external calibration parameters (position and orientation), while the relationship between the subject and the scene is expressed in terms of an anthropometric model.

The framework defines three different modules: a face/non-face classification module, a depth estimation module and a search module.

The face classifier is composed of a set of convolutional neural networks (CNN) which were trained to differentiate between face and non-face patterns. In order to increase the range of facial orientations that could be detected, two classifiers were trained; the first was trained using a training set composed of front view faces, while the training set used to train the second classifier is composed of side view faces.

The depth estimation module computes the distance between the observer and the structures present in the scene by means of kinetic depth; a sequence of images is captured while the camera is moved in a controlled fashion following a predetermined trajectory. A novel hierarchical multilevel algorithm was used to mix disparity measures obtained by comparing different sequence frames, and then obtain a rough but fast depth estimation, increasing the quality of the depth map as time goes by.

The search module first combines the depth information and the geometric model in order to constrain the search space, and then performs a ranked search, where the places more likely to contain a face are tested first; the ranking of the search space elements is defined in terms of the same geometrical model used to define the constraints.

The main difference between the on-line face detection system proposed here and most of the approaches for face detection published to date (reviewed in chapter 2), is the fact that while in those approaches the observer is a disembodied entity which has no relationship with the scene depicted by the image being analysed, the approach proposed here acknowledges the relationship between the observer, the scene and the subject, as well as the fact that the observer inhabits the same space as the subject and reacts in a similar time frame.

The relationship between the observer and the scene can be better expressed in terms of "situatedness" and "embodiment", as defined in the artificial intelligence paradigm

proposed by Brooks (1991), if we define the face detection system as an agent. An agent can be considered to be a system that senses the environment, and acts upon it in pursuit of its own goals.

An agent is said to be situated when it interacts directly with the world, without using a formal description of the environment, while embodiment implies that the agent has a body and interacts directly with the world, by sensing it and acting on it.

As pointed out by Hallam and Malcolm (1994), situatedness implies that the agent must be able to recover from its environment, through its sensors, whatever is needed to determine the appropriate course of action, and must be able to participate in its environment. This in turn implies that the agent dynamics operate at a speed similar to the external world. Also embodiment allows the agent to exploit its interaction with the world in order to simplify perceptual and motor problems.

A face detection system based on the framework described in this thesis can be considered as an embodied agent, since the observer (camera) is physically present in the world, and interacts with the world in a way that allows it to simplify the execution of the main task. Also such a face detection system can be said to be situated since it is able to react in real-time to events in the scene, and the course of action (detecting a face) is determined from information obtained from its sensors.

1.4 Thesis organisation

This thesis is organised as follows:

- In the second chapter a review of the face detection techniques published to date is presented, dividing the different approaches by the kind of facial representation used, which in turn determines the complexity of face detection procedure.

- In the third chapter the details of the implementation of two different face classifiers is given: the first classifier is composed of one or more convolutional neural networks trained to discriminate between face and non-face patterns, while the second classifier models the face–non-face pattern distribution in terms of a set of Gaussian clusters. For each kind of classifier, two instances were produced,

each using a different, independent training set, one composed of front view face patterns and another composed of side view face patterns. The classifiers were trained using standard test images, and the results obtained are compared with results of similar classifiers known from the literature.

- Chapter four proposes an approach for kinetic depth estimation. The method estimates depth from the disparity measure obtained from a sequence of images captured with a moving camera, mounted on top of a mobile robot which follows a predetermined path. A novel method is used to minimise the computational effort needed, and the adverse effects of object occlusion, by mixing different disparity measurements, providing a fast response and an increase of the quality of the measurements as time goes by. The approach is tested with artificial and real images.

- In chapter five, I explain how the search space is constrained, using depth information and the geometrical relationships between the observer, scene and subject. I also propose a search algorithm which takes advantage of these geometric relationships to rank the elements of the search space, in order to obtain fast and on-the-fly detection. The system is tested on a set of images, and the results are given.

- Finally, in chapter six, a discussion and conclusion of the work presented in this thesis is given.

Chapter 2

Face Detection:A Review

2.1 Introduction

Face detection is the classification of a stimulus as being a face or not. While it is a fairly straight forward procedure under controlled conditions, it becomes a difficult and interesting problem under normal conditions, where the background is cluttered and there are uncontrolled lighting variations.

The face detection problem, which has received particular attention from the vision community due to the importance of the human face as a visual stimulus, can be considered as a particular case of the more general problem of object detection. Human faces are complex 3-dimensional objects, with a well-defined structure, although there is some variability in face configuration between class members, and variations due to changes in facial expression. Another characteristic of faces as an object is that they are naturally found within a limited range of orientations: human faces normally have a more or less upright position, and have a limited range for panning and tilting movements. Human faces are a favourite object to test detection and recognition algorithms.

Although faces are 3-dimensional objects, all the approaches considered in this review use a two-dimensional representation of the face. Normally, most of the methods are designed to detect faces from a constrained range of views, normally the up-right frontal view. Rotations around the axis perpendicular to the image plane do not represent a major problem since the face image can always be restored to an orientation suitable

for detection. The main problem occurs when there is a rotation around the other two axes, since the two-dimensional representation of the face can not then be rotated to comply with the view range accepted by the detector.

2.2 Face Detection Review

There are several different approaches to solving the task of face detection that have been suggested in the last ten years. The main feature that distinguishes those methods is the kind of representation for the face structure that is used; most of these methods represent the face either as a two-dimensional grey level pattern or as a two-dimensional composite of features (eyes, nose, eye-brows, mouth, sideline of the face), which has a particular geometrical configuration.

2.2.1 Faces as grey level patterns

This approach involves the training of a classifier to distinguish between two different classes, the face and non-face classes. The face patterns usually represent the upright front view of faces, with a small variation in size and rotation of the image around the camera axis. The training set of face patterns can be easily constructed; at the moment there are several face databases available on the Internet which can be used to define a comprehensive set of face patterns[1]. The interesting problem is to select a representative set of non-face examples. Non-face patterns are, in principle, all of those patterns that are not faces, which is a huge set. In order to keep the number of non-face patterns manageable, it is important to select those non-face patterns that are likely to be misclassified. This is usually achieved using a bootstrap method, which extracts non-face patterns from images without faces in them (*e.g.* landscapes, abstract images) by means of selecting those which are classified as being faces by the classifier trained with an initial set of faces and random patterns as non-face examples. The new non-face patterns extracted are then added to the training set. This technique was first proposed by Sung and Poggio (1994), and will be explained in detail in chapter 3.

[1] The full list of the face databases available on the internet at the time of writing is listed in appendix B.

Normally the patterns are represented as low resolution images, which is almost equivalent to using low frequency images (more precisely low frequency images which have high frequency noise on the boundaries of the pixels). It is known that low spatial frequencies are particularly important for face perception; they carry most of the energy, are resistant to degradation and contain most of the relevant information needed for face processing (Sergent, 1989)[2]. It is also necessary to use small patterns to keep the classification problem computationally tractable. Normally the range of sizes of the pattern used to represent the faces varies from 11×11 to 20×20 (Colmenarez and Huang, 1997; Rowley et al., 1998, respectively).

Most of the approaches use an 8 bit quantisation level, giving 256 grey levels, though Colmenarez and Huang (1997) re-quantised the images to 4 levels, choosing the threshold values via image equalisation.

The face pattern is normally represented in a square grid, but most approaches mask the vertical borders and corners of the pattern. Those pixels usually represent the background of the image. This masking procedure can be considered equivalent to defining a rectangular pattern, instead of a square one, that fits better the shape of human faces (see figure 2.1).

In a normal environment it is not possible to control the lighting conditions of the image from which the pattern to be analysed is extracted. Variations of the illumination and the distribution of the pattern histogram will make the classification more difficult. Though some variation is normally left for the classifier to learn, a preprocessing stage which minimises those variations is commonly used (see figure 2.1).

A prototypical example of this preprocessing stage is the one proposed by Sung and Poggio (1994). It involves masking the borders and corners of the pattern, then applying an illumination correction procedure, by subtracting a best-fit brightness plane from the pattern which helps to minimise the effect of strong directed lighting, and finally using a histogram equalisation procedure in order to improve contrast and to correct the effect of the response curves of different cameras.

[2] It is important to note that only feature-based approaches to face detection use high resolution images, since they need to detect internal features (such as eyes, nose and mouth), which would not be well characterised in low resolution images.

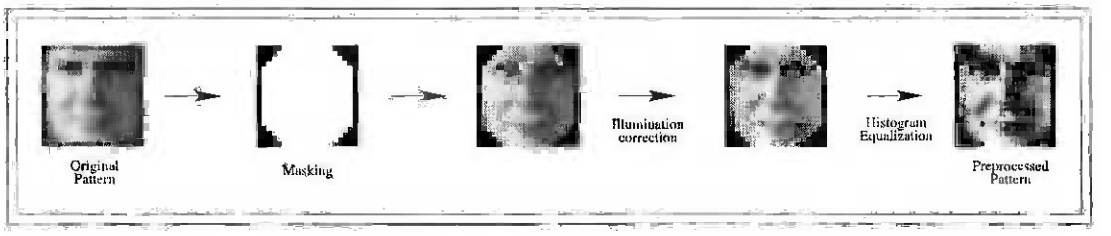


Figure 2.1: Typical preprocessing stage, after Sung and Poggio (1994)

Types of classifiers

There are two typical approaches to classifying the patterns. The first is explicitly to model the distribution sampled by the training set and the second is to use neural networks or other implicit methods to perform the classification.

A delicate issue related to this kind of approach is how to perform the search for faces in an image. The face is represented as a fixed size pattern, which could be located in almost any position of the image at different scales. The standard method is to test each position on an image pyramid[3], extracting a pattern from each position and using the classifier to test it. All those methods described below adopt that procedure.

Explicit probabilistic methods

Turk and Pentland (1991) proposed one of the first pattern classification methods for face detection. They chose to represent a face image in terms of the principal components of a distribution of face images. This proposed method — also known as eigenfaces — is based on the work of Kirby and Sirovich (1990). The method uses principal component analysis (PCA) to obtain a set of vectors that best account for the distribution of face images on the image space, *i.e.* the eigenvectors with the highest related eigenvalues of the covariance matrix of the face image distribution, which maximise the variance of the distribution. The term "eigenfaces" comes from the fact that when the eigenvectors are reconstructed as images they resemble a face (see figure 2.2).

Turk and Pentland chose the M most significant eigenvectors to define a subspace,

[3] An image pyramid is composed of a set of scaled copies of the original image; each image is stored on a level of the pyramid and is a reduced version of the image stored on the previous level (Tanimoto and Pavlidis, 1975).

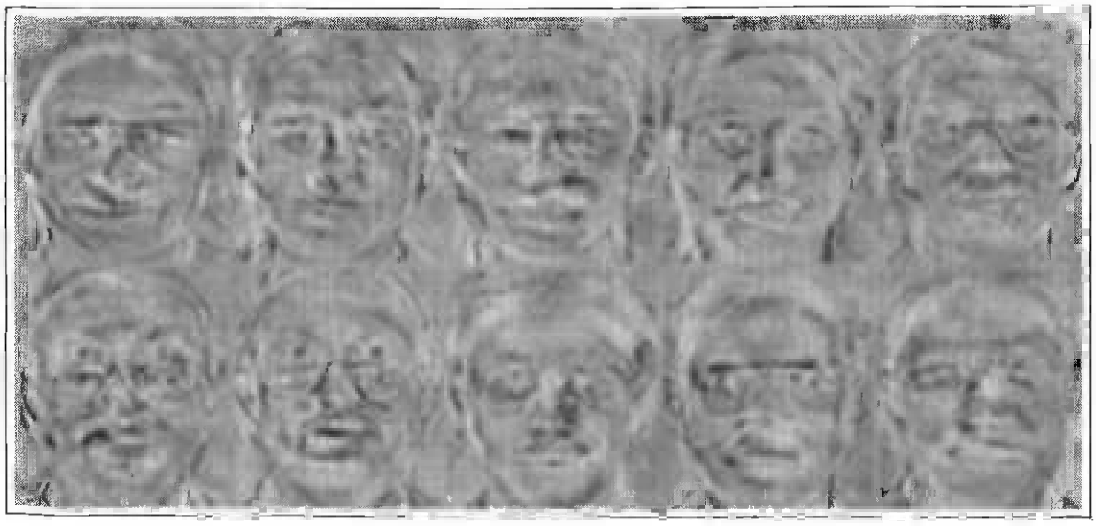


Figure 2.2: Example of eigenfaces: the ten most significant eigenfaces obtained from a set of face images.

the "face space". A face image is thus projected into this subspace and is characterised by the set of M weights obtained from the projection, achieving by this a compact representation of the face image.

Their work is originally focused on face recognition, and the classification is based on two distances, the subspace Euclidean distance between a face image and a particular class within the subspace, and the "distance from face space" or reconstruction error. While the first distance is useful to classify a face as a known face or not, the second one is necessary to make sure that the projected image is actually a face image. They note in their work that this second distance could be useful for the face detection task, by calculating it for every location on the image to be analysed; the result of this is a "face map" where local minima suggest the presence of a face.

The main disadvantage of their implementation is that it is very computationally expensive: they use relatively high resolution grey level images (128×128 pixels), given that the objective of their work is face recognition, which requires more information to achieve good results. In order to improve the performance of their face detector they express it in terms of a set of convolution operations, which are efficiently computed using a fast Fourier transform. The main contribution of their work is the definition of the "distance from face space" (DFFS[4]) and the "distance in the face space" (DIFS)

[4] Generalised in further work by other authors as "distance from feature space".

as useful metrics for the classification process.

The seminal work of Sung and Poggio (1994) defined most of the common features shared by subsequent pattern classification approaches for face detection. Their main contributions, apart from the classification method, are the definition of an adequate preprocessing stage, and the use of a bootstrapping algorithm to generate non-face patterns.

Their approach uses 19×19 grey level patterns of face and non-face images. The face patterns (4,150 of them) are obtained by resizing hand-cropped faces from real images. More than one face pattern is obtained from each face image, by using slightly rotated and mirrored versions of the original face. The non-face patterns (43,166 of them) were obtained iteratively using the bootstrap method.

The distribution of face and non-face patterns is modelled by a set of 12 clusters; 6 for face and 6 for non-face patterns. Those clusters are obtained by means of independently applying a modified K-means algorithm to the face and non-face databases. The main characteristic of this version of the K-means algorithm is that it uses the normalised Mahalanobis distance; this allows the formation of elliptical clusters, which fit the distribution of the data better than the spherical clusters obtained using an Euclidean metric.

The patterns to be classified are matched against the face distribution model. Twelve distances are calculated, between the pattern and each of the clusters, similar to the ones used by Turk and Pentland (1991). Each distance is composed of two components: the first component is the normalised Mahalanobis distance between the pattern and the cluster centroid projections in the subspace spanned by the 75 largest eigenvectors of the cluster covariance matrix, and the second component is the Euclidean distance between the test pattern and its projection in the subspace.

The distances obtained by this method are fed into a multi-layer perceptron, with 24 input units, 24 hidden units and one output unit, all of them using a sigmoidal activation function. The output unit returns a 1 if the set of distances fed into the network corresponds to a face pattern or 0 if that is not the case.

In order to evaluate the performance of their system, they use two different sets of

images. The first one (test set A) is a collection of 301 high quality frontal and near frontal face mugshots of 71 different people, which were taken with a high quality CCD camera. The second set (test set B) is composed of 23 images which contain 149 faces of different scales and varying quality. They report a detection rate of 96.7% with 3 false detections on the first set, and a detection rate of 84.6% with 13 false detections on the second set. A detailed description of their approach will be given in chapter 3.

Colmenarez and Huang (1997) propose a method to perform the classification using a probability model which optimises the divergence between the face and non-face classes. An observation is classified as being a face or not by first estimating the likelihood ratio using the probability model and then comparing it with a fixed threshold.

The probability model is defined as a first order Markov process, which is found by means of a learning process which maximises the divergence between the two classes represented in a training set. The measure of divergence is given by the Kullback-Leibler relative information distance (Gray, 1990), which gives a non-negative measure of the difference between two different probability distributions and is equal to zero when they are identical.

The pattern is represented using a grid of 11×11 pixels. The only preprocessing consists of an image quantisation to four levels using threshold values obtained from histogram equalisation. There is no illumination correction step; the variation in illumination in the image is expected to be learned by the classification algorithm.

The probability model is constructed in the following way: for each pair of pixels of the set of images in the training set, the probability functions for the face and non-face classes are estimated using a joint histogram, and from there the Kullback-Leibler divergence measure is found for each pixel pair. Given a table with the divergence measures, an optimisation technique is used to find the Markov process that maximises the divergence between the two classes. The optimisation problem is simplified and treated as the construction of a minimum weight spanning tree. The probability model obtained is used to construct a set of tables that are used to compute the likelihood ratio of an observation. A pattern is classified as being a face if the likelihood ratio computed for it is above a certain threshold.

They report results of four instances of their classifier, which were tested against the Carnegie Mellon University (CMU) set of test images[5]. The detection rates vary from 98.0% to 86.6% with the number of false detections being roughly proportional to the detection rate (*e.g.* the better the detection rate, the higher the number of false detections), varying from 6,133 to 12,758 false detections, while testing 26,337,890 windows.

It is important to note that of the different approaches reviewed here, this is the system which uses the smallest face representation. Another remarkable feature of this method is that once the likelihood tables have been computed, the classification process involves only a small number of operations (*e.g.* in their implementation they estimate around 100 fixed-point additions), which is directly reflected in the speed of the search process.

Implicit probabilistic methods

Vaillant et al. (1994) proposed a neural network based approach for face detection. They use a variation of the convolutional neural network proposed by LeCun et al. (1989), which has been used for handwritten character recognition.

The network is composed of three hidden layers and one output layer. The input or distribution layer has 20×20 units, the first hidden layer has four sets of 16×16 units or feature maps, where each unit is connected to a 5×5 region of the input layer in such way that the whole input area is covered. All the units in each feature map share weights. The second hidden layer is composed in a similar fashion to the previous one with the main difference that the size of each map is only 8×8 and the region covered by each unit is only 2×2. The third hidden layer is composed of four units, each of them being fed by the outputs of each of the feature maps on the previous layer. The output layer is a single unit which is connected to the four units on the third hidden layer.

The training set is composed of what they call "face patches": 32×48 grey level images which contain a 20×20 face image in the centre. The database contains 1792 face patches, and an equal number of non-face or background patches.

[5] A detailed description of this test set is given in section 2.3.

They train the network to perform three slightly different tasks: a complete localisation, a rough localisation and a precise localisation. The complete localisation task trains the network to classify between perfectly centred face patches and background patches. For the rough localisation task, the network is fed with centred and shifted face patches as positive examples and background patches as counter examples; in the case of the positive examples the desired output is α for a perfectly centred face, and $\alpha(2e^{-\lambda\sqrt{x^2+y^2}} - 1)$ otherwise; x and y is the difference between the centre of the face and the centre of the patch, and λ determines the decay of the desired output as the face moves away from the centre; for the counter examples the desired output is $-\alpha$. With this it is expected that the network will give a positive response inversely proportional to the distance of the face from the centre of the patch (the maximum response being in those cases where the face is perfectly centred), and a negative response if a background pattern is presented. For perfect localisation the network is trained to produce an output of α when a perfectly centred face is presented and $-\alpha$ when a shifted face is presented; the background patches are not used in this case.

The detection process use the networks trained to perform a rough localisation on the image, using a Gaussian pyramid in order to achieve scale independence. The detections obtained by this classifier are used as hypotheses, which are refined by applying the perfect localisation network. Finally a "grouping" algorithm is used to eliminate multiple detections. They report positive results, although not in a quantitative form. Most of the characteristics of their work were adopted and improved in the work of Rowley et al. (1998) which will be reviewed next.

Rowley et al. (1998) devise a two stage system for face detection: the first stage is composed of a set of one or more neural network based classifiers which are used to scan the image, in the second stage multiple detections are eliminated, and given the case of using more than one neural network, an arbitration procedure is used to combine their outputs into a single one.

Each classifier is based on a multi-layer perceptron with three layers. The first, or distribution layer, is arranged as a grid of 20×20 pixels. The connections between the units in the hidden layer and the distribution layer are organised so that each unit in the hidden layer is only connected to a particular local area within the distribution

layer (receptive field). These different receptive fields have different shapes: horizontal rectangles, small and big squares. The reason for using this particular configuration is that the receptive fields are expected to fit the geometrical features present on human faces (eyes, mouth, nose). The receptive fields are arranged in sets which cover the whole area of the input layer, and for different experiments each network uses two or three of these sets. The output unit is fed from the output of all the hidden layer units. All the units use a hyperbolic tangent activation function and the network is trained to give an answer of $+1$ if a pattern is classified as being a face and -1 otherwise. The network was trained with the back-propagation algorithm (Rumelhart et al., 1986) using a training set composed of 15,750 face patterns and approximately 9,000 non-face patterns, obtained using a variation of the bootstrap method proposed by Sung and Poggio (1994); the patterns are preprocessed using the normalisation method proposed in the same paper.

In the second stage, overlapping detections are merged and results of two or more networks working in tandem on the same pattern are combined. A face in the image to be analysed is normally detected more than once, the detections being close to each other in position and scale; to merge them the number of detections in the neighbourhood of each location is counted. Those locations where the counter is above a specified threshold are classified as being a face, and the final detection location is defined as the centroid of the detections in the neighbourhood. When a location is classified to be a face, all those other locations that overlap with it and represent a smaller number of detections are discarded.

In order to improve the performance of the classifier, Rowley et al. report the use of more than one network to analyse each pattern. The networks have a similar architecture, and use the same set of face patterns for training, but each uses a different set of non-face patterns; each network generates its own set of non-face patterns by using different initialisation parameters during training (random initial weights, presentation order of non-face images used for the bootstrap method, and initial random patterns). With this the networks, while having a similar performance, will produce different errors, given the different set of non-face patterns used and the different training conditions. In order to arbitrate between two networks, the hypothetical face detections

produced by each network while processing an image are stored in an image pyramid. Rowley et al. then propose two different ways to combine the results. The first option is to use a logical operator AND or OR between the two pyramids, the second is to use a neural network to eliminate false detections.

For testing the performance of the network, they collect a set of 130 test images which contain 507 faces (the CMU test set) and from where 83,099,211 windows were extracted. With a single network without arbitration they report a detection rate of 92.5% with 945 false detections and for a classifier composed of three networks arbitrated with a neural network with 10 hidden units and merging results, a detection rate of 83.6% with 10 false detections was reported, while using the same test set.

In order to allow the classifier to detect faces at any degree of rotation in the image plane an extension of their approach is described in Rowley et al. (1998). They use a *router network* which is fed with the same 20×20 pattern used in the original approach and produces an estimation of the face angle; this angle is used to rotate the original pattern, which is then fed into the classifier.

Osuna et al. (1997) propose the use of Support Vector Machines (SVM) (Vapnik, 1995) to train the classifier. The SVM is a pattern classification algorithm which tries to minimise an upper bound on the generalisation error, instead of the more common approach of minimising the training error. For large data sets, the problem of training the classifiers is a difficult one and to overcome this problem they propose a decomposition algorithm to break down the training problem into a set of smaller problems, which they proved to converge to a general solution. They use a training set of approximately 50,000 19×19 grey level patterns. The non-face patterns were gathered using the bootstrap algorithm.

They use the two Sung and Poggio sets of test images to evaluate the performance of their system, achieving a detection rate of 97.1% with 4 false alarms for the test set A, and a detection rate of 74.2% with 20 false alarms for the test set B. They compare their results with those of Sung and Poggio, achieving a similar performance, although the run-time of their system was estimated to be 30 times faster.

In general, pattern classification methods for face detection have uneven perfor-

mances, as can be seen in table 2.1, although an evaluation of the classifier performance based on the direct comparison of quantitative results is not very meaningful, since there are many factors that vary between approaches, such as the size and quality of the training set, the number of windows analysed during testing and the complexity of the system.

	Detected faces	Detection rate	False Alarm rate
Sung and Poggio (1994)	126/149	84.6%	1/750,997
Osuna et al. (1997)	115/155	74.2%	1/269,184
Rowley et al. (1998) (Without arbitration stage)	469/507	92.5%	1/96,402
Rowley et al. (1998) (With arbitration stage)	424/507	83.6%	1/8,309,921

Table 2.1: Summary of results of pattern recognition approaches for face detection. On the first and second columns the number of detected faces over the number of faces in the test set, and the detection rate is given. Column three gives the false alarm rate (number of false detections per analysed window).

2.2.2 Faces as geometrical configuration of features

In these alternative approaches the face is represented as a set of features which maintain a particular configuration. These features can be classified into two types: internal features (*e.g.* eyes, mouth, eyebrows, nose) or external features (*e.g.* chin, sides and top of the head blob). The use of internal features is a much more popular option, since they are much more stable than external features, which in non-controlled conditions can be affected by the background clutter. The number and type of features used to characterise the face varies: normally the number could vary between two (Han et al., 1997) and six features (Yow and Cipolla, 1997); the ones which are less prone to change with facial expression are selected.

The search for features is normally done by applying one or more operations over the image, such as Gaussian filters, Canny edge detectors (Canny, 1986), morphological operators (Serra, 1982), edge linking algorithms, and template matching. This part varies from approach to approach but the objective is the same: to extract candidate features from the scene.

Most of the feature-based face detection methods use a bottom-up approach, where the full model is constructed iteratively by means of grouping simpler structures. This grouping could be performed in several steps (Yow and Cipolla, 1997) or in one single step (Govindaraju et al., 1990; Han et al., 1997).

Labelling is a common procedure used to reduce the number of tentative face candidates which can be formed out of a set of single features, although from the papers analysed in this review it has only been explicitly mentioned in Leung et al. (1995). Labelling implies the classification of each detected feature as belonging to a particular class (*e.g.* as being an eye, mouth, nose, etc.). If a candidate face is represented with a composite of N features, and M tentative features are extracted from the image, then the number of different combinations of features that can be matched as a face is M^N, which makes the solution of the problem infeasible for a large number of features detected or for a large number of features considered in the representation. If each feature is labelled, the number of possible candidates is reduced to $\prod_{i=1}^{N} M_i$ where M_i is the number of features belonging to class i that were detected in the image.

In 1990, Govindaraju et al. presented one of the first papers where detection of faces is addressed as the main problem. Their approach is oriented to the location of faces in newspaper photographs and, because of this, some constraints are provided for the search process. The faces are the main subject in the image, there are no occlusions, the picture has a good contrast, the people look squarely to the camera and the sizes of the faces fall in a range determined by the size of the photograph and the number of people featuring in it.

They use a set of three external features to model the face: the two sideline curves and the top-head curve which together describe most of the outline of the head blob, and are joined with springs to form a closed loop. No internal features are used in the model.

The first step is the extraction of features, which is done with a set of imaging operations applied to the whole image: edge extraction, thinning the edges, linking them to obtain contours, and then segmenting them at discontinuity points. Each feature is described by a 4-tuple which encodes the position, length, and curvature

of the segments, and finally each feature is labelled as being a right, left or top side of the face. Candidates are then selected by comparing the different combination of features found in the image with the model; a cost function is calculated comparing the parameters of each candidate with the model. The model parameters are determined empirically.

The use of the contour curves of the head as the features used to represent a face is not the best selection since they are not very stable, especially if the background is cluttered. They test their system on 10 images, and report that it always detects the faces in the image, although false detections are often generated.

Leung et al. (1995) propose a feature-based method for face detection in three stages. A face is represented as a collection of five features: the two eyes, two nostrils and the nose-lip junction.

In the first stage the features are extracted from the image by convolving it with a set of Gaussian filters with different scales and orientations. The responses of the set of filters at a particular location is used as a representation of the local image brightness. The features are detected and classified by comparing them with a set of prototype vectors that represents each of the considered features. As was mentioned before, the classification of features into well-defined classes (labelling) helps to reduce the complexity of the search, which takes place in the second stage.

The face model is represented in terms of a multivariate Gaussian distribution of inter-feature distances. In order to achieve scale independence, the model is normalised; a scale factor is estimated for each inter-feature vector extracted from the image prior to matching, under a maximum likelihood criterion.

The search spaces consist of the possible combinations of features that can be matched against the model. Although feature labelling helps to reduce significantly the number of combinations, further improvement is achieved by using the model to guide the search.

An incomplete feature vector is composed by selecting two "strong" features obtained in the first stage, "strong" meaning that the quality of the match during the labelling process is above a predetermined threshold. Using the incomplete feature

vector, the mean and covariance of the location of the missing features are found using the conditional mean estimator and conditional error covariance (Anderson and Moore, 1979). The expected location and error covariance is used to look for the missing features in the image, and if the search is successful a hypothetical face is recorded. This procedure is repeated for each pair of features found in the image.

In the third stage, the group of hypothetical faces found are ranked by comparing them in pairs to decide which one is more face-like. The comparison is set in terms of the probability of a group of features being a face and the probability of not being a face; the former probability is calculated from the estimated detection probability of the features that compose the face, and the statistics of the distribution of the latter by using a Monte-Carlo simulation over a random set of points in the image.

They test their system with a set of 150 test images depicting people with different facial expressions and a cluttered background. They report a detection rate of 86%, and 95% on a subset of images where rotations in depth were eliminated.

Yow and Cipolla (1997) propose a bottom-up, feature-based method for face detection, with three stages: pre-attentive feature selection, attentive feature grouping and Bayesian classification.

Their method is based on a face model which describes the face at different levels of complexity. At the most basic level, single features (eyes, eyebrows, nose and mouth) are described; at the next level these single features are grouped in pairs, and finally at the highest level the pairs are grouped into *Partial Face Groups* (PFG), which are composed of at least 4 single features or into a complete face with 6 features. The PFGs are an alternative way of representing a complete candidate face, allowing the detection of real face candidates which have one or more missing features due to occlusion or loss during the imaging process.

In the first stage, tentative features are extracted from the image using standard imaging techniques (band-pass filtering, Canny edge detection and edge linking). Each feature is then represented with a vector which encodes feature information (edge length, edge strength and grey level variance of the region). The four feature classes are modelled as Gaussian distributions, described by a feature mean and covariance

matrix. These models are constructed from a set of features selected from training images. Each candidate feature found is then classified as belonging to one of the classes, by comparing the Mahalanobis distance between the feature vector and the mean vector of each class with a predefined threshold. The features that can not be classified as belonging to a particular class are discarded.

In the second stage the single features found in the first stage are grouped into face candidates. First, the single features are grouped into vertical and horizontal pairs (*e.g.* right and left eye form a horizontal pair, mouth and nose form a vertical pair), then the pairs are grouped into PFGs, and the PFGs into face candidates. Every group formed in this way is represented with a vector that encodes certain characteristics extracted from the region drawn between the two sub-structures that are being grouped (*e.g.* ratio of feature lengths, aspect ratio of feature region, mean grey level in the region). A Gaussian model is used to represent each of the groups defined in the face representation and a procedure similar to the one used in the first stage is used to classify them.

The purpose of the last stage is the rejection of false face candidates obtained in the previous stage. For this purpose, each of the face candidates are presented to a belief network (or Bayesian network), which encodes dependencies between the different features of a face candidate. A probability measure is assigned to each of the single features that compose a face candidate; this measure depends on the ratio between the Mahalanobis distance between the feature and its class and the corresponding rejection threshold. Those values are propagated through the network and a posterior probability measure for the face candidate is obtained.

They test their approach on a database of 110 images of faces with variations in the scale, orientation and viewpoint, achieving a face detection rate of 84.5% with 31 false detections. It is important to note that in order to detect faces at different scales, the parameters of the filters used in the first stage should be tuned to a particular value in order to achieve an optimum performance, although a certain degree of scale independence is achieved. It has also been shown that their system is capable of dealing with rotated and occluded faces.

Han et al. (1997) proposed a hybrid face detection system with three stages. In

the first stage, they locate eye-like segments in the image using a set of morphological operators. From that set of eye-like segments, potential eye pairs are selected in the second stage by eliminating those combinations which do not comply with a set of geometrical constraints. The last step is verification. They use a neural network classifier similar to the one used by Rowley et al. (1995) as a coarse verifier to eliminate false face-like candidates, then the "distance-from-feature-space" proposed by Moghaddam and Pentland (1995a) is applied as a fine verification step in order to eliminate the remaining non-face candidates and eliminate overlapping.

Although this approach has all of the characteristics of a feature-based detection algorithm, the last two stages use two different pattern recognition techniques to eliminate false detections. This could be considered as a way of dealing with the localisation problem (see page 26), by means of using low cost global operations to obtain a set of feature-based face candidates, that can be used to drive the search of a more computationally expensive pattern recognition method.

They test their system with a test set of 122 images containing 130 faces with different orientations on a cluttered background. They report a detection rate of 93.8% with 25 false detections. Their system is remarkably invariant to rotation (it can detect faces up side down).

The results obtained with this kind of classifier are comparable to those of the pattern recognition methods in terms of the detection rate and in number of false alarms, although it has to be said that in most of these cases, the feature based face detectors were tested using a homogeneous test set, with relatively large faces and good quality. A summary of the results obtained by the methods described in this subsection are given in table 2.2.

	Detected faces	Detection rate	False Alarms
Leung et al. (1995)	120/150	86%	Not Reported
Yow and Cipolla (1997)	93/110	84.5%	31
Han et al. (1997)	122/130	93.8%	25

Table 2.2: Summary of results of feature based approaches for face detection. On the first and second columns the number of detected faces over the number of faces in the test set is shown, and the detection rate is given. Column three gives the number of false alarms.

2.3 Discussion

2.3.1 Comparison of Performance

In order to make a fair comparison between the results of different approaches, it is necessary to have a standard set of images that can be used as benchmark to compare results between different approaches. A good example is the Carnegie Mellon University (CMU) image set which can be found on the world wide web[6]. There are two sets, the first one containing 107 images, which vary in quality and size which were obtained from the world wide web, broadcast television and scanned from photographs, newspapers, magazines, and also the 23 images used by Sung and Poggio. This gives a total of 130 images with 507 frontal faces in them. The second set consists of a subset of the FERET database (Phillips et al., 1997), which is a set of images with good illumination and plain background, and only one face per image. The purpose of this database is to be used to test the sensitivity of the detector to the variations in the view-point. This image database, or at least past of it (the test set of Sung and Poggio) has been widely used by most of the face detection approaches published after 1994; however it is not a common choice for the approaches that represent a face as a set of geometrical features, since the minimum size of a detectable face in these cases tend to be large (> 40 px^2) and most of the images in the CMU database contain small sized faces. Each of the feature based approaches uses a different image database for the evaluation, and there is no common test set.

Most of the latest systems report a performance that varies from 80% to 98%, with a reasonable number of false detections. However, the complexity of the different approaches tends to vary, and with it the computing time needed to analyse an image.

2.3.2 Face location

It is important to make a distinction between "face detection" and "face location". Face detection, as defined at the beginning of this chapter (see page 9), is "the classification of a stimulus as being a face or not". On the other hand, face location is the problem of finding where in the image the face is located.

[6] http://www.ius.cs.cmu.edu/IUS/har1/har/usr0/har/faces/test/

For the grey level patterns approach to face detection the problem is well differentiated; the only task of the classifier is to distinguish between a face and non-face pattern, the problem of looking for it is a different one. Normally the search is an exhaustive process which implies testing each pixel of each image plane of an image pyramid; in the different reports the performance of the classifier *per se* is the main issue, and the location problem is not directly tackled, although some authors propose methods to speed up the process. Vaillant et al. (1994) use networks trained to perform a rough localisation, and then use another network to refine the results. In a similar vein, Rowley et al. (1998) propose a modification of their approach based on using a classifier which detects faces which are at most 5 pixels off-centre in a 30×30 window, allowing by this the use of bigger steps while scanning the image. Ben-Yacoub (1997) proposed a method for fast face detection using neural networks, by using the analogy of the neural network as a bank of filters that are applied to the whole image, and reformulating the problem in terms of convolutions, which can be efficiently calculated in the frequency domain. It is important to note that the preprocessing procedures used by other methods have to be modified since the histogram equalisation and illumination correction are designed to work with individual patterns and are not suitable for global processing. The authors use instead a normalisation and centering process that can be expressed in terms of convolution operations. A complete system for face detection based on the work of Ben-Yacoub is described in Fasel (1998).

In the feature detection methods, it is not possible to differentiate between face detection and location. In these approaches the first step is to extract the different features that define the face representation, and the face candidates are selected by choosing those features that can be combined to form a face, according to the model used to represent it.

The type of representation used defines the way the search is performed. Tsotsos (1990) classifies visual search in two categories: bottom-up or unbounded visual search and task-directed or bounded visual search.

With unbounded visual search the goals are not known in advance, or even if they are they are not used to direct the search. This is the case in feature-based methods, — at least in its coarsest conception — where a prototype face can be composed of features

found anywhere in the image (it is unbounded), and although in our case we know the goal in advance (in terms of the model that describes the relationships between the different face features), the only role the face model plays in the search process is to decide when a set of features in the image constitutes a correct match. It was proved by Tsotsos (1989) that unbounded visual search is an NP complete problem, being inherently exponential in the size of the image and therefore not computationally feasible.

On the other hand, the pattern classification approaches to face detection fall in the bounded visual search category, where the target is used to direct the matching process by limiting the image area where the matching procedure is applied. This constraint forces us explicitly to test every single position of the images at different scales. While this is a computationally expensive task it is feasible and it scales well. It was also shown by Tsotsos (1989) that bounded visual search has linear time complexity; *i.e.* the necessary time required to test the whole image increases linearly as the size[7] of the image does.

In practice, feature-based methods apply different strategies to make the solution of the problem viable: labelling is normally used in order to avoid the combinatorial explosion produced by considering a large number of features, and the face model is used to direct the search by constraining the number of possible combinations that can be classified as a face.

One of the most interesting characteristics of the feature-based methods for face detection is scale independence, given the nature of the search process where there is no spatial boundary to constrain the size of the possible face candidates. In practice, this advantage is quashed by the different constraints used to reduce the complexity of the system, *e.g.* labeling implies the classification of a feature as belonging to a particular class, a bounded search problem on its own, which is not implicitly scale independent.

[7] Number of pixels in the image.

2.4 Conclusion

The pattern recognition methods seem to be a more appropriate method to detect faces, because they scale better, with linear time complexity, and are not constrained by scale.

The main drawback to a pattern recognition method, and a problem still to be solved, is localisation: all the approaches considered in this review use a brute force method, analysing every single pixel in the scene at different scales. This is particularly important when a real-time response is needed *e.g.* human-machine interaction, surveillance, robotics. The next step is to define how to constrain the search space using an attentional process to guide the search.

Chapter 3

Face Detection: Implementation

3.1 Introduction

This chapter is devoted to a detailed description of the implementation of the classifiers used for the face detection system. Two different kinds of classifiers are tried: the first method is a variant of the multi-layer Perceptron (MLP) architecture proposed by Rowley et al. (1995), and the second is similar to the one proposed by Sung and Poggio (1994).

Each classifier was trained using two different test sets, one composed entirely of front view faces and the other one with side view faces, with the purpose of increasing the range of face orientations that can be classified. Each classifier was tested using the test sets of Sung and Poggio (1998) and Rowley et al. (1998).

The chapter is organised as follows: first a description will be given of the training sets used and how they were generated, followed by a detailed description of the implementation of each classifier. An explanation of the merging procedure used to eliminate multiple detections is given afterwards and finally the method used to test the classifiers' performance is presented followed by a summary of the results obtained and a discussion.

3.2 The Training Set

Two different kinds of training sets were generated: one for front view faces, and one for side view faces. In the rest of this section a description is given of how the face and non-face patterns which compose the different training sets were obtained.

3.2.1 The face patterns

The face patterns were extracted from a set of portrait images; 614 of them were classified as being front view faces, and 242 as being side view faces. 343 portraits were collected in the laboratory and the rest were obtained from the Internet and from face databases of other research centres (see appendix B). All the side view images depict a face with a right side view orientation.

From each portrait image an inner face area was hand-segmented. The inner face area is the one that encloses the main face features: eyes, eyebrows, nose and mouth. Since the size and position of the faces found in the portrait images had a wide degree of variation, a normalisation procedure was needed. For this reason a set of fiducial points were selected for each view: for the front view faces the external corners of the eye, the tip of the nose and the centre of the mouth were used; for the side view faces the external corners of the eyes and the nose lobe were used. The criterion used to choose the fiducial points is stability, since their position does not considerably change with different facial expressions.

To normalise the size of the faces a scaling factor is computed using the geometrical relationship between the different fiducial points extracted. For the front view the scaling factor is the distance between the middle of the segment that joins the two corners of the eyes, and the mouth; and for the side view faces it is the distance between the middle of the segment that joins the two corners of the eyes, and the lobe of the nose (see figure 3.1). In both cases the orientation of the faces is normalised using the angle of the segment that joins the two corners of the eyes with respect to a horizontal.

From each normalised and segmented face, 28 versions of it are artificially generated

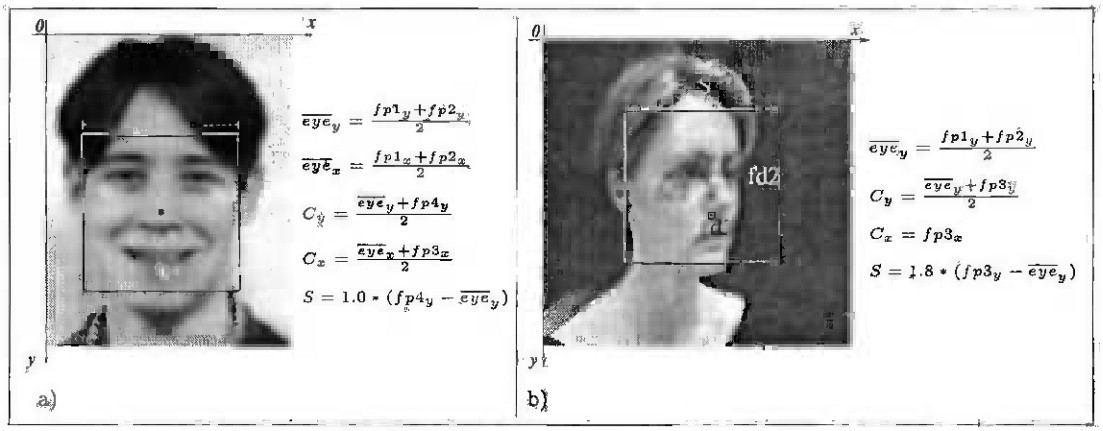


Figure 3.1: Face Segmentation: (**a**) front view case, (**b**) side view case. In both cases the fiducial points used to normalise position and scale are shown, with the arithmetical relationship between them that define the centre and scale of the segmentation square.

for the front view faces, and 40 for the side view faces. These versions are generated by randomly varying the size of the image within a range of $\pm 10\%$, and by rotating the image with a random angle between $\pm 10°$. Since front view faces represent a vertically symmetrical object, half of the 28 generated images were obtained from the original segmented image, and the other half from a horizontally flipped version of it.

After normalising the size and orientation, each generated image is scaled to obtain a grey-level pattern of 20×20 pixels using eight bit quantisation.

As explained in the previous chapter, it is a common procedure to preprocess the patterns in order to minimise the variance produced by different illumination conditions and variations of the histogram. For the implementation of the two classifiers described in this chapter, the preprocessing method first proposed by Sung and Poggio (1994) was used. The procedure consists of three steps: masking, illumination correction and histogram equalisation.

Masking is necessary in order to normalise the boundary of the segmented inner face, and to eliminate parts of the background that might have been included in the pattern. Two different masks were used, one for each view. The front view mask is similar to the one used by Sung and Poggio (1994) and Rowley et al. (1995), and in the case of the side view faces, the mask was defined by eliminating the background on the average of the side view patterns obtained (see figure 3.2). After masking, the

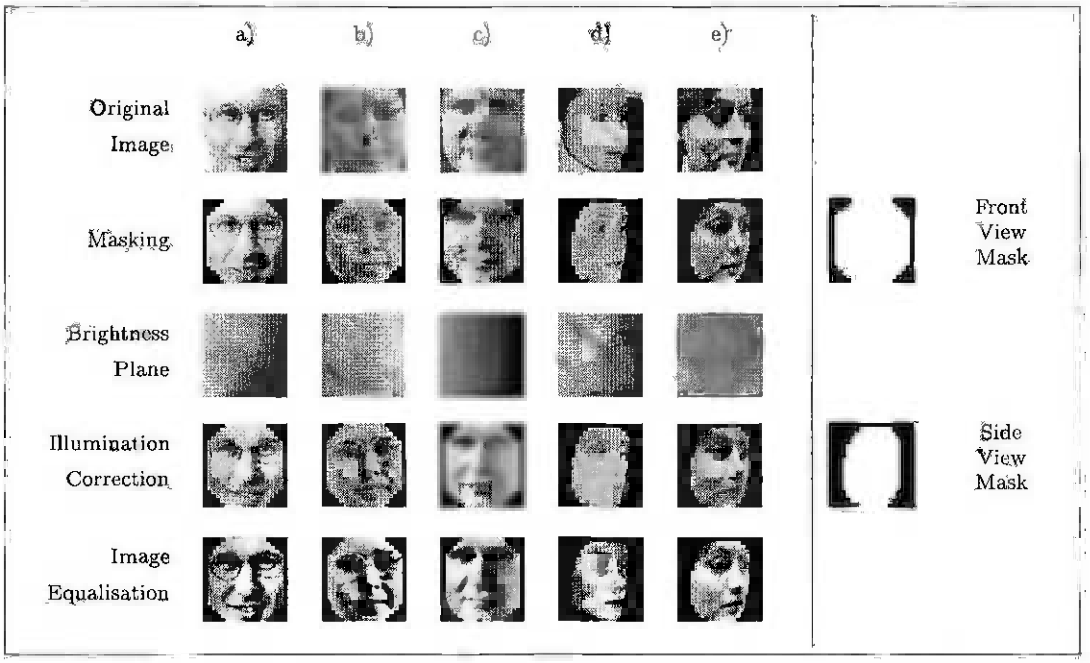


Figure 3.2: Face preprocessing: columns (**a**), (**b**) and **c**) are examples of front view faces, columns (**d**) and (e) are side view faces.

front and side view patterns have only 320 and 234 significant pixels respectively.

The illumination correction procedure is accomplished by subtracting a brightness best fit plane from the 20×20 pattern. The best fit plane is obtained in the following way: first an average vertical and horizontal brightness is computed for each image column and row (YL and XL respectively on figure 3.3), then a line is fitted to XL and YL using linear regression (XI and YI on the same figure). The best fit plane is the one that intersects the lines XI and YI, whose independent parameter has been adjusted so that they intersect on the origin. The difference between the pattern and the brightness plane is then normalised so the different values of the pattern vary between 0 and 255.

The next step is to equalise the grey-level pattern histogram; with this I expect to distribute the brightness levels in the image along the whole brightness scale, which improves the contrast of the image. A detailed description of this technique can be found in Sonka et al. (1993).

Using the procedure described in this section, a total of 17,181 and 9,680 different patterns were generated for the front and side view cases respectively.

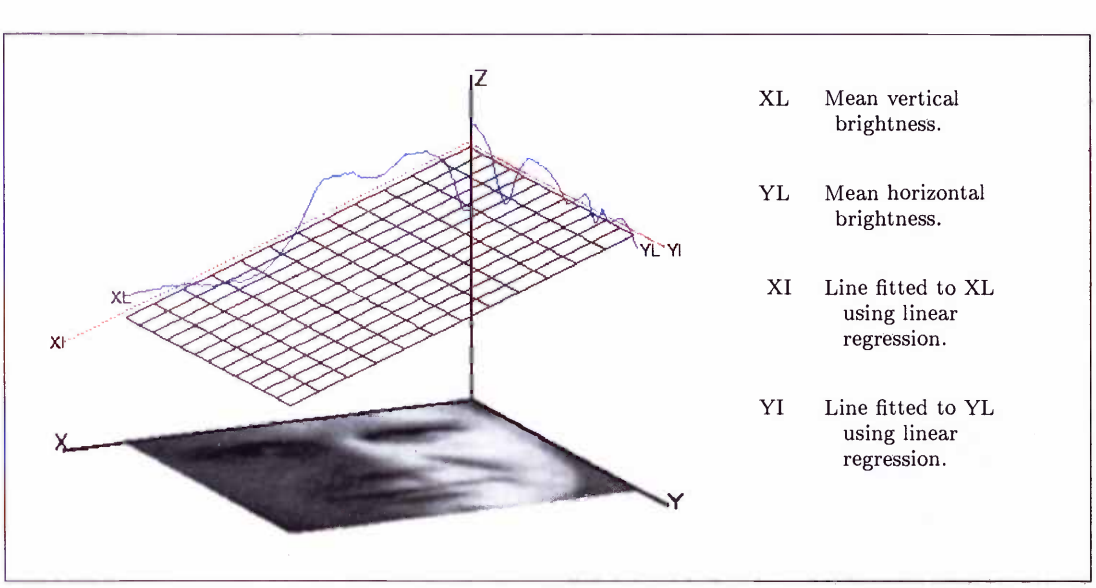


Figure 3.3: Fitting plane to a image brightness.

3.2.2 The non-face patterns

The generation of the face pattern set is a straightforward procedure since there is an explicit source of data from which we can extract the patterns (*i.e.* face portraits). However the face set is obviously not an exhaustive enumeration of all those patterns that represent faces, but instead it is a representative collection of face patterns; from such a training set we expect the classifier to be able to generalise into a wider set of face patterns.

In the case of the non-face set, the number of possible patterns that can belong to it is very large and it is important to find criteria to select those patterns that would form a representative set. For example, it would be useful to select those patterns that lie on the boundary of the face set, the ones that are likely to be classified as being faces when they are not.

This is the paradigm on which the bootstrap method proposed by Sung and Poggio in 1994 is based. It iteratively selects non-face patterns from images without faces, using a partially trained classifier. After each iteration a certain number of non-face patterns that were found in the images without faces are added to the training set, and the classifier is retrained with the new enlarged training set; this procedure is repeated until the number of erroneously extracted faces is small or we have collected

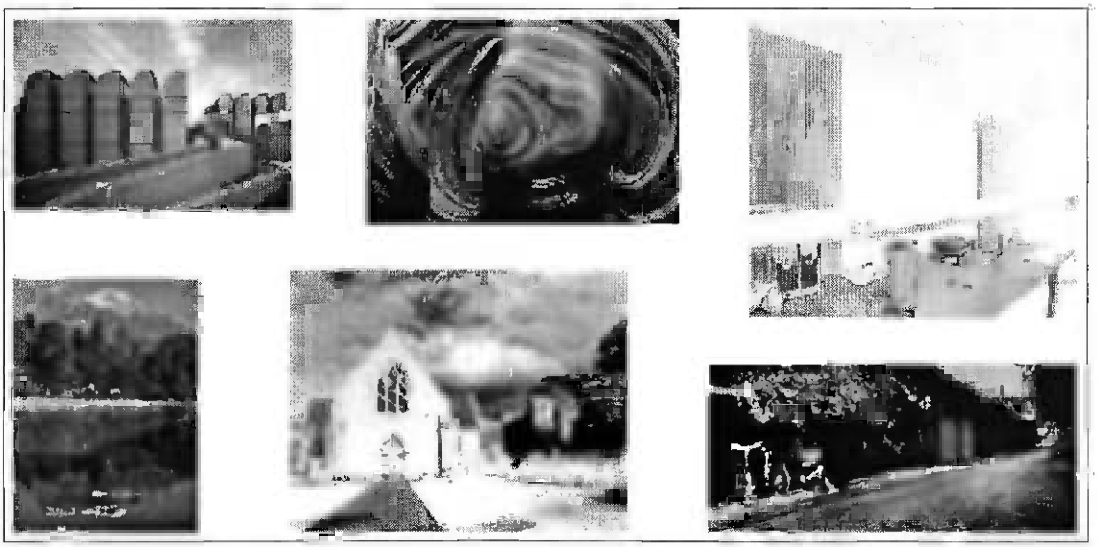


Figure 3.4: Examples of the images used to extract the non-face patterns.

a reasonable number of non-face examples. The method has been widely used in the context of face detection in a number of different approaches (*e.g.* Rowley et al., 1995; Osuna et al., 1997; Colmenarez and Huang, 1997). This is the method used here to generate the non-face training sets.

First a set of images which do not contain faces have to be collected. In our case 278 images were collected, mainly from the Internet, although some were scanned from magazines and photographs, and the rest are images grabbed in the laboratory with a CCD camera (see figure 3.4).

The CNN classifier was used for the generation of the training set — a detailed description of it is given in the section 3.3. Initially the classifier is trained using a set which is composed of the face patterns previously generated and 1,000 random patterns as counter-examples. Once the classifier has been trained, it is used to search for face-like patterns in the face-less images previously collected. The position and scale of each of the patterns extracted from each image is selected at random. In the work described here 10 different images were used at each iteration, and from each of them a maximum of 250 patterns were selected. After each iteration the extracted non-face patterns are added to the training set. The network is trained again using the enlarged training set, and the procedure is repeated. See table 3.1.

1. Train the network with the face patterns & 1000 random non-face patterns.
2. Discard the non-face patterns from the training set.
3. Using the trained network, randomly extract 'faces' from a set of 10 non-face images. From each image extract 250 patterns and then add them to the training set.
4. Retrain the network with the new training set.
5. If no new non-face patterns are found, or the total number of non-face patterns extracted is big enough, finish; otherwise to step 3.

Table 3.1: The bootstrap algorithm.

Six different training sets were generated using this method: three for the front view case and three for the side view case. In each training set the same front and side view patterns were used, while the non-face patterns are different in each case; this was achieved by first, using different initial parameters before training the classifiers, and second, using a different presentation order of the images without faces during the bootstrap algorithm. Each classifier evolved differently, although all of them were trained to perform the same task.

For the front view classifiers, the number of different non-face patterns in each training set are 10,230, 10,198 and 10,509, while for the side view classifiers 12,730, 13,564, and 12,571 non-face patterns were generated for each training set.

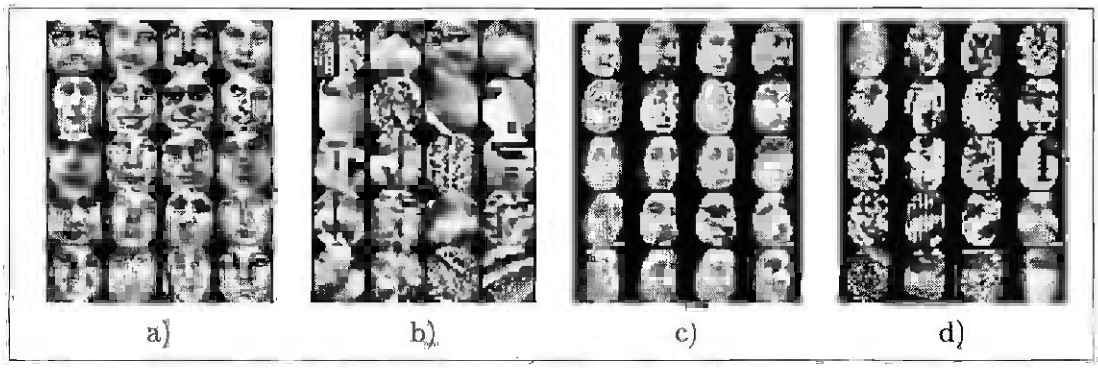


Figure 3.5: Examples of the training sets: (**a**) Front view face patterns, (**b**) Front view non-face patterns, (**c**) Side view face patterns, (**d**) Side view non-face patterns.

3.3 The Classifier: CNN Classifier

The convolutional neural network (CNN) is a multi-layer perceptron that is normally used for classification of visual patterns. This architecture was first proposed by LeCun et al. (1989), with the purpose of achieving a certain degree of translation and distortion invariance on two-dimensional pattern recognition problems. This architecture and some variations of it have been successfully used for optical character recognition (*O*CR), speech processing, on-line handwriting recognition and face detection among others (LeCun et al., 1990; Lang et al., 1990; Guyon et al., 1991; Vaillant et al., 1994, respectively). The three main characteristics that define a CNN classifier are: the use of retinal connections (receptive fields), weight-sharing and sometimes spatial or temporal sub-sampling.

A receptive field is a concept borrowed from physiological models of the retina of vertebrates and is defined by Bruce et al. (1996) as *"the area in the retina in which light causes a response in a particular nerve cell"*. In connectionist models, a receptive field occurs when a perceptron unit is not fully connected to all the outputs of the previous layer units, but only to a small contiguous area of them. There are two benefits which are clearly obtained from the use of receptive fields: it is possible for the network to take advantage of the two-dimensional structure normally found in natural patterns where there is a high local correlation, and the number of parameters to be trained is considerably reduced, thus allowing a reduction in the size of the training set without causing over-fitting.

Another important feature of the CNNs is the use of weight-sharing, which was originally introduced by Rumelhart et al. (1986) to achieve translation and rotation invariance (the T-*C* problem). Weight sharing occurs when a set of units which have retinal connections to the previous layer have the same set of weights (it is assumed that the receptive field's shape and size covered by each unit is the same). The effect of sharing weights is that different units are performing the same operation over different parts of the input, and can be considered as feature detectors, with the main difference being that the parameters of the detector are not set *a priori*, but instead are defined during the training process in terms of what is convenient for the minimisation of the

error. The use of weight sharing also diminishes the number of parameters to train without decreasing the connectivity between layers.

The search for features in an image is normally carried out by computing the convolution of the image with a feature detector. An implementation of this procedure can be achieved by using a neural network in which the units in the layer are arranged on a grid, each of them covering a particular area of the previous layer and sharing weights. The result of the propagation of the input values will produce a "feature map" on the output of the units that compose the grid. The only difference from a feature map produced by the conventional method is that the feature map is "compressed" due to the activation function of the units, and the different thresholding values (biases) used in each unit.

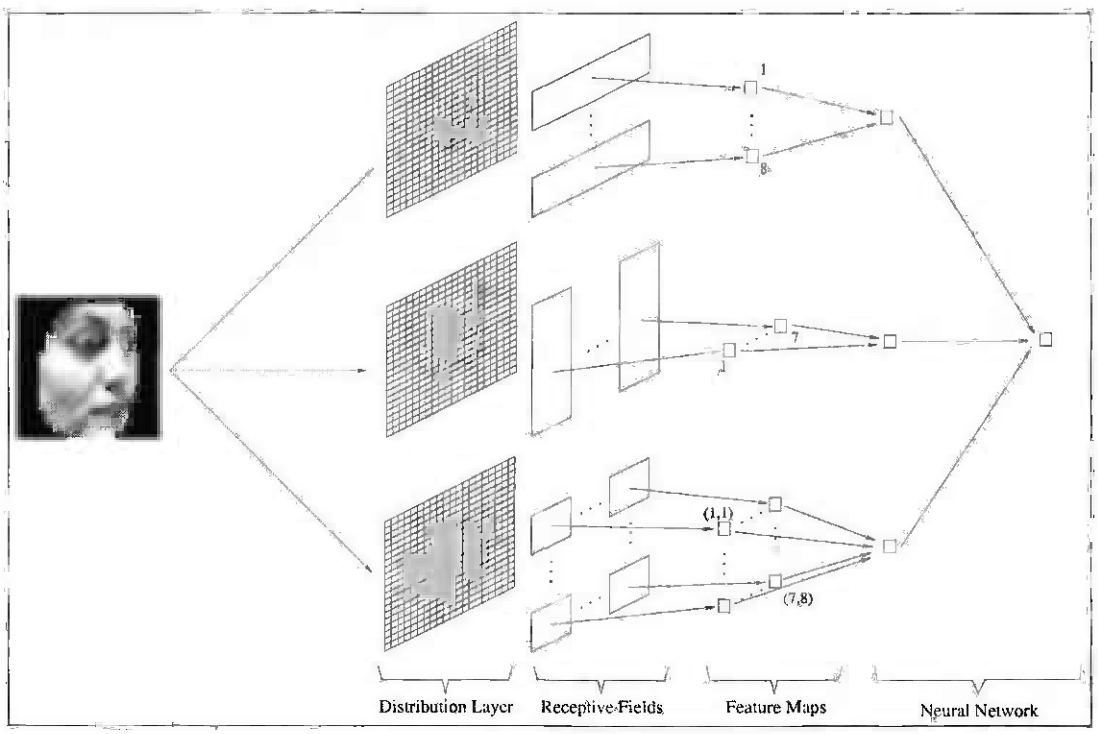


Figure 3.6: Convolutional classifier.

The architecture of the network described in this section is very similar to the one used by Rowley et al. (1995), the main difference being the use of weight sharing. In their approach (see section 2.2.1) receptive fields are arranged in a grid as a convolutional neural network, but without sharing weights between units.

The network used here is composed of 4 layers: an input or distribution layer, 2 hidden layers and an output layer. The distribution layer is a grid of 20×20 pixels, which receive inputs in the interval [0, 1]. The second layer of the network is composed of one or more groups of feature maps, each of them containing three feature maps, each of them characterised by the shape of the receptive field used: the first feature map has 8 units arranged in a column with a 18×6 receptive field, the second one has 7 units organised as a row with a 6×20 receptive field and the last one is composed of 56 units organised in a 7×8 matrix with a 6×6 receptive field. The shape and size of the receptive fields used was designed to match those features that are likely to occur in face patterns (*e.g.* the eyes, mouth and nose with the first two feature maps), or more general features (*e.g.* corners, edges, with the third feature map). The third layer is composed of one unit per group of receptive fields, each of them being fully connected to a particular feature map. The last layer is a single unit connected to all the units in the third layer (see figure 3.6). All the units of the network have a hyperbolic tangent activation function. The network was trained to give an output of +1 when a face pattern was presented and −1 otherwise.

The training method used was back-propagation with momentum (Plaut et al., 1986). As mentioned in section 3.2.2, six networks were trained, each with a different set of non-face patterns. Each network has two sets of feature maps with a total of 7,588 connections and only 825 free parameters. The weight sharing was achieved during training by averaging the weight update values of those shared connections in each layer before weight update.

3.4 The Classifier: Sung Approach

Another face detection method was implemented mainly for comparison reasons. The approach selected was the one proposed by Sung and Poggio (1994), which was reviewed in chapter 2. The face and non-face pattern distribution is modelled using a set of 12 Gaussian clusters. In order to classify a pattern, a two-valued metric is calculated between the pattern and each of the model centroids. This set of distances is fed into a multi-layer Perceptron trained to classify between a face and a non-face pattern.

The same training set used to train the CNN classifier was used for the training of the classifier described in this section. Each two-dimensional pattern is transformed into a column vector $\mathbf{x}$ by first eliminating those elements of the 20×20 matrix that are masked, and then re-ordering the pixels into a single column. With this we achieve a reduction of the number of dimensions of the vector space needed to represent the faces: instead of using a 400 dimensional space for the front or side view pattern we only need a 320 or 234 dimensional space respectively.

Both face and non-face patterns are modelled using sets of k Gaussian clusters; in this implementation a value $k = 6$ was used. Each cluster is composed of n_j vectors and determined by two parameters: the sample mean vector μ_j and the sample covariance matrix Σ_j, which are defined by

$$\mu_j = \frac{1}{n_j}\sum_{i=1}^{n_j}\mathbf{x}_{ji} \quad , \quad \Sigma_j = \frac{1}{n_j - 1}\sum_{i=1}^{n_j}(\mathbf{x}_{ji} - \mu_j)(\mathbf{x}_{ji} - \mu_j)^\top \quad , \quad j \in 1 \cdots k \tag{3.1}$$

In order to create the cluster a variation of the K-means algorithm is used; the main difference is that it uses the normalised Mahalanobis distance instead of the standard Euclidean distance. In order to understand the properties of the normalised Mahalanobis metric, the standard Mahalanobis distance will be discused first.

The Mahalanobis distance between a vector $\mathbf{x}$ and a cluster centroid described by a Gaussian distribution defined in terms of the mean vector μ and the covariance matrix Σ is defined as

$$\mathcal{D}(\mathbf{x}, \mu, \Sigma) = (\mathbf{x} - \mu)^\top \Sigma^{-1} (\mathbf{x} - \mu). \tag{3.2}$$

The difference between the Euclidean and the Mahalanobis distance is that while with the Euclidean distance a set of equidistant points to the center of the cluster centroid will form a spherical surface, with the Mahalanobis distance they will form an ellipsoid surface whose shape and orientation is determined by the covariance matrix Σ. The Mahalanobis distance can be conceived as a "weighted" Euclidean distance, where the orientation of the ellipsoid surface is aligned with major axis of the cluster. In figure 3.7 we can see a representation of two sets of points equidistant from a Gaussian centroid using Euclidean and Mahalanobis metrics: in both cases the distance between the set of points on each surface and the centroid is the same (6 units), but it can observed how the Mahalanobis distance provides a better description of the data cluster.

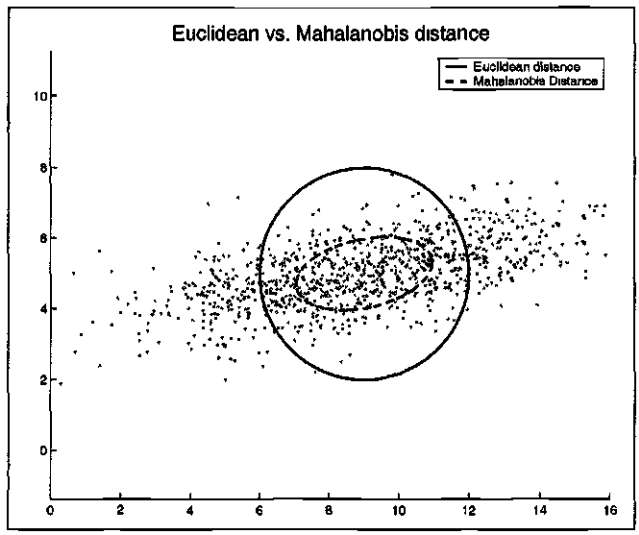


Figure 3.7: Euclidean *vs.* Mahalanobis distance.

The normalised Mahalanobis distance between the vector $\mathbf{x}$ and the j-th cluster modeled by the Gaussian distribution $\mathcal{G}(\mu_j, \Sigma_j)$ is defined by

$$\mathcal{D}(\mathbf{x}, \Sigma_j, \mu_j) = -\ln(\mathcal{G}(\mu_j, \Sigma_j)) = \frac{1}{2}(N \ln 2\pi + \ln|\Sigma_j| + \bar{\mathbf{x}}_j^{\top} \ \Sigma_j^{-1} \ \bar{\mathbf{x}}_j) \tag{3.3}$$

where N is the dimensionality of the vector $\mathbf{x}$ and $\bar{\mathbf{x}} = \mathbf{x} - \mu$. The difference between the Mahalanobis distance and its normalised version is that the latter originates directly from the Gaussian probablility that models the cluster, which integrates to unity.

As it was pointed out by Sung and Poggio (1994), the use of the normalised version of the Mahalanobis distance is important for stability reasons when used in a "K-means" based algorithm, where distance between a vector and a set of cluster centroid have to be calculated. While the Euclidean distance gives absolute values, the standard and normalised Mahalanobis distance between a vector and a cluster is relative to the variance inherent in the cluster; with the standard Mahalanobis distance, a spatial displacement between a vector and a cluster centroid for large clusters with large covariance tends to produce smaller distance measures than it would for a smaller cluster. Therefore, if we use the standard Mahalanobis distance, we have the risk of large clusters increasing their size and eventually engulfing the small clusters. The normalised Mahalanobis distance minimises this effect by having a term in the equation which is directly proportional to the cluster variance ($\ln|\Sigma|$), thus minimising the effect described above.

1. Obtain K=6 cluster centres from the training set using Learning Vector Quantisation (LVQ). Assign each pattern in the training set to the nearest cluster centre using Euclidean distance.
2. Initialise the covariance matrices of each cluster to be the identity matrix.
3. Recalculate the cluster centres as the mean of the patterns assigned to it.
4. Assign each pattern to the nearest cluster centre using the normalised Mahalanobis distance. If there are no changes in the composition of the clusters (no patterns are re-assigned) or the number of internal-loop iterations (*i.e.* steps 3 to 4)have exceeded a certain limit go to step 5, otherwise go to step 3.
5. Recalculate the covariance matrices for all the clusters.
6. Assign each pattern to the nearest cluster centre using the normalised Mahalanobis distance. If there are no changes in the composition of the clusters (no patterns are re-assigned) or the number of external-loop iterations (*i.e.* steps 3 to 6) have exceeded a certain limit finish, otherwise go to step 3.

Table 3.2: The elliptical K-means Algorithm (Sung and Poggio, 1994).

A description of the normalised Mahalanobis version of the K-means algorithm is given in table 3.2.

Once the distribution model has been generated, principal component analysis (PCA) (Jolliffe, 1986) is applied to each cluster, which involves solving the equation

$$\Sigma\Phi = \Phi\Lambda \tag{3.4}$$

where Φ and Λ are respectively the eigenvector and diagonal eigenvalue matrices of the covariance matrix Σ. The eigenvector matrix obtained defines a new orthonormal coordinate system where the eigenvectors act as the principal components and the corresponding eigenvalues are the variances of the data projected into it (Fukunaga, 1990).

Principal component analysis is mainly used to reduce the dimensionality of the data by mapping it into a lower dimensional space; for this the set of the M eigenvectors with the largest eigenvalues is used to construct a sub-matrix Φ_M which is used to project an N-dimensional vector $\mathbf{x}$ into the M-dimensional subspace spanned by Φ_M using the equation $\mathbf{y}_M = \Phi_M^{\mathsf{T}}\bar{\mathbf{x}}$, where $\bar{\mathbf{x}} = \mathbf{x} - \mu$.

Each pattern is represented by the set of distances between the pattern and each

of the twelve clusters. Each distance is composed of two values, the first one being the normalised Mahalanobis distance between the pattern and the cluster centroid on the subspace spanned by the M_j largest eigenvectors, also called "distance in feature space" (DIFS), and the second one is the Euclidean distance between the pattern and its projection in the subspace, also called "distance from feature space" (DFFS) or residual reconstruction error. This two-valued distance metric has been also used for object detection (Moghaddam and Pentland, 1995b). The criterion to determine the number of effective eigenvectors used in each cluster is to eliminate all those trailing eigenvectors such that the sum of the omitted eigenvalues is smaller than the largest eigenvalue. In the original paper by Sung and Poggio the number of eigenvectors was standardised to 75 for all the clusters used; in the implementation of their algorithm described here the criterion just mentioned gave a different value for the two different classes of clusters, in the two different views. For the front view classifier, 99 and 163 eigenvectors were used for the face and non-face clusters respectively and for the side view faces 32 and 85 eigenvectors were used for the face and non-face clusters respectively.

The normalised Mahalanobis distance in the M-dimensional subspace (DIFS) is then defined by

$$\begin{aligned} \mathcal{D}_1 &= \frac{1}{2}(M \ln 2\pi + \ln|\Sigma_M| + \bar{\mathbf{x}}^\top \Sigma_M^{-1} \bar{\mathbf{x}}) \\ &= \frac{1}{2}(M \ln 2\pi + \ln|\Sigma_M| + \bar{\mathbf{x}}^\top [\Phi_M \Lambda_M^{-1} \Phi_M^\top] \bar{\mathbf{x}}) \qquad (3.5) \\ &= \frac{1}{2}(M \ln 2\pi + \ln|\Sigma_M| + \mathbf{y}^\top \Lambda_M^{-1} \mathbf{y}) \end{aligned}$$

(3.6)

where Λ_M is the diagonal matrix of the M largest eigenvalues. For the implementation the second and third terms of the equation 3.5 are expressed in terms of the λ_i

$$\mathcal{D}_1 = \frac{1}{2}(M \ln 2\pi + \sum_{i=1}^{M} \ln \lambda_i + \sum_{i=1}^{M} \frac{y_i^2}{\lambda_i}) \qquad (3.7)$$

— this is more convenient when the matrix Σ_M is near-singular, and speeds up the computation (Fukunaga, 1990).

The distance from feature space (DFFS) is defined as the Euclidean distance between the pattern and its projection on the subspace and it is equivalent to the residual

reconstruction error, which is defined by

$$\mathcal{D}_2 = \sum_{i=M+1}^{N} y_i^2 \tag{3.8}$$

The training set is formed with the combination of the face patterns and the three sets of non-face patterns obtained by the procedure described in section 3.2. After combining them and eliminating redundancies I obtain for the front view case a training set with 17,181 face and 30,845 non-face patterns; in the side view case the training set is composed of 9,680 face and 29,944 non-face patterns. Each pattern in the training set is transformed into a vectorial representation and the two-value distances to each cluster are calculated and saved as a new training set, which is used to train the classifier.

The classifier used is a MLP composed of three layers: the input or distribution layer which has 24 units, the hidden layer which has 18 units[1] and the output layer with a single unit. Each layer is fully connected to the previous one and all the units have a hyperbolic tangent activation function. The network was trained to give an output of $+1$ if the pattern is a face and -1 otherwise.

3.5 Searching for faces and merging results

To evaluate the performance of the classifiers we need to implement a procedure to locate faces in images. A problem to be solved is that the classifiers were trained to detect 20×20 face patterns and although they have a certain degree of scale invariance, the images to be scanned have to be transformed in order to achieve detection over a wider range of scales. The common approach for this problem is to represent each image with an "image pyramid" (Tanimoto and Pavlidis, 1975) which consists of a set of scaled down copies of an original image. An image pyramid is defined by two parameters: the number of levels of the pyramid l and the scale difference between levels S. In the work described in this chapter a Gaussian pyramid (Burt, 1988) (where the original image is convolved with a Gaussian filter before scaling it down), was used.

[1] Different numbers of hidden units were tried, and the best results were obtained with 18 units.

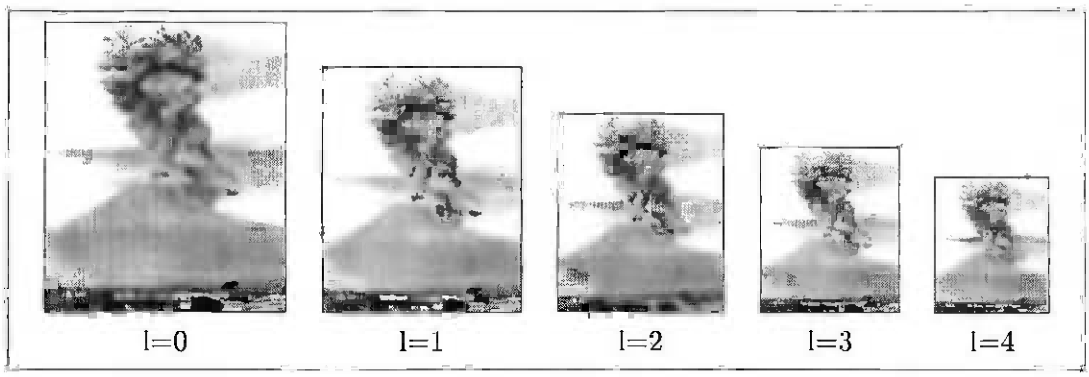


Figure 3.8: An example of an image pyramid.

The scale factor used was determined by taking into account the tolerance to scale variation of the classifier. As it was explained in subsection 3.2.1, an artificial ±10% face-size variation is induced when creating the training set, and we expect that the clasifier will correctly classify faces within such limits.

Since the absolute face scale variance is 20% a scale factor of 6/5 was considered to be appropriate, but since we consider the original image as being the lowest level in the Gaussian pyramid, which needed to be scaled down in order to create the pyramid, the inverse of the scale factor mentioned before was used ($f = 5/6$).

The number of levels of the pyramid is such that the smallest side of the image on the top level is bigger than 20 pixels. The use of this representation guarantees that faces at different scales will be represented as a 20×20 pattern in one level of the Gaussian pyramid. An example of an image pyramid is shown in figure 3.8.

The search for faces is an exhaustive process where a 20×20 pattern is extracted from every position at every level of the Gaussian pyramid and tested using the classifier; if the pattern is classified as being a face, the output value of the classifier and the position of the detection in the original image (level 0 of the pyramid) and the level of the pyramid where it was found are recorded.

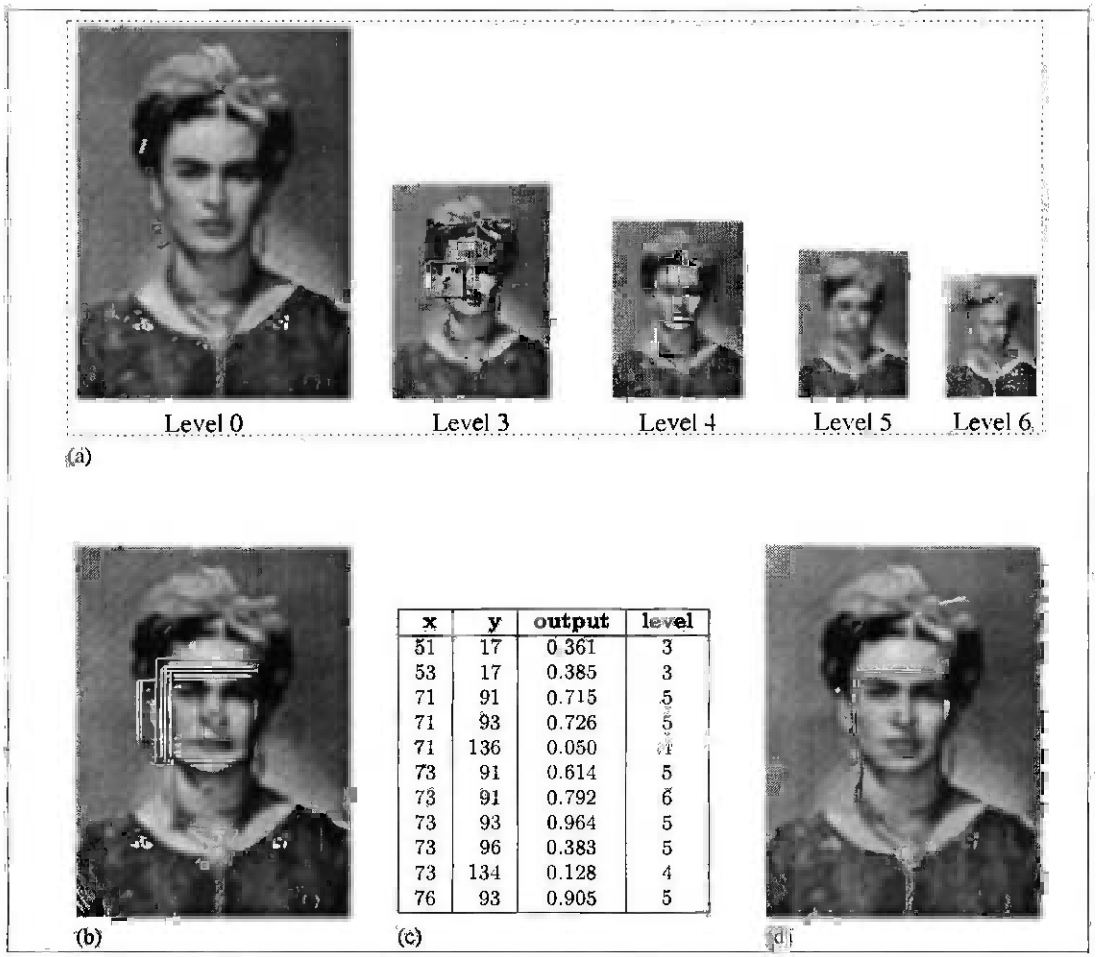
After the search is finished it is normal to find that a true detection in the image comprises a set of neighbouring positions with small variations in the face size (level on the pyramid). This phenomenon is expected given the variation of the face scale and position of the faces that form the training set, and it is an expression of the

classifier generalisation capability. On the other hand, false detections tend to appear as isolated examples.

In order to merge redundant detections and to diminish the number of false detections a three stage elimination procedure was implemented. In the first stage those hypothetical detections with a recorded output value lower than a particular threshold are discarded. In the second stage, an iterative procedure is implemented to merge redundant detections and to eliminate overlapping. In this stage, the number of times a hypothetical detection is merged is recorded in a counter assigned to each detection. The last step consists of an elimination of the isolated detections. It is important to notice that the elimination procedure is done off-line, after the whole image has been scanned.

The first stage takes advantage of the correlation between the output of the classifier and the face-likeness of the analysed pattern, in order to eliminate spurious detections. The optimal detection threshold value (τ_o) varies from network to network, and it has to be determined empirically for each of the trained networks.

In the second stage every pair of detections is tested for redundancy and overlap. Two detections are considered to be redundant if they are not more than four pyramid levels away from each other, and if the Euclidean distance between them is smaller than a predefined threshold τ_d; the threshold value was defined for the comparison of detections on the lowest level of the image pyramid, and it is scaled up to compare detections occurring on a higher level ($\tau_d(n) = \tau_d(\frac{5}{6})^{-n}$), where n is the detection's pyramid level). In the case where two detections belong to different pyramid levels, the threshold τ_d corresponding to the higher level is used. If the two detections are found to be redundant the one with the lowest output value is eliminated, and the counter of the other one is incremented. This process is repeated until no more redundancies are found. The next step is to test those detections that overlap. It is considered that two detections overlap if the ratio between the area of the intersection of two squares centred on each detection with side lengths equal to $20(\frac{5}{6})^{-n}$ and the area of the smaller square is greater than a threshold τ_ω. If the two detections overlap the one with the lowest output and counter value is eliminated. This process is repeated until no more overlappings are found.



x	y	output	level
51	17	0.361	3
53	17	0.385	3
71	91	0.715	5
71	93	0.726	5
71	136	0.050	4
73	91	0.614	5
73	91	0.792	6
73	93	0.964	5
73	96	0.383	5
73	134	0.128	4
76	93	0.905	5

Figure 3.9: Example of the merging procedure: (a) Different detections at different levels after search procedure is finished. (b) Face detections marked back to level 0. (c) List of detected points. **(d)** A single detection after merging and elimination. At the end of the procedure the detections on levels 5 and 6 were merged, and the detections on levels 3 and 4 were eliminated, obtaining by this the final detection seen in (d).

The last step is to eliminate those detections whose associated counter has a value smaller that the threshold τ_y. An example of the merging and elimination procedure can be seen in figure 3.9.

3.6 Experimental Results

To measure the performance of the classifiers described in this section the test images used by Sung and Poggio (1994) and Rowley et al. (1995) were used. The Sung test set is composed of 23 images with 155 faces in them and the Rowley test set, which includes the Sung test set, is composed of 107 images with 352 faces in them. The images were

originally obtained by the authors from the Internet, television broadcasts and scanned photographs. The whole set is available from a web site at Carnegie Mellon University (see appendix B).

An exhaustive search for faces was applied to each image, using the different classifiers. The CNN classifier was tested using the whole set; a total of 83,620,076 windows were analysed for each face view. The DFFS-DIFS approach was tested only with the Sung test set, analysing 9,665,939 different windows.

The evaluation is automatically performed using a similar procedure to the one used to merge two different detections; the list of detections produced by each network is compared with a list of the face positions in the test images (the test list). A real detection occurs when it can be merged with a "detection" on the test list; those detections that can not be matched against the faces listed in the test list are classified as false detections. The faces in the test list were manually classified as being front, right side or left side views; this information is used to measure the sensitivity of each kind of classifier, to front or side view faces.

In the case of the CNN classifiers, three networks were trained for each view, each of them using a different training set; the same face patterns were used in each case, while the corresponding non-face patterns were independently extracted from a different set of images without faces. Each network was trained a variable number of times (between 10 and 12), and the two best trained examples from each view were selected, using the generalisation capability of each network as the main criterion.

The main difference between the CNN approach described in this chapter and the one presented by Rowley et al. (1998) is that they use retinal connections without weight-sharing while my networks do share weights. For comparison reasons the training procedure of the previous paragraph was repeated on a non weight-sharing network using the same training sets; the architecture of this network is identical to the one described in the section 3.3, the main difference is the number of free parameters to train was increased from 825 in the original network to 7,737 in the non-weight sharing network. In the case of the DFFS-DIFS classifier the only difference from the Sung and Poggio approach is that the training set used was not generated with the classifier

CNN Classifier								
		Detected faces			Detection rate			False alarms rate
Front View	Network F1	457			90.14%			1/4483
		419	26	12	92.09%	78.79%	63.16%	(18654)
	Network F2	437			86.19%			1/4907
		400	24	13	87 91%	72.73%	68.42%	(17042)
Side View	Network S1	419			82.64%			1/2777
		376	26	17	82.64%	78.79%	89.47%	(60216)
	Network S2	393			77.51%			1/3088
		349	28	16	76.70%	84.85%	84.21%	(54164)
Front View	Network F3	428			84.42%			1/828
		392	24	12	86.15%	72.73%	63.16%	(100965)
	Network F4	465			91.72%			1/743
		421	30	14	92.53%	90.91%	73.68%	(112604)
Side View	Network S3	291			57.40%			1/10105
		253	24	14	55.60%	72.73%	73.68%	(16551)
	Network S4	315			62.13%			1/6820
		281	20	14	61.76%	60.61%	73.68%	(24523)
Front View	Network FM1	449			88.56%			1/8064
		412	25	12	90.55 %	75.76 %	63.15%	(10369)
Side View	Network SM1	256			50.49%			1/31710
		229	14	13	50.32%	42.42%	68.42%	(5274)
Front View	Network 1 Rowley et al. (1998)	462			91.12%			1/87935
		—	—	—	—	—	—	(945)

Table 3.3: Non-merged results: In the first column (*Detected faces*), the number in the top cell is the number of detected faces (out of 507); the three numbers in the lower cells are the number of front, left and right side views of faces in the test images that were detected (out of 455, 33 and 19 respectively). In the second column the detection rate is given. The third column (*False alarms rate*), gives at the top the ratio of false alarms per analysed window, and at the bottom the total number of false alarms. The networks F1, F2, S1 and S2 share weights, the networks F3, F4, S3 and S4 do not share weights, and the networks FM1 and SM1 are the mixture of the networks F1 and F2, and S1 and S2 respectively. For comparison the results obtained with a similar network by Rowley et al. (1998) are given in the bottom row.

itself, but instead the one generated with the CNN classifier was used.

Also the architecture of the CNN described in this chapter differs from the one proposed by Rowley et al. (1998) in the shape of the "receptive fields" used, that define how the units in the first set of hidden layers are connected to the input layer: while the receptive fields networks described here use overlapping vertical and horizontal bars and squares, Rowley et al. use overlapping horizontal bars, and non-overlapping squares in two sizes.

CNN Classifier								
		Detected faces			Detection rate			False alarms rate
Front View	Network F1	422			83.23%			1/15009
	$\tau_o = 0.0$, $\tau_v = 0.0$	393	18	11	86.37%	54.54%	57.89%	(5571)
	Network F2	403			79.49%			1/17288
	$\tau_o = 0.0$, $\tau_v = 0.0$	376	17	10	82.63%	51.51%	52.63%	(4837)
Side View	Network S1	195			38.46%			1/15119
	$\tau_o = 0.0$, $\tau_v = 0.0$	171	18	6	37.58%	54.55%	31.57%	(11061)
	Network S2	163			32.14%			1/16632
	$\tau_o = 0.0$, $\tau_v = 0.0$	144	14	5	31.64%	42.42%	26.32%	(10055)
Front View	Network F3	343			67.65%			1/8367
	$\tau_o = 0.0$, $\tau_v = 0.0$	322	15	6	70.76%	45.45%	31.57%	(9993)
	Network F4	373			73.57%			1/8458
	$\tau_o = 0.0$, $\tau_v = 0.0$	344	19	10	75.60%	57.57%	52.63%	(9887)
Side View	Network S3	133			26.23%			1/30166
	$\tau_o = 0.0$, $\tau_v = 0.0$	109	16	8	23.96%	48.48%	42.11%	(5544)
	Network S4	148			32.52%			1/22903
	$\tau_o = 0.0$, $\tau_v = 0.0$	130	11	7	28.57%	33.33%	36.84%	(7302)
Front View	Networks FM1	416			82 05%			1/21884
	$\tau_o = 0.0$, $\tau_v = 0.0$	387	19	10	85.05 %	57.58%	52.63%	(3821)
Side View	Networks SM1	135			26.63%			1/65869
	$\tau_o = 0.0$, $\tau_v = 0.0$	114	11	10	25.05%	33.33%	52.63%	(2539)
Front View	Network 1 Rowley et al. (1998)	458			90.33%			1/188861 (440)
		—	—	—	—	—	—	

Table 3.4: Merged results: The organisation of this table is analogous to table 3.3. Below each network label (column 2), the merging thresholding values τ_o and τ_v used are shown.

For the CNN classifiers, two main experiments were tried: in the first one each network was tested on its own, and in the second one the outputs of two weight sharing classifiers were combined with a small neural network (3 hidden units, one output unit and hyperbolic tangent activation function). The results without merging are shown in table 3.3, the results of the same networks after merging are shown in table 3.4. For comparison, at the bottom of each table the best results obtained by Rowley et al. (1998) are also shown. Their classifier is similar to the one reported here: it has 2 sets of hidden units and 2,905 connections, with no weight sharing.

In table 3.5 the results obtained by our implementation of the DFFS-DIFS classifier are shown. The classifier was tested only with the Sung and Poggio test set. At the bottom of the table, the results obtained by Sung and Poggio (1998) are presented

for comparison. For all the merged results shown, the thresholding values used in the merging procedure are $\tau_o = 0.0$ and $\tau_v = 0.0$ which implies that only overlapping windows will be merged and there will be no elimination of non-merged false detections.

DFFS-DIFS Classifier								
		Detected faces			Detection rate			False alarms rate
Front View	Not Merged	134			86.45 %			1/624
		125	5	4	89.29%	62.5%	57.14%	(15501)
	Merged	118			76.13 %			1/4380
	$\tau_o = 0.0$, $\tau_v = 0.0$	114	3	1	81.42 %	37.5 %	14.28 %	(2207)
Side view	Not Merged	139			89.68 %			1/57
		124	8	7	88.57%	100%	100%	(339913)
Front View	Multilayer Classifier Sung and Poggio (1998)	119			79.87%			1/1948922
		—	—	—	—	—	—	(5)

Table 3.5: Merged and non-merged results for the DFFS-DIFS classifier. The organisation of this table is analogous to table 3.3, the main difference being that this classifier was tested only with the Sung&Poggio test set which contains a total of 155 faces, where 140 of them are classified as front view faces, 8 of them as left side view faces and 7 of them as right side view faces. For comparison, the results obtained with a similar network by Sung and Poggio (1998) are given in the bottom row. The merged results of the side view classifier are not shown, since no meaningful results can be obtained given the poor performance of the classifier (the number of false detections is excessive).

As can be seen in tables 3.3 and 3.5, the detection rates of the different classifiers are similar to the results obtained by Rowley et al. and Sung and Poggio, while their false detection rate is much higher, especially in the case of the DFFS-DIFS classifier and front view non-weight-sharing networks.

In tables 3.4 and 3.5, where the results of the classifiers after merging are shown, it can be seen that the weight-sharing front view classifiers produce the best results; the reduction in the face detection rate is relatively small ($\simeq 7\%$) compared with the other classifiers, while the false detection rate is importantly reduced. In the case of the side view classifiers, there is an abrupt reduction of the detection rate after merging, leaving a very low detection rate. This can be explained by the high false alarm rate obtained on the side view classifiers, and a low quality of the detection, which mean first, that some of the detections reported on the non-merged results were product more of chance than an actual correct classification, *e.g.* the average false alarm rate for the non-merged networks S1 and S2 is 1/2932, a large number since that would

imply, if we consider the false alarms to be distributed uniformly on the image, a false alarm in an image area of 54 px^2.

One of the main advantages of sharing weights is the reduction of free parameters, and the improvement of the generalisation capabilities of the network, and given that the CNN networks perform much better than the non-weight-sharing networks the results suggest that the generalisation was the cause of the poor performance. As mentioned in section 3.2, 17,181 front view face patterns and 9,680 side face patterns were generated from 614 and 242 portraits respectively, while the face database used by Rowley et al. contains 15,750 face patterns which were generated from 1,050 portraits. For this reason I decided to enlarge the face databases: 2,338 and 423 new face portraits were added to the front and side view portrait sets respectively; 17,712 front view patterns and 17,730 side view patterns were generated from there. The non-face patterns were generated again using the bootstrap procedure; six different non-face sets were generated, containing 12,932, 12,216 and 12,615 patterns respectively for the front view cases and 19,171, 22,815 and 21,938 patterns for the side view cases.

Using this new training set the different classifiers were trained again and tested using the same procedure used with the first training set. The results are shown in tables 3.6, 3.7 and 3.8.

3.7 Discussion

In this section the results presented in the previous section will be discussed. The issues to be discussed are: the comparison of performance between the weight sharing and the non-weight-sharing networks, and the general performance of the classifier compared with the performance of the results provided by Rowley et al. (1998) and Sung and Poggio (1998).

Since different classifiers were used for front and side view faces, their performance is analysed first independently of each other, and then the general performance will be considered.

CNN Classifier (Extended Training Set)								
		Detected faces			Detection rate			False alarms rate
Front View	Network NF1	390			76.92%			1/17975
		363	16	11	79.78%	48.48%	57.89%	(4652)
	Network NF2	370			72.98%			1/17527
		348	13	9	76.48%	39.39%	47.37%	(4771)
Side View	Network NS1	284			56.02%			1/4612
		251	20	13	55.16%	60.61%	68.42%	(36263)
	Network NS2	387			76.33%			1/2334
		344	27	16	75.60%	81.81%	84.21%	(71665)
Front View	Network NF3	377			74.36%			1/22352
		358	13	6	78.68%	39.39%	31.57%	(3741)
	Network NF4	392			77.32%			1/19017
		364	20	8	80.00%	60.60%	42.10%	(4397)
Side View	Network NS3	315			62.13%			1/6820
		281	20	14	61.76%	60.61%	73.68%	(24523)
	Network NS4	215			42 40%			1/10105
		253	24	14	55.6%	75.76%	73.68%	(16551)
Front View	Network NFM1	350			69.03%			1/88207
		329	13	8	72.31 %	39.39%	42.11%	(948)
Side View	Network NSM1	201			39.65%			1/22982
		174	17	10	38.24%	51.51%	52.63%	(7277)
Front View	Network 1 Rowley et al. (1998)	462			91.12%			1/87935
		—	—	—	—	—	—	(945)

Table 3.6: Non-merged results of CNN classifiers trained with extended training set: In the first column (*Detected faces*), the number in the top cell is the number of detected faces (out of 507); the three numbers in the lower cells are the number of front, left and right side view faces in the test images that were detected (out of 455, 33 and 19 respectively). In the second column the detection rate is given. The third column (*False alarm rate*), gives at the top the ratio of false alarms per analysed window, and at the bottom the total number of false alarms. The networks NF1, NF2, NS1 and NS2 share weights, the networks NF3, NF4, NS3 and NS4 do not share weights, and the networks NFM1 and NSM1 are the mixture of the networks NF1 and NF2, and NS1 and NS2 respectively. For comparison the results obtained with a similar network by Rowley et al. (1998) are given in the bottom row.

3.7.1 Front View Faces

The detection rate in tables 3.3-3.8 is given as a percentage of the total number of faces in the test set, and with respect to the number of front, left-side and right-side faces. As can be seen, the absolute detection rate is very similar to the front view detection rate, since front view faces constitutes a majority of the test set (455 out of 507). For

CNN Classifier (Extended Training Set)								
		Detected faces			Detection rate			False alarms rate
Front View	Network NF1	368			72.58%			1/37380
	$\tau_o = 0.0$, $\tau_v = 0.0$	346	14	8	76.04%	42.42%	42.10%	(2237)
	Network NF2	352			69.42%			1/62217
	$\tau_o = 0.0$, $\tau_v = 0.0$	333	11	8	73.18%	33.33%	42.10%	(1344)
Side View	Network NS1	127			25.05%			1/16731
	$\tau_o = 0.0$, $\tau_v = 0.0$	108	10	9	23.74%	30.30%	47.37%	(9996)
	Network NS2	155			30.57%			1/10863
	$\tau_o = 0.0$, $\tau_v = 0.0$	141	8	6	31%	24.24%	31.58%	(15395)
Front View	Network NF3	356			70.22%			1/44836
	$\tau_o = 0.0$, $\tau_v = 0.0$	340	12	4	74.72%	36.36%	21.05%	(1865)
	Network NF4	381			75.14%			1/35090
	$\tau_o = 0.0$, $\tau_v = 0.0$	354	19	8	77.80%	57.57%	42.10%	(1359)
Side View	Network NS3	148			29.19%			1/22903
	$\tau_o = 0.0$, $\tau_v = 0.0$	130	11	7	28.57%	33.33%	36.84%	(7302)
	Network NS4	133			26.23%			1/30166
	$\tau_o = 0.0$, $\tau_v = 0.0$	109	16	8	23.95%	48.48%	42.10%	(5544)
Front View	Network NFM1	338			66.67 %			1/158973
	$\tau_o = 0.0$, $\tau_v = 0.0$	320	11	7	70.33 %	33.33 %	36.84%	(526)
Side View	Network NSM1	92			%18.14			1/52279
	$\tau_o = 0.0$, $\tau_v = 0.0$	74	10	8	16.26%	30.30%	42.11%	(3199)
Front View	Network 1 Rowley et al. (1998)	458			90.33%			1/188861
		—	—	—	—	—	—	(440)

Table 3.7: Merged results of classifiers trained with extended training set: The organisation of this table is analogous to the table 3.6 with the difference that below each network label (column 2), the merging thresholding values τ_o and τ_v used are shown.

this reason it was considered appropriate to refer to the absolute detection rate in the following argument in this subsection.

In the case of front view classifiers, the results obtained with the CNN classifiers trained with the first data set show that the performance of the weight-sharing networks (F1 and F2) is much better than the non-weight-sharing networks (F3 and F4). Before merging, the detection rate of the sharing and non-weight sharing classifiers is similar, while the number of false detections for the non-weight-sharing classifiers is approximately six times higher than the weight-sharing classifiers. After merging, the number of false detections is considerably reduced, while the detection rate is reduced by almost 7% for the weight-sharing classifiers and 18% for the non-weight-sharing

DFFS-DIFS Classifier (Extended Training Set)								
		Detected faces			Detection rate			False alarms rate
Front View	Not Merged	131			84.51%			1/9561
		123	4	4	87.86%	50.00%	57.14%	(1011)
	Merged	127			81.93%			1/29650
	$\tau_o = 0.0$, $\tau_v = 0.0$	119	4	4	76.77%	50.00%	57.14%	(326)
Side View	Not Merged	109			70.32%			1/160
		97	7	5	69.29%	87.50%	71.42%	(120585)
	Merged	45			29.03%			1/5184
	$\tau_o = 0.0$, $\tau_v = 0.0$	41	3	1	29.28%	37.5%	14.28%	(3729)
Front View	Multilayer Classifier Sung and Poggio (1998)	119			79.87%			1/1948922
		—	—	—	—	—	—	(5)

Table 3.8: Merged and non-merged results of the DFFS-DIFS classifier trained with extended training set. The organisation of this table is analogous to the table 3.6, with the difference that this classifier was tested only with the Sung&Poggio test set which contains a total of 155 faces, where 140 of them were classified as front view faces, 8 of them as left side view faces and 7 of them as right side view faces. For comparison the results obtained with a similar network by Sung and Poggio (1998) are given in the bottom row.

classifiers.

In the case of the extended training set for the front view classifiers, the difference in performance between the two different types of CNN classifiers tested is small, the non weight-sharing networks (NF3 and NF4) having a better performance, especially in terms of the number of false detections; also, the detection rate reduction caused by the merging procedure is on average only 3.95% for the weight-sharing networks (NF1 and NF2) and 3.16% for the non-weight-sharing networks.

By comparing the merged results from the classifiers trained with the original and extended training sets, it is clear that there is an overall improvement, especially in terms of an important diminution of the false detection rate. The improvement is more dramatic in the case of the non weight-sharing classifiers, where the average false detection rate of the classifiers NF3 and NF4 is almost a fifth of that obtained from the classifiers F3 and F4, while the detection rate slightly improved. In the case of the weight-sharing networks there is also an improvement in the results where the average false detection rate of the networks NF1 and NF2 is a third of that obtained with the networks F1 and F2, although the general improvement is damped by a 10%

diminution of the face detection rate.

In the case of the Mixture of Networks classifier, in both cases (the networks trained with the original and extended data set), the performance improves especially in terms of diminution of the false detection rate, while suffering a small diminution of the detection rate.

In general, this justifies the hypothesis that the architecture proposed by Rowley et al. can be improved by using the CNN paradigm, adding weight sharing to the retinal connections used by the cited authors; which allows us to obtain acceptable results with a relatively small data set.

The results obtained by the DFFS-DIFS classifier are helpful as they allow us to compare the two technologies directly, as the same training set was used for each.

The DFFS-DIFS classifier trained with the original front-view training set has a reasonable performance in terms of the detection rate, while performing poorly in terms of the false alarm rate. A difference in performance of 10% between the non-merged and merged results was obtained, while the number of false face detections decrease by a factor of 7.

With the extended training set, much better results were obtained, with a very small difference of the detection rate obtained between the non-merged and merged classifiers, and a considerable reduction on the number of false face detections (3 times less).

In general the difference in performance between the CNN classifiers and the DFFS-DIFS classifiers is similar in terms of number of detected faces, while the performance of the CNN classifiers is much better particularly with the results obtained with the original training set. As well as the non-weight-sharing CNN networks, the DFFS-DIFS classifier needs a larger training set in order to obtain reasonable results. This is understandable, since the method models a 321-dimensional face with 12 Gaussian clusters, with the implication that the number of training patterns and the variance of the training set should be large enough to allow the correct definition of each cluster.

3.7.2 The side view classifiers

In general terms, the performance of the side view classifiers was considered to be inadequate. Before continuing, it is important to note that since the number of faces in the test set classified as being either being right-side or left-side face represents only slightly more than 10% of the total number of faces, it would be more appropriate to discuss the detection rate of the side-view classifiers using the view dependent results shown in tables 3.3-3.8 (the three bottom cells of the second column on each row in the mentioned tables).

With both training sets, a very large number of false detections are produced, deeming meaningless the results obtained after the merging procedure; in the best case, the difference in the detection rate produced by the merging procedure is 21% (Network NS1, right-side view), but on average is around 30 percent. In the case of the DFFS-DIFS classifier the results are unsatisfactory, producing a very large number of false detections.

It should be noted that the percentage of front and side view faces detected are similar, and that in terms of the absolute number of faces detected, the side view classifier performs better detecting front view faces than side view faces (but worse than any of the front view classifiers); after a visual inspection of the detections of front view faces by the side view classifier, I found that the side view classifier correctly classify slightly off centre front view faces This can be explained as a side effect of the more intrusive mask used for the side view classifiers; when presented with an off-centre front view face, the mask obscures the background on the side of the face, while the half of the front view face when masked resembles a side view face. An example of a figure showing front view faces detected by a side view classifier is shown in figure 3.10.

The poor performance obtained for the side view classifiers is understandable — the problem is much more complex than the classification of front view faces, since for small changes in face orientation the amount of change of the face configuration is larger for the side view patterns than that of the front view patterns, with the consequence that there is a larger variance on the training set, and therefore a larger training set is necessary to adequately represent the whole class. Also, the side view patterns have

only 234 significant elements, a small number compared with the 320 used for the front view patterns.

Figure 3.10: Detection of front view faces by the side view classifier (SMi).

3.7.3 The merging procedure

The merging procedure is not perfect; a certain number of correctly detected faces will be eliminated, *i.e.* when a correctly detected face overlaps with a false detection which has a larger output, or when the detection area of two real faces overlaps. For example, the images (**a**) and (**b**) in figure 3.11 show an example of the first case, where one of the four detections shown is correct but its corresponding output value is the smallest one; since it completely overlaps with the other three detections it is eliminated during the merging procedure. Since the merging procedure assumes a strong positive correlation between the classifier output and the *faceness* of the analysed pattern, the number of correctly detected faces that are eliminated depends on the classifier generalisation capabilities.

It was noted that in very even areas of the image the histogram equalisation within the preprocessing stage has the side effect of increasing the contrast of the pattern, which produces a pattern where there was none before. While it is useful to increase

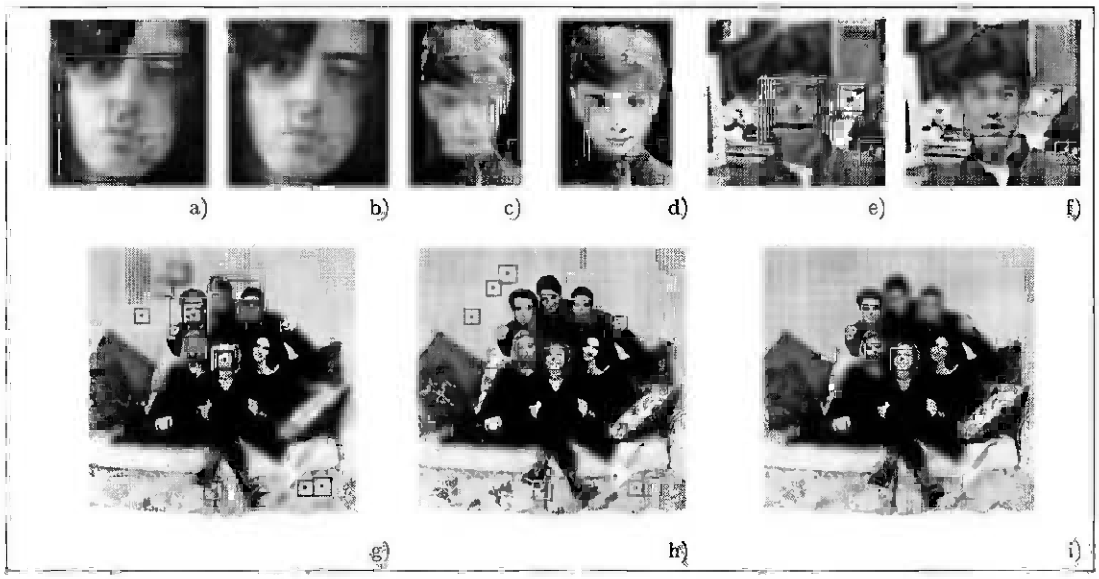


Figure 3.11: Face detection examples in images from the Rowley et al. (1998) test set, using the front view weight-sharing classifiers. The image pair (**a**,**b**) shows an example of how a correct detection is eliminated during the merging procedure. Image pairs (**c**,**d**), (**e**,**f**) and (**g**,**h**) are examples of detection before and after the merging procedure with standard merging thresholds ($\tau_o = 0.0$, $\tau_v = 0.0$), while image (**i**) shows the results obtained after applying the merging procedure to image (**g**) with merging thresholds $\tau_o = 0.0$, $\tau_v = 3.0$. The detections shown in the images (**c** - **i**) were obtained with the classifier FM1, while the images (**a**) and (**b**) show results obtained with the classifier F1.

the contrast of images with poor illumination, the spurious patterns thereby created can sometimes be confused by the classifier with a face. It would be advisable to use the variance of the pattern intensity as a measure of texture, which would indicate how likely it is to find a face in that area.

The image pairs (**c**,**d**), (**e**,**f**) and (**g**,**h**) in figure 3.11 show some examples of different images that compose the test set, before and after merging. The merging parameters τ_o and τ_v are both equal to zero, with the effect that none of the false detections will be eliminated, only multiple detections will be merged into single non-overlapping detections. Most of these false detections can be eliminated by choosing an appropriate set of parameters, *e.g.* the image (**i**) in the same figure shows how all the false detections are eliminated by using a particular set of thresholding values ($\tau_o = 0.0$, $\tau_v = 3.0$ in this case).

The optimal thresholding values are different for each classifier, given the different training conditions and data used for each of them. In order to observe how different

thresholding values affect the classifier's response, the receiver operative characteristic curve (ROC) was plotted for each classifier; the ROC curve plots the face detection rate *vs.* the number of false detections per analysed window for different thresholding values (see figures 3.12– 3.16).

The ideal point on a ROC curve is that which gives the best compromise between detection rate and false alarms and is normally selected as the inflection point of the curve. Since the ROC curves used to describe the behaviour of the classifiers depend on two thresholding values, we can define the ideal point as the intersection of the projection of the curve onto the planes ZY and XY.

With the ROC curve we can also obtain a qualitative insight into the classifier performance, since for a classifier which does not perform well, the merging procedure — where the thresholding values are applied — will behave erratically, producing a rough curve when evaluated. For example, the side view classifiers tend to have a much rougher ROC curve as can be seen on the plot c in figure 3.16, or plot **b** in figure 3.12.

3.8 Conclusions

In this chapter the details for the implementation of face detectors are given. A variation on the approach of Rowley et al. (1998) is proposed, which differs from the original approach in that weight sharing was used in a neural network with retinal connections, obtaining a fully convolutional neural network. It was proved that better generalisation is obtained, with the implication that a smaller training set is needed in order to achieve similar results.

In order to detect a larger range of face orientations two different training sets were tried, one composed with only front view face patterns and another with side view face patterns. While good results were obtained with front view classifier, the detection of side view faces proved to be a harder problem; the method should be refined in order to obtain better results, by diminishing the variance of the side view training set by constraining the range of facial orientations contained in the class, thus dividing the problem into a larger set of viewer-centered representations of the face object.

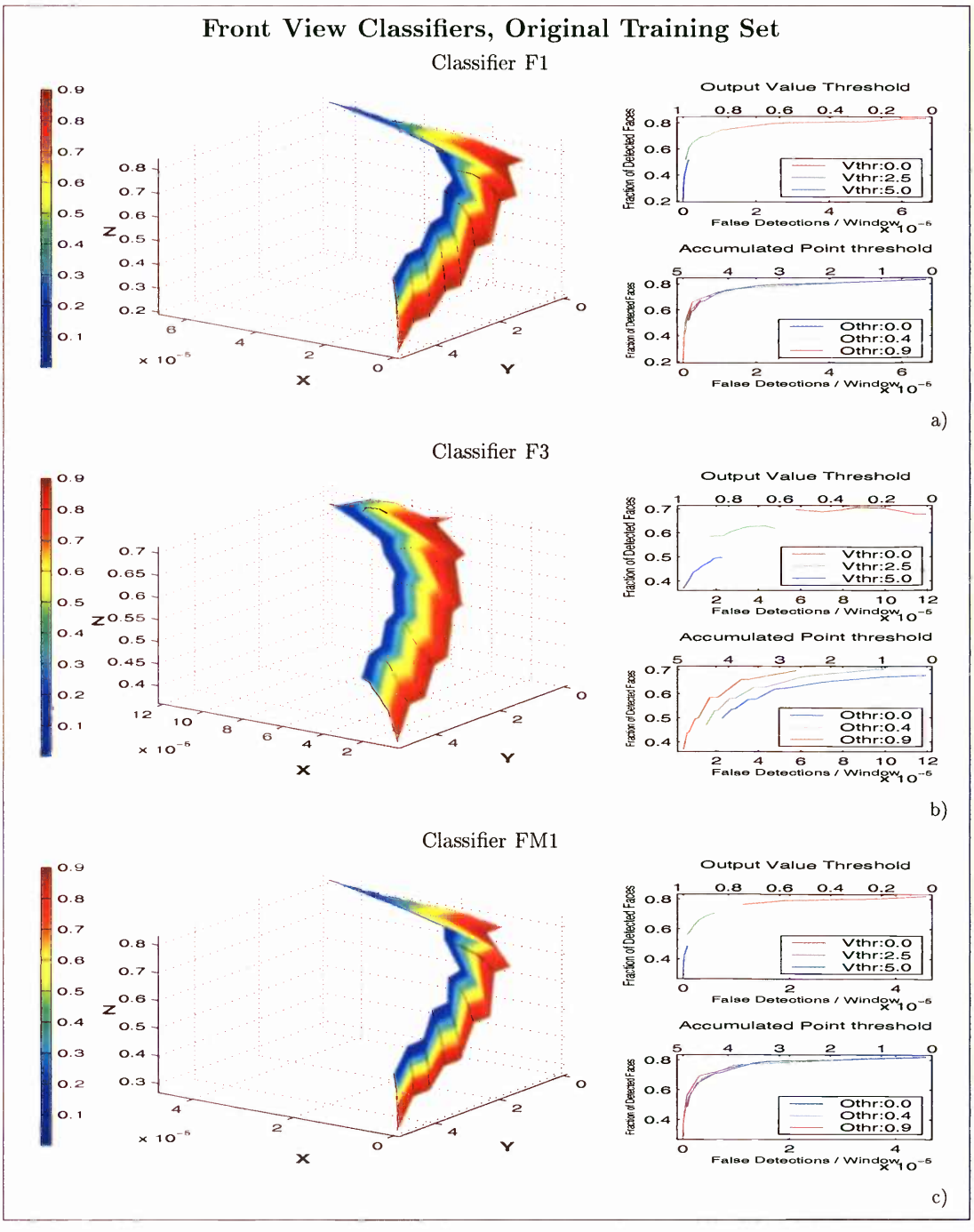


Figure 3.12: ROC curve of front view classifiers (original training set). In each figure the response of the classifier to variation of the merger parameters is shown. In the figure on the left, the **X** axis corresponds to the number of false detections per analysed window, the **Y** axis corresponds to the value of the *accumulated point threshold* τ_v used which varies from 0 to 5 in 0.5 intervals. The **Z** axis corresponds to the fraction of detected faces. The colour of the graph indicates the value used by the output threshold τ_o at each point. The graphs on the right hand side show the projection of the graph on the XZ plane: in the top-right graph the projection of the curve for three different values of τ_v while varying τ_o is shown, while the bottom-right graph shows the projection of the curve for three different values of τ_o while varying τ_v.

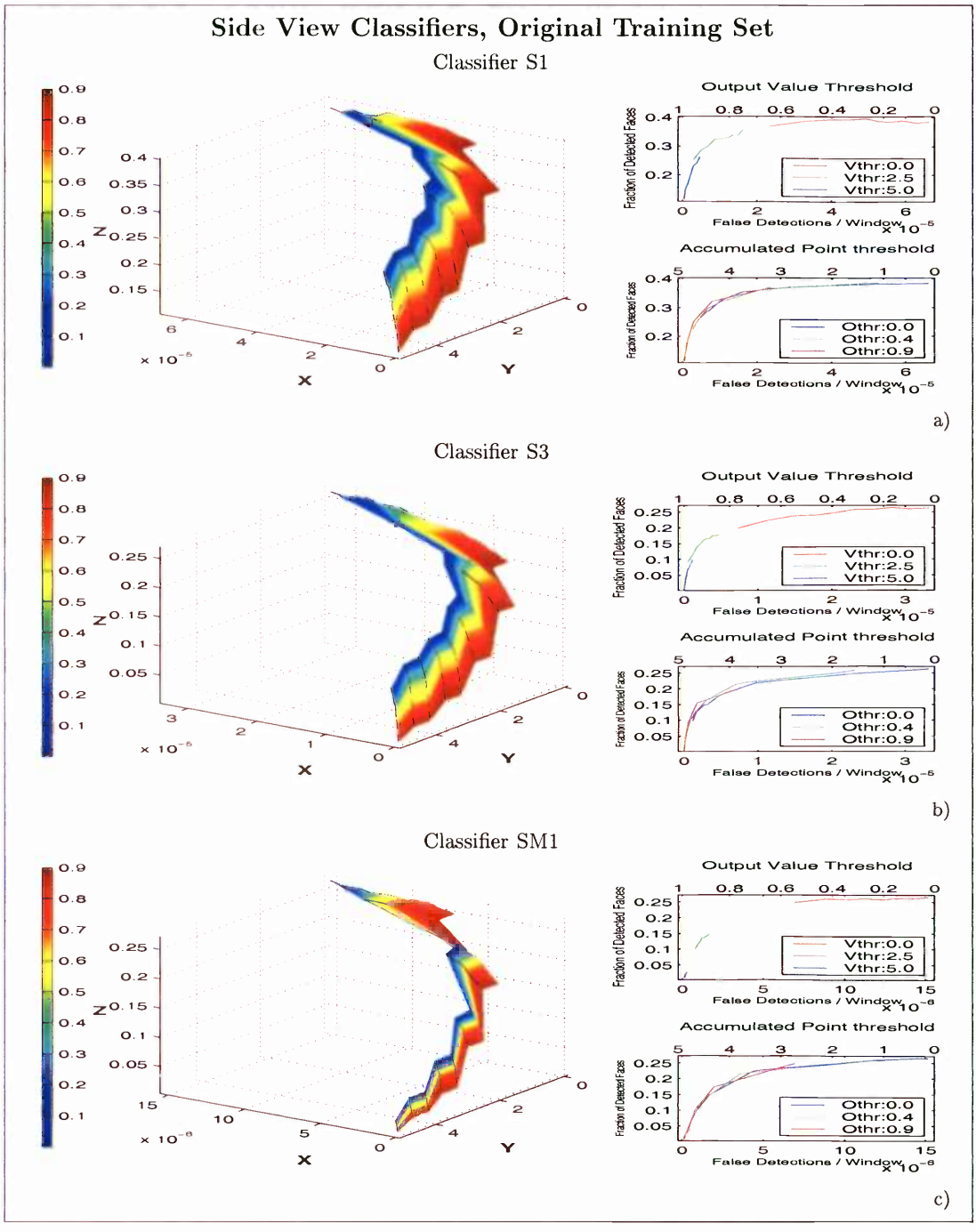


Figure 3.13: ROC curve of side view classifiers (original training set). In each figure the response of the classifier to variation of the merger parameters is shown. In the figure on the left, the **X** axis corresponds to the number of false detections per analysed window, the **Y** axis corresponds to the value of the *accumulated point threshold* τ_v used which varies from 0 to 5 in 0.5 intervals. The **Z** axis corresponds to the fraction of detected faces. The colour of the graph indicates the value used by the output threshold τ_o at each point. The graphs on the right hand side show the projection of the graph on the XZ plane: in the top-right graph the projection of the curve for three different values of τ_v while varying τ_o is shown, while the bottom-right graph shows the projection of the curve for three different values of τ_o while varying τ_v.

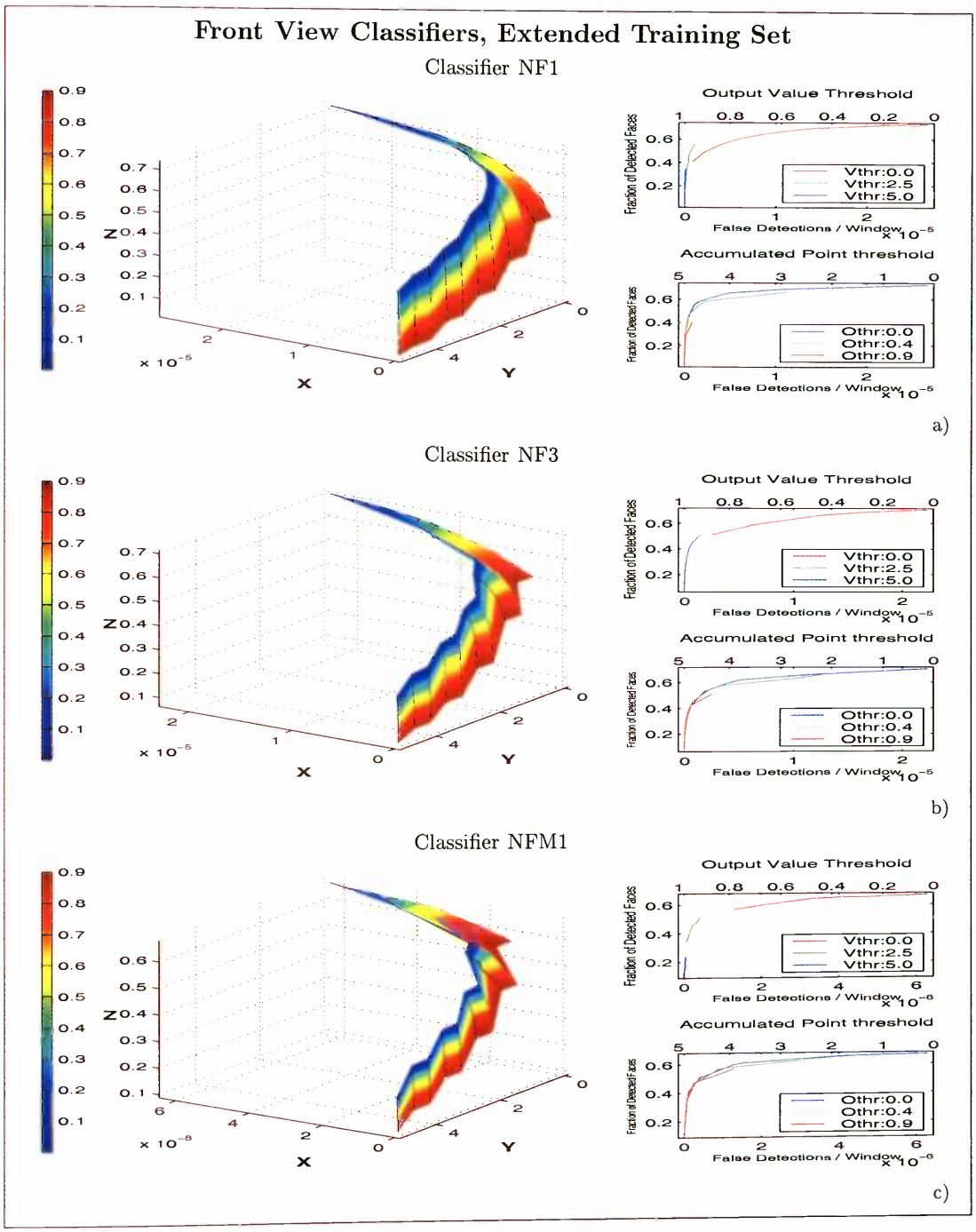


Figure 3.14: ROC curve of front view classifiers (extended training set). In each figure the response of the classifier to variation of the merger parameters is shown. In the figure on the left, the **X** axis corresponds to the number of false detections per analysed window, the **Y** axis corresponds to the value of the *accumulated point threshold* τ_v used which varies from 0 to 5 in 0.5 intervals. The **Z** axis corresponds to the fraction of detected faces. The colour of the graph indicates the value used by the output threshold τ_o at each point. The graphs on the right hand side show the projection of the graph on the XZ plane: in the top-right graph the projection of the curve for three different values of τ_v while varying τ_o is shown, while the bottom-right graph shows the projection of the curve for three different values of τ_o while varying τ_v.

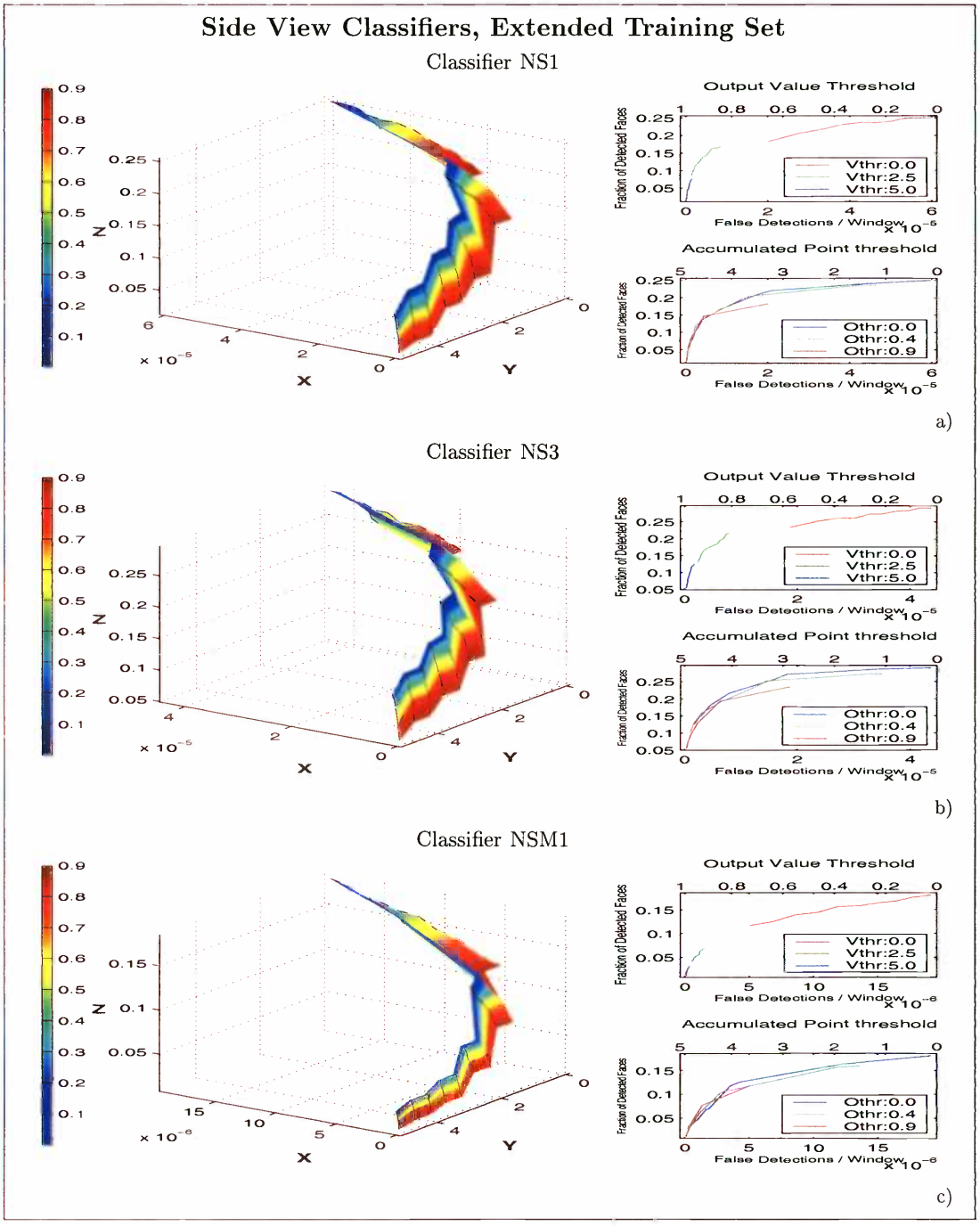


Figure 3.15: ROC curve of side view classifiers (extended training set). In each figure the response of the classifier to variation of the merger parameters is shown. In the figure on the left, the **X** axis corresponds to the number of false detections per analysed window, the **Y** axis corresponds to the value of the *accumulated point threshold* τ_v used which varies from 0 to 5 in 0.5 intervals. The **Z** axis corresponds to the fraction of detected faces. The colour of the graph indicates the value used by the output threshold τ_o at each point.The graphs on the right hand side show the projection of the graph on the XZ plane: in the top-right graph the projection of the curve for three different values of τ_v while varying τ_o is shown, while the bottom-right graph shows the projection of the curve for three different values of τ_o while varying τ_v.

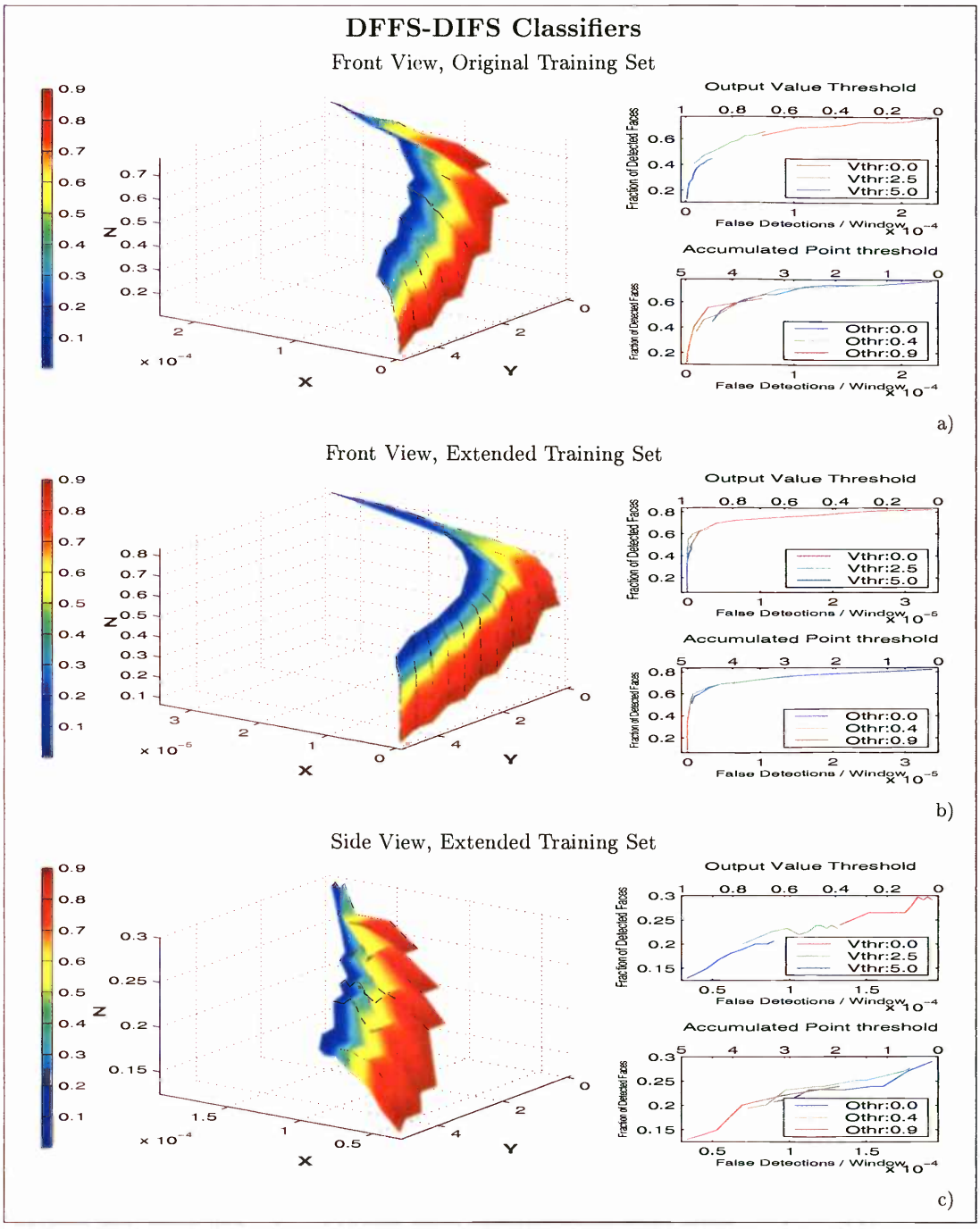


Figure 3.16: ROC curve of DFFS-DIFS classifiers (original and extended training set). In each figure the response of the classifier to variation of the merger parameters is shown. In the figure on the left, the **X** axis corresponds to the number of false detections per analysed window, the **Y** axis corresponds to the value of the *accumulated point threshold* τ_v used which varies from 0 to 5 in 0.5 intervals. The **Z** axis corresponds to the fraction of detected faces. The colour of the graph indicates the value used by the output threshold τ_o at each point. The graphs on the right hand side show the projection of the graph on the XZ plane: in the top-right graph the projection of the curve for three different values of τ_v while varying τ_o is shown, while the bottom-right graph shows the projection of the curve for three different values of τ_o while varying τ_v.

Chapter 4

Depth Estimation

4.1 Introduction

The depth in the scene is the main source of constraints for the simplification of the search for faces in a scene; by estimating a depth map of the scene, a rough description of the 3-D structure of the world around the robot can be recovered, which helps to define not only where one is likely to find a face but also which is the range of relative face sizes that might be found in the scene.

In this chapter a depth estimation procedure is proposed. The method estimates the depth from a sequence of images obtained from a camera that is moving in a controlled fashion. The procedure is close in spirit to the work of Matthies et al. (1989), while adding some features to make it suitable for our necessities: it uses a hierarchical multilevel algorithm for combining different depth measurements, which allows us to obtain rough but fast depth estimation, increasing the quality of the measurements as the times goes by. The procedure is tested with a set of artificial and real images.

The chapter is organised as follows: in the first section a review of depth estimation techniques is given, while the second section describes the theoretical framework for the proposed depth estimation procedure. In the third and fourth sections the method used to estimate the disparity between different frames is explained, together with an uncertainty analysis of the depth estimation procedure. The method used to mix different disparity maps is described in the fifth section. Section six deals with the practical aspects related to the physical implementation of the algorithm and finally,

section seven describes the set of experiments used to test the proposed method, and presents the results obtained.

4.2 A small review of depth estimation techniques

Automatic depth perception is a field of computational vision which attempts to recover the 3D structure of a scene from a set of correlated images. The techniques described in this review are called "passive" methods, in the sense that they rely on external energy sources in order to extract information from the environment. On the other hand there are the so-called "active", or "range finding" methods, which emit a controlled beam of energy and then, by measuring its reflection or time-of-flight, are able to obtain 3-D information from the environment[1]. A general review of the different methods used for depth estimation can be found in (Jarvis, 1983).

The study of how biological systems perceive the world through their eyes (visual perception) has been the main source of inspiration for the development of algorithms for automatic depth perception, therefore it is necessary to describe first the biological cues from which depth can be derived by the optical system.

There are three main biological phenomena that have been used as a basis for the development of automatic depth estimation systems. The first two are vergence and accommodation, known as "physiological" clues — first suggested by Berkeley (1709) — and the third is binocular stereopsis.

- Vergence: When both eyes fixate a particular object in the scene, the eyes swivel in order to keep the object centred on the retina; the eyes turn inwards while fixating a close object and are nearly parallel when they are fixating on the horizon.
- Accommodation: Change in shape of, or movement of, the lens in the imaging system is used to bring an object in the scene to focus.

[1] The term active vision not only refers to the sensing technology, as in this case, but also to the paradigm where the observer has an active role in the vision process, where by a direct interaction with the world or by exploiting the structural characteristics of a sequence of images, the task in mind can be solved (see Aloimonos et al. (1988) and Blake and Yuille (1992)).

- Binocular stereopsis (Stereo Disparity): Animals with partially overlapping visual fields (with the two eyes situated in the same plane) perceive depth by comparing the images projected onto the retina of each eye which, given the interocular separation, observe the same scene from slightly different positions. When observing a particular point of the scene, the eyes fixate on it, by swiveling in such way that the point is projected at the centre of each eye's retina; the other points in the scene (observed points) are projected at different positions on the retina of each eye and the difference between those positions is called binocular disparity. From the binocular disparity we can estimate the distance between the fixation and observed point, also called "relative depth". The absolute depth –the distance between the observer and the observed point – can be recovered if the distance between the observer and the fixation point is known (which can be recovered by triangulation if we know the vergence angle and the interocular distance). The disparity is roughly proportional to the relative depth, and inversely proportional to the squared distance between the fixation point and the observer.

Most of the algorithms for depth estimation that have been proposed rely on the binocular disparity, although some are based on physiological cues to estimate the disparity. For example, Jarvis (1983) discusses the use of auto-focus in order to estimate the depth in the scene: the camera lens position that gives the best focus at a particular point in the scene is found, and then with the knowledge of the camera parameters (focal length, and focal plane to image plane distance), the depth at that position can be found. In a similar vein, Pentland (1987) uses the amount of defocus (blur) as the main cue to estimate depth: the focus in an optic system depends only on the distance between the observed surface and the lens system, so by measuring the gradient of defocusing and with the knowledge of the camera parameters, the depth in the scene can be recovered.

There are other cues for depth perception which require higher level processing (*e.g.* the recognition of features such as lines or shapes) and do not require binocular vision. These cues, called "pictorial" by Bruce et al. (1996), rely mainly on the perspective projection of the 3D scene on the retina. For example, the convergence of parallel lines

at the horizon gives an impression of depth; the vertical position of the projection on the retina of objects lying on the ground increases with distance (up to the vanishing point); an object closer to the observer might occlude another one which is far away.

4.2.1 Depth from motion

The approach for face detection described in this chapter relies on the motion of the observer to estimate the depth in the scene, for this reason, in the rest of this section a more detailed review of depth from motion techniques proposed is given.

A phenomenon that in principle is equivalent to binocular stereoscopy is motion parallax. Motion parallax is defined as the movement of the image of an object over the retina; the rate of movement is directly proportional to the velocity of the object relative to the eye, and inversely proportional to the distance between the object and the eye. While binocular stereoscopy relies on the spatial configuration of the two eyes in order to induce the disparity, motion parallax relies on the movement of the observer to induce a similar disparity. Binocular disparity and motion parallax represent spatial and temporal samplings of the same 3-D information (Bruce et al., 1996).

In either case, the first step to take in order to estimate depth is to solve the correspondence problem, which involves matching corresponding elements in a pair of images (Marr and Poggio, 1976; Ullman, 1979). The solution of the correspondence problem is expressed in terms of a disparity map for binocular stereopsis and with an optical flow field, where the time between frames is taken into account, for a moving observer.

Optical flow is defined as the vector field that describes the variation of brightness patterns in the retina produced by the relative movement between the observer and the environment. It is important to note that, although the optical flow can be obtained by solving the correspondence problem, it can also be obtained by other means: correlation based methods that match small windows between frames (*e.g.* Anandan, 1984), gradient intensity based methods, that measure the image velocity from spatio-temporal derivatives of the image intensities (*e.g.* Horn and Schunck, 1981; Lucas and Kanade, 1981), or frequency based methods, that use orientation sensitive

filters in the Fourier domain to estimate the image velocities, (*e.g.* Simoncelli and Adelson (1991) combine gradient- and frequency-based methods). A thorough survey of different techniques used to estimate the optical flow can be found in (Beauchemin and Barron, 1995).

The general problem of motion perception (or structure from motion) involves recovering the motion and the structure of the scene from a sequence of images. The approaches used for the solution of this problem can be classified as discrete-time or continuous-time methods: discrete-time methods are based on the relative position of features being tracked, while continuous-time methods rely on the image velocity (optical flow).

The first step in discrete-time methods is to find the correspondence between frames of a number of image features present in all the frames, and then use their positions to calculate the essential matrix, which encodes the rotation and translation of the camera between frames. After the essential matrix has been found, the translation rotation and depth can be recovered from it. A typical example is the eight-point approach proposed by Longuet-Higgins (1981). The main disadvantage of this method is that it is quite sensitive to noise, although some more robust algorithms have been proposed recently. A complete explanation of the algorithms used by this kind of method can be found in (Faugeras, 1993; Kanatani, 1993, chapters 7 and 6 respectively).

Continuous-time methods compute the observer motion from optical flow, which involves solving the equation that relates the movement of the observer, expressed as a rotation and translation velocity vector $[\boldsymbol{\Omega}, \mathbf{T}]$, and the optical flow[2]. One of the first examples of this kind of approach is given by Koenderink and van Doorn (1976) who show that patterns of relative motion within a small region of the flow field can be described as the sum of an expansion, a rotation and a deformation. These transformations can be used to estimate invariant properties of the motion and surface shape. Longuet-Higgins and Prazdny (1980) proposed a method for the determination of unrestricted motion of the observer (translation and rotation) with respect to planar surfaces from the optical flow. Heeger and Jepson (1992) split the optical flow equation into three sets of equations, which allows them to solve first for translation, and then

[2] The full description of this relationship is given in section 4.3.

for rotation and finally depth.

As was pointed out by Adiv (1985), the problem of motion perception is basically the same for the discrete and continuous methods, the latter being an approximation of the former under the following conditions:

- The ratio between the Z component of the observer's translational speed and environment's absolute depth is small ($T_Z/Z \ll 1$)[3].
- The rotation parameters are small.
- The camera field of view is not very large.

The solution of the general problem of motion perception is a difficult task, given the complex relationships between the optical flow and the unrestricted movement of the observer; the problem is greatly simplified if the motion of the observer is known. The first approaches for depth estimation which take advantage of using a predetermined translation movement utilise a forward motion, which produces an expansion of the image around the focus of expansion[4] (FOE); in those cases the velocity vectors that compose the optical flow point radially outwards from the FOE, while their magnitude is directly proportional to their distance from the FOE and inversely proportional to the depth. For example, Williams (1980) presents an algorithm for depth estimation using a forward-controlled camera movement and an explicit model of the scene, in terms of surfaces parallel to the image plane or the ground plane. The depth is estimated by minimising the difference between an artificial image generated by extrapolating the original scene image (given the scene surface model and the known camera translation) and the actual image obtained after the motion. In 1983, Buxton and Buxton proposed the use of the visual motion of edges in the scene in order to infer depth under forward motion. The edges are detected by computing spatio-temporal zero-crossings by using a generalisation of the Marr and Hildreth edge detector operator (1980). In the field of robot navigation, Perkins and Hayes (1994) implement a real-time depth estimator that works in a mobile robot as the basis of a vision-based navigation system. Under

[3] namelly, the inverse of the time to contact.

[4] The focus of expansion can be defined as the point where the translation vector, T, pierces the image plane (see appendix A).

the constraint that the robot moves in a trajectory parallel to the optical axis of the camera, they define a radial optical flow field. The optical flow is estimated by tracking a set of labelled edges as the robot moves. The main disadvantage of the forward motion is that the accuracy of depth estimates diminishes as the distance between the image features being tracked and the FOE diminishes, with the result that the depth in those areas close to the FOE can not be reliably estimated. On the other hand, if the observer moves sideways an even accuracy in the optical flow field is achieved, for example, Matthies et al. (1989) propose an on-line incremental method, that uses a Kalman filter to reduce the uncertainty of the measurements over time; the camera is translated along a trajectory parallel to the image plane in order to induce disparity from which the depth is recovered.

Certain approaches for depth estimation adopt the paradigm of "active vision" introduced by Aloimonos et al. (1988), in which a purposive strategy is used in order to constrain the solution of the problem: the observer is no longer a passive subject, instead it becomes an active agent that interacts with the world in order to simplify the solution of a particular task. For example, concerning the solution of the problem of structure from motion Aloimonos et al. (1988) shows that a closed form solution is possible if a binocular observer keeps tracking a point on a moving object (or while the observer is moving). Hayes (1989) implemented a real-time, biologically inspired depth estimation algorithm based in the work Ballard et al. (1988), where the relative depth is estimated from the spatio-temporal derivatives of a sequence of images grabbed while the camera is moving in a controlled manner, keeping the camera fixated on the centre of the scene. The absolute depth is recovered using the distance to the fixation point, which is estimated by triangulation since the vergence angle is known. The image spatio-temporal derivatives are calculated using a Datacube MaxVideo, producing a full frame depth map 12.5 times a second. Huang and Aloimonos (1991) use an active approach to compute the relative depth of a scene from the spatio-temporal derivatives, for different kinds of translations: along the optical axis, parallel to the scene, general translation (without rotation) and the general case (translation and rotation of the observer).

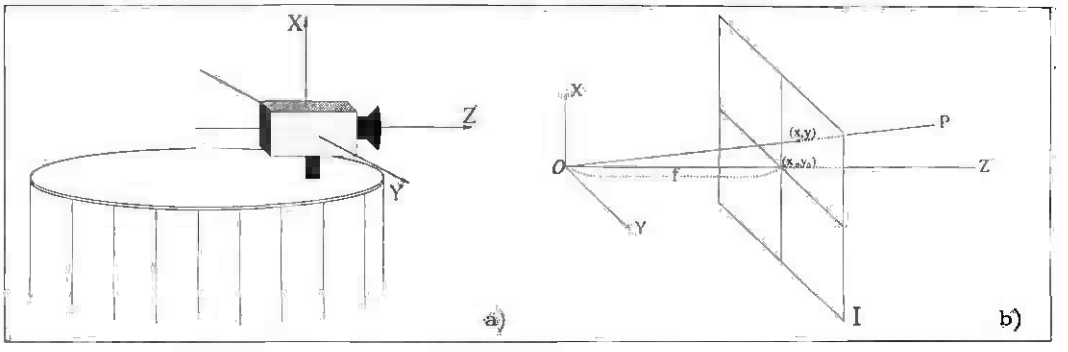


Figure 4.1: Figure(a) shows the camera mounted in the mobile robot, and how the coordinate frame is attached to it. Figure (b) shows the perspective projection of the scene into the image plane (the camera is focused at infinity).

4.3 Depth estimation

In this section the theoretical framework used for the depth estimation algorithm proposed in this chapter will be given. The relationship between the movement of the observer and the projection of the 3-D scene in the image plane is analysed, and then a description of the method used to estimate the disparity between frames is given.

4.3.1 Camera model

A pinhole camera model was used to establish the geometric relationship between the information in the 3-D scene in front of the camera and its projection in the 2D image plane. This relationship is expressed as

$$x = \frac{fX}{Z} + x_0, \quad y = \frac{fY}{Z} + y_0 \tag{4.1}$$

where (X, Y, Z) are the coordinates of a point P in the 3D space, (x, y) are the coordinates of the projection into the image plane, (x_0, y_0) are the coordinates of the intersection of the camera Z axis and image plane (optical centre)[5] and f is a conversion factor (analogous to the camera focal length) between meters in the real world and pixels in the image plane. A right-handed coordinate system centred at the camera viewpoint with the axis Z in the same direction as the camera optical axis was used (see figure 4.1).

[5] For simplicity in the rest of this chapter it will be assumed that the optical centre (x_0, y_0) coincides with the origin of the image plane.

The pinhole camera model is ideal; when using real cameras a calibration process is needed in order to make the measurements obtained with the camera compliant with the model and to eliminate possible optical distortions in the image. A full description of the calibration process is described in the appendix A.

4.3.2 Camera Motion and Depth

The projection into the image plane of a space point $\mathbf{P} = [X, Y, Z]^\top$ moving with a translational velocity $\mathbf{T} = [T_x, T_y, T_z]^\top$ and an angular velocity $\boldsymbol{\Omega} = [\Omega_x, \Omega_y, \Omega_z]^\top$, defined with respect to the camera axis frame, is given by (Longuet-Higgins and Prazdny, 1980):

$$\begin{bmatrix} u \\ v \end{bmatrix} = \frac{1}{Z} \begin{bmatrix} -f & 0 & X \\ 0 & -f & Y \end{bmatrix} \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix} + \begin{bmatrix} \frac{XY}{f} & -(1+\frac{X^2}{f}) & Y \\ (1+\frac{Y^2}{f}) & -\frac{XY}{f} & -X \end{bmatrix} \begin{bmatrix} \Omega_x \\ \Omega_y \\ \Omega_z \end{bmatrix} \quad (4.2)$$

where $[u, v]^\top$ is the optical flow vector. This equation also expresses the case when the point P is static and the camera is moved with translational velocity $-\mathbf{T}$ and angular velocity $-\boldsymbol{\Omega}$ (*i.e.* it is the relative motion of observer and object that induces the optical flow).

From the equation 4.2 it can be seen that the depth in the scene, Z, can only be recovered if there is translation. Rotation alone does not convey any depth information; for this reason, to induce changes in the image plane from which the depth in the scene could be extracted, a translation movement is used.

Matthies et al. (1989) show that the most effective kind of translation is parallel to the image plane; the accuracy of the depth estimation is increased as the distance from the image features and the focus of expansion (FOE) increases. When the camera moves along the X or Y axis of the camera coordinate system the FOE is set at infinity, and a uniform accuracy is obtained for every point in the image. Because the camera is mounted on a mobile robot, the only possible translation is parallel to the ground plane; the camera is translated by moving the robot along the camera's Y axis.

Since the camera is only moved horizontally along the camera Y axis the motion parameters are $\{\mathbf{T} = (0, T_y, 0)^\top,\ \boldsymbol{\Omega} = (0, 0, 0)^\top\}$; substituting them into the equation

4.2 and solving for Z we obtain

$$Z = \frac{-fT_y}{v}.$$

Given that the conditions outlined by Adiv (1985) are fulfilled (see page 71), we can rewrite the previous equation in terms of image disparity and camera position as:

$$Z = \frac{-f\Delta Y}{\Delta y} \tag{4.3}$$

where ΔY is the camera displacement, and Δy is the image disparity. Uncertainties due to the estimation of the different variables of the last equation have to be taken into account in order to have a faithful model; an analysis of the different sources of variation and its representation will be given in section 4.5.

4.4 Estimation of disparity

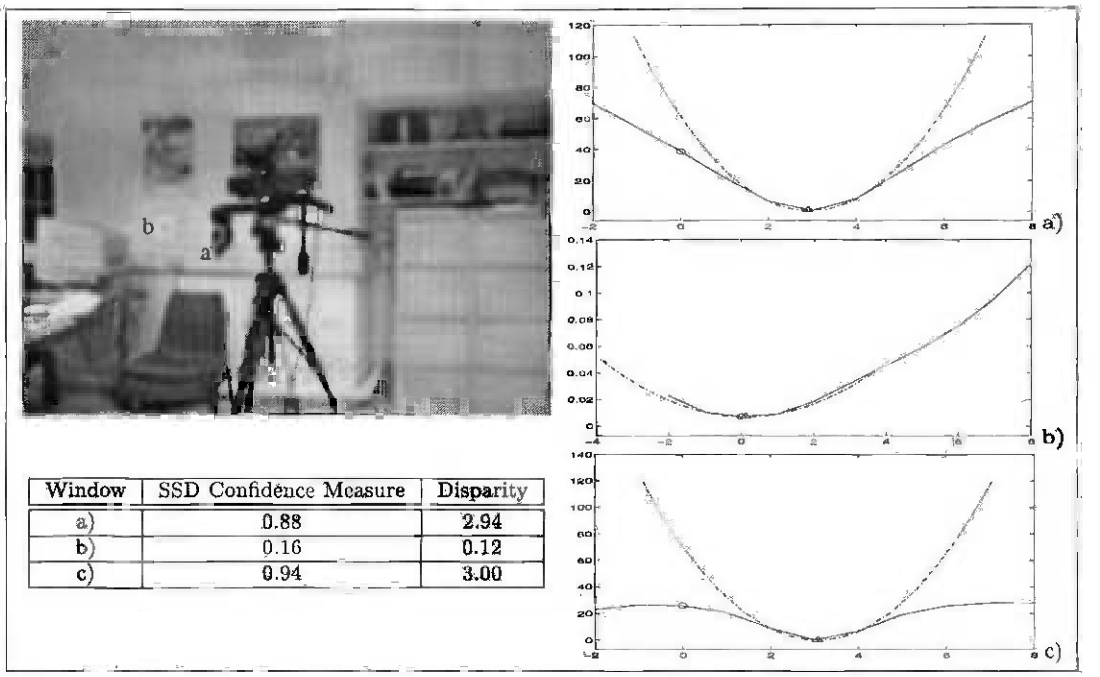
For the estimation of the disparity between two images a correlation-based technique called the sum of squared differences (SSD) was used (Anandan, 1984). This technique integrates the square differences of the intensity over small windows between two images; the result is an error measure of how well the windows compared match.

Given two images I_1 and I_2, the SSD error at the coordinates (x, y) in the image is given by

$$\begin{aligned} e(\Delta x, \Delta y; x, y) &= \int\int w(\alpha, \beta)[I_1(x - \Delta x + \alpha, y - \Delta y + \beta) - \\ &\qquad I_2(x + \alpha, y + \beta)]^2 \, d\alpha \, d\beta \end{aligned} \tag{4.4}$$

where $(\Delta x,\ \Delta y)$ expresses the difference of the position of the windows, α and β index over the windows used, and $w(\alpha, \beta)$ is symmetric, non-negative weighting function. In the work presented here the weight function used is defined as $w(\alpha, \beta) = 1$.

For each pixel in the first image, I_1, a window is extracted and compared with the adjacent windows in the second image, I_2. The set of SSD error measures in the vicinity of the position of the window forms what is called the SSD error surface. The best match occurs at the minimum point of the error surface; the difference in position between the windows with the best match is the estimated disparity $\Delta\tilde{y}$. As pointed out in section 4.3.2, since the camera movement used to induce the disparity is a translation



Window	SSD Confidence Measure	Disparity
a)	0.88	2.94
b)	0.16	0.12
c)	0.94	3.00

Figure 4.2: Example of SSD matching: The image I_1 in the top left shows the areas related to the SSD surfaces shown in plots a-c marked with a white square. The images a-c show the plot of the SSD surface around the minimum with a solid line, and the parabola fitted to it with a dashed line, the search starting point is marked with a circle, and the parabola's minimum with a triangle. The table at the bottom left shows the estimated disparity values for each case, with their respective confidence measure. For display purposes the size of the matching windows used for this example is 11×11, which gives a much smoother output than the 7×7 and 9×5 windows used for the experiments described at the end of this chapter.

along the Y axis of the camera coordinate system, we only expect to find disparities in the direction of the images scan lines, therefore we obtain a unidimensional SSD surface.

An important property of the SSD technique is that it provides a confidence measure of the disparity estimation; incorrect estimates are often obtained for those areas in the image that are homogeneous, or that become occluded between frames. Anandan (1984) proposes the use of the following confidence measure

$$C = \frac{e(\Delta\tilde{y}-1) - 2e(\Delta\tilde{y}) + e(\Delta\tilde{y}+1)}{e(\Delta\tilde{y}-1) + 2e(\Delta\tilde{y}) + e(\Delta\tilde{y}+1)} \tag{4.5}$$

which gives values in the interval $[0,1]$ (low and high confidence respectively). The numerator in the equation estimates the SSD error surface's second derivative with a 1×3 Laplacian operator, which, being centred at the SSD surface's minimum, will produce the second derivative of the parabola fitted to the minimum and its two adjacent points, and is directly proportional to the curvature of the SSD surface. The denom-

inator normalises the expression and makes it inversely proportional to the minimum value on the SSD surface.

In the implementation of the SSD disparity estimator described in this chapter, in a similar fashion to Matthies et al. (1989), instead of directly using the minimum of the SSD surface as the true estimate of the disparity, a parabola $a(\Delta y)^2 + b\Delta y + c = 0$ is fitted to the points $(e_{min}(\Delta\tilde{y} - 1),\ e_{min}(\Delta\tilde{y}),\ e_{min}(\Delta\tilde{y} + 1))$ and its minimum $(\Delta y = -b/2a)$ is used as a sub-pixel precision estimate of the disparity.

The confidence measure proposed by Anandan uses the minimum of the SSD error surface which is always greater than or equal to zero to estimate the disparity; when using the sub-pixel disparity estimation this is not necessarily the case, since the real minimum of the parabola could fall below zero. This tends to happens in those image areas with a high intensity gradient, where the minimum SSD error is small and the two lateral points around the minimum of the SSD surface are uneven. The effect of this on the confidence measure is that values larger than one can be obtained; in practice it was found that confidence values slightly larger than one are obtained when the three points used to characterise the SSD error surface at the position of the best match are produced by comparing image areas that represent the same spatial surface (where the changes in depth are smooth), while the confidence related to those image areas where depth discontinuities are present will produce even larger values (between 1.5 and 2.0).

For practical reasons, in order to keep the confidence measure bounded in the interval $[0, 1]$ a new confidence measure was used. The confidence is estimated using the following equation

$$C = \begin{cases} \frac{e(\Delta y-1)-2e(\Delta y)+e(\Delta y+1)}{e(\Delta y-1)+2e(\Delta y)+e(\Delta y+1)} & e(\Delta y) \geq 0 \\ \\ \frac{e(\Delta y-1)-2e(\Delta y)+e(\Delta y+1)}{e(\Delta y-1)-6e(\Delta y)+e(\Delta y+1)} & e(\Delta y) < 0 \end{cases} \tag{4.6}$$

which ensures that only magnitude of the error measure is taken into account when estimating the confidence.

The confidence measure is convenient since it allows us to rank the estimated disparities in terms of their reliability, and then eliminate those measures whose confidence is below a certain threshold. We also need to express the reliability in more general

terms, so that we can obtain an estimate of disparity error that would be expected.

An analytic expression for the uncertainty of the disparity was given by Matthies et al. (1989); they show that the variance of the disparity $\sigma^2_{\Delta y}$ is expressed as

$$\sigma^2_{\Delta y} = \frac{2\sigma_n^2}{a} \tag{4.7}$$

where σ_n^2 is the variance of uncorrelated Gaussian noise inherent to the camera sensor, and a is the quadratic term of the parabola fitted to the minimum of the SSD error surface. The shape of the parabola depends directly on the "average roughness" of the intensity surface; an area with small intensity changes will produce small differences in the error surface, and as a result the parabola fitted at the minimum of the surface error will have a wide aperture (a will be a small number), on the other hand if there are abrupt intensity changes, the parabola will have a small aperture (a will be a large number).

4.5 Depth uncertainty analysis

It is important to take into account how the uncertainty of the different variables involved in the depth equation propagates into the depth estimation process. The uncertainty in the depth estimation depends on the uncertainties of the disparity measure $\sigma^2_{\Delta y}$, the camera displacement σ^2_Y and the focal length σ^2_f; since these parameters are uncorrelated the uncertainty of the depth is given by

$$\sigma_Z^2 = \sigma^2_{\Delta y}\left(\frac{dZ}{d\Delta y}\right)^2 + \sigma_f^2\left(\frac{dZ}{df}\right)^2 + \sigma_Y^2\left(\frac{dZ}{d\Delta Y}\right)^2$$

where σ_Z^2 is the variance of the depth measure. Substituting equation 4.3 into this we obtain

$$\sigma_Z^2 = \sigma^2_{\Delta y}\left(\frac{f\Delta Y}{\Delta y^2}\right)^2 + \sigma_f^2\left(-\frac{\Delta Y}{\Delta y}\right)^2 + \sigma_Y^2\left(-\frac{f}{\Delta y}\right)^2. \tag{4.8}$$

In order to understand how much weight should be given to these different sources of error it is necessary to analyse them in the context of the physical implementation of the algorithm.

In the case of the uncertainty of the disparity estimation process ($\sigma^2_{\Delta y}$), it is directly proportional to the square of the distance between the camera and the objects situated

in the scene; this means that the quality of the disparity estimation rapidly degenerates with the distance to the object. It is also directly proportional to the uncertainty of the disparity estimation σ_d, which is described by equation 4.7, and which heavily depends on the local characteristics of the intensity image, such as illumination, and image gradient.

The uncertainty in the focal length (σ_f^2) has little influence on the accuracy of the depth estimation process; if we consider the variance of the focal length — estimated in appendix A as 0.0076 px^2 — and since the actual displacement of the camera is relatively small (in practice, camera displacements of 0.045 m were used), the standard error due to variance in focal length is of the order of 4 mm.

As was explained before, the images are grabbed at regular intervals while steadily moving the robot in a direction parallel to the image plane. Variations in the robot velocity, or in the timing of the image grabbing process, produce variations between the planned camera position for each frame and the actual camera position when the image was taken. These variations will induce changes in the estimated depth, but since the position of the camera is recorded at the moment each frame is taken and the difference in position between frames is used as the measure of the camera displacement, the main variation in displacement should be imputed to the robot odometry system, which "assigns" a particular position in space to each frame. Odometry has a bad reputation when it is used to calculate the position of a robot in navigation tasks, because when the robot moves long absolute distances, small errors are accumulated producing gross errors; in the case discussed here, the robot is moved only once, for a short distance, at relatively low speed. Another possible source of error could be a non-perfectly–parallel robot movement with respect to the image plane; such a problem could arise through imperfections on the floor surface, non-equal speed on the robot wheels or an error on the rotation of the pan-tilt unit controlling the camera position. The variations due to errors on the planned robot movement (non-perfectly–parallel movement and odometry errors) were considered negligible given the small distance used while moving the robot and the low speeds used during the capture process.

The main source of uncertainty in the depth estimation process is therefore the error in the disparity computation: its dependency on the distance to the object has a

greater influence than the other two cases. Also, the errors in the focal length and camera displacement are stable variables which depend on the internal characteristics of the hardware used, and their effect can be minimised by selecting appropriate working conditions, while the variance of the disparity estimation depends mainly on environmental conditions such as the degree of texture on the scene surfaces or the scene lighting (which affects the amount of Gaussian noise produced by the camera sensor) which can not be controlled. For these reasons it was considered safe to approximate the variance of the depth map obtained as

$$\sigma_Z^2 \simeq \sigma_d^2 \left(\frac{f \Delta Y}{\Delta y^2} \right)^2 . \tag{4.9}$$

4.6 Improving the quality of the depth map

A particular problem that occurs while estimating the depth in the scene from a sequence of images is occlusion. As was explained before, the disparity is basically estimated by comparing two image frames, each of them representing the projection of the 3-D scene into the image plane from a different position. Occlusion happens when the relative change in position between the camera and an object provokes the latter to obstruct a region of the image that would have been visible otherwise. The immediate consequence is that some regions in the image that were visible in one frame will not be in the other one, thus rendering disparity estimation in those areas impossible.

The degree of occlusion in an image pair depends on the distance between the camera and the objects in the scene, the size of the objects and the camera displacement between frames; of these three parameters only the third one is under our control, and we can minimise the occluded areas by moving the camera in small steps, with the disadvantage that the range of depth that can be estimated is constrained thereby.

In order to minimise the effects of occlusion, increase the depth range and optimise the quality of the depth measurements obtained, a multi-level disparity estimator was implemented. The method uses increasingly larger camera displacements to estimate a set of disparity maps which are then mixed to form a unique, more reliable disparity map. Every disparity map is calculated with respect to a central frame and at each level two frames are grabbed, each of them being captured with the camera situated

1. Move the camera along the Y axis steadily, and grab a set of frames $\mathcal{F}_i$, $i = 0...n-1$ at regular intervals (n is an odd number). Record the camera position at the instant each frame was grabbed using the robot odometry.

2. The disparity is estimated in $j = 1..(n-1)/2$ levels. In each level two disparity maps estimated between the central frame $\mathcal{F}_c$, $c = n-(n-1)/2$ and the frames $\mathcal{F}_i$, $i = c \pm j$, and then mixed to form a unique disparity $\mathcal{D}_j$.

3. From the second level onwards, the disparity map obtained in the previous level $\mathcal{D}_{j-1}$ is mixed with the one obtained at the current level.

4. The confidence map for the final disparity map is calculated and then used to threshold the disparity map in order to discard unreliable measures. Finally, given the known camera displacement and focal length, the depth map is obtained using the equation 4.3.

Table 4.1: The Depth estimation procedure.

at the same distance from the camera's central position but in opposite directions (see figure 4.3). Two disparity maps are estimated at each level and mixed to form a single disparity map. In the same fashion, when the disparity maps at each level have been mixed, the resulting map is mixed with the map obtained at the previous level. The last step is to eliminate those disparity estimations with a low confidence measure (below a certain predefined threshold), and to calculate the depth map using equation 4.3. The mixing procedure is explained in the following subsection. An algorithmic description of the process is given in table 4.1.

4.6.1 Mixing disparity maps

The objective behind using a mixture of disparity maps is twofold. First, we attempt to minimise occlusion, without sacrificing the range of depths that could be estimated, and also we try to improve the quality of the disparity estimation.

On each level two disparity maps are mixed; since they are the product of camera translations in different directions it is expected that most of the occlusions that are produced by the camera's movement in one direction will not be produced if the camera is translated in the opposite direction. Also, by mixing disparity maps due to different translations (mixing results between levels), we manage to avoid gross occlusions.

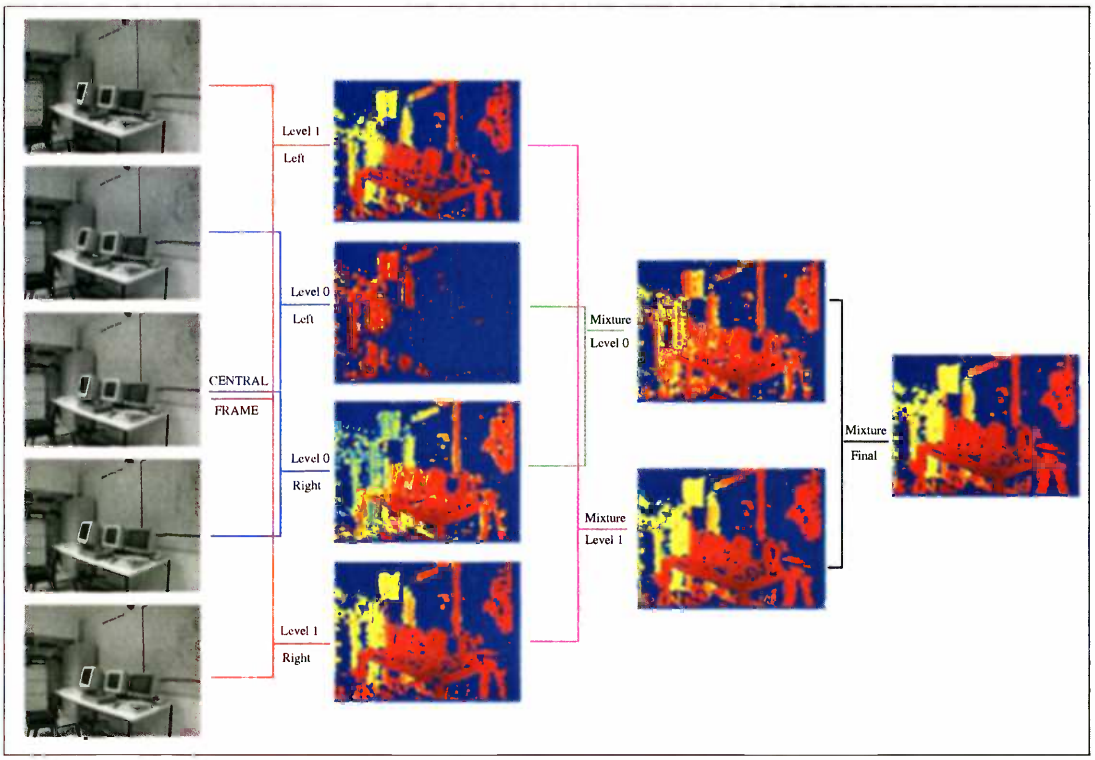


Figure 4.3: Diagram of the multilevel depth estimation procedure (two levels used).

The other advantage of mixing disparity maps is that the we can obtain a better estimate of the disparity, since we have more than one measure for each point in the image. The strategy used to combine two disparity maps is by using a weighted sum of the different measures, where the weights are chosen by the reliability of each measurement. The reliability is expressed in terms of the depth variance σ_Z^2, in order to avoid the scaling problems that arise when the two disparity maps to be mixed are not due to the same translational amount, as happens when the camera movement is not quite precise or when the disparity maps to be mixed were produced at different levels.

The depth mixture is defined as

$$\hat{Z} = W_1 Z_1 + W_2 Z_2 \ , \qquad W_1 + W_2 = 1 \tag{4.10}$$

where $\hat{Z}$ is the optimally mixed depth, Z_1, Z_2 are the two depth measures to be mixed and W_1, W_2 are the respective weights. Since the two depth estimations are independent, the variance of the mixture is defined as

$$\sigma_{\hat{Z}}^2 = W_1^2\sigma_{Z_1}^2 + W_2^2\sigma_{Z_2}^2 \tag{4.11}$$

The depth reliability is optimised by minimising the variance of the mixed depth measure; under the constraint $W_1 + W_2 = 1$, the weights that minimise the equation 4.11 are

$$W_1 = \frac{\sigma_{Z_2}^2}{\sigma_{Z_1}^2 + \sigma_{Z_2}^2}, \qquad W_2 = \frac{\sigma_{Z_1}^2}{\sigma_{Z_1}^2 + \sigma_{Z_2}^2}. \tag{4.12}$$

By substituting equation 4.12 into equation 4.11 we find that the variance of the optimal depth estimate is expressed as

$$\sigma_{\hat{Z}}^2 = \frac{\sigma_{Z_1}^2\sigma_{Z_2}^2}{\sigma_{Z_1}^2 + \sigma_{Z_2}^2}. \tag{4.13}$$

Although a depth estimation quality measure can be obtained from the depth variance, it is not a straightforward procedure since it is strongly influenced by the distance between the objects in the scene and the camera; on the other hand, the disparity variance is independent of the 3-D scene structure, allowing us to directly extract a confidence measure from it. The use of the Anandan confidence measure, which is described in terms of the disparity variance and the fitting error, is a convenient option given its bounded nature which facilitates the task of choosing a threshold value to eliminate unreliable measures.

As explained before, the Anandan confidence measure is defined in terms of the parabola fitted to the minimum of the SSD surface. Therefore, we need to find how the parabolic parameters from each of the disparity maps to be mixed are propagated into the optimal disparity map. It is important to note that although the mixture is defined in terms of the depth variance, what is actually mixed is the disparity maps.

The first step taken in order to mix two disparity maps is to estimate their depths (Z_1, Z_2) and their variances ($\sigma_{Z_1}^2$, $\sigma_{Z_2}^2$) using equations 4.3 and 4.9 respectively. The optimal depth and its variance are then calculated using the equations 4.10, 4.12 and 4.13. The next step is to find for each depth measure the parameters of the parabola $\hat{a}\Delta y^2 + \hat{b}\Delta y + \hat{c} = 0$, from which we could reconstruct the optimal depth and variance

maps just calculated. For this it is necessary to describe the relationship between the parabola parameters and the disparity measures obtained and the optimal depth and variance equations.

The optimal depth variance can be described in terms of their lowest level denominators by substituting the disparity variance given by equation 4.7 into depth variance equation 4.9, and then the result into equation 4.13:

$$\sigma_{\hat{Z}}^2 = 2f^2\sigma_n^2 \frac{\Delta Y_1^2 \Delta Y_2^2}{a_1 \Delta y_1^4 \Delta Y_2^2 + a_2 \Delta y_2^4 \Delta Y_1^2}. \tag{4.14}$$

In turn we can also express the optimal depth variance in terms of an hypothetical optimal quadratic term, camera displacement and image disparity as follows:

$$\sigma_{\hat{Z}}^2 = 2f^2\sigma_n^2 \frac{\Delta \hat{Y}^2}{\hat{a}\Delta \hat{y}^4} \tag{4.15}$$

if we combine the last two equations we find that the relationship between the free parameters of the original depth maps and the optimal depth map is given by

$$\hat{a}\frac{\Delta \hat{y}^4}{\Delta \hat{Y}^2} = a_1 \frac{\Delta y_1^4}{\Delta Y_1^2} + a_2 \frac{\Delta y_2^4}{\Delta Y_2^2}.$$

The optimal camera displacement $\Delta \hat{Y}$ has to be the same for the whole map; for simplicity the largest camera displacement of the two maps to be mixed is assigned to it. The optimal disparity Δy is calculated by substituting the optimal depth $\hat{Z}$ into equation 4.3. The optimal quadratic term is then expressed as:

$$\hat{a} = \frac{\hat{Z}^4}{f^4 \Delta Y_1^2}\left(a_1 \frac{\Delta y_1^4}{\Delta Y_1^2} + a_2 \frac{\Delta y_2^4}{\Delta Y_2^2}\right), \quad \Delta Y_1 > \Delta Y_2. \tag{4.16}$$

The optimal linear term $\hat{b}$ can be defined in terms the disparity $\Delta \hat{y}$, and the quadratic term $\hat{a}$, by using the expression that gives the minimum of the parabola $\Delta \hat{y} = -\hat{b}/2\hat{a}$.

$$\hat{b} = -2f\frac{\hat{a}\Delta Y_1}{\hat{Z}}. \tag{4.17}$$

The independent term $\hat{c}$ depends on the amount of noise present in the images and it defines the matching error at the minimum of the parabola

$$e(x) = ax^2 + bx + c\,, \quad x = -\frac{b}{2a}$$

While the mixing procedure directly involves the quadratic and linear terms of the parabola, it does not affect the matching error, and therefore $\hat{c}$; for this reason it was

decided to define it in terms of the weighted average of the matching error of the two disparities to be mixed $(e_1(\Delta y_1), e_2(\Delta y_2))$. The optimal independent term is then defined as:

$$\hat{c} = e_1(\Delta y_1)W_1 + e_2(\Delta y_2)W_2 + \frac{\hat{b}^2}{4\hat{a}} \tag{4.18}$$

where $(W_1,\ W_2)$ are the averaging weights defined in equation 4.12.

Once the parabolic terms that define the optimal disparity measure have been calculated, its confidence measure can be estimated by substituting the error values at the minimum and adjacent points of the optimal parabola, into equation 4.6.

4.7 Implementation of the depth estimation procedure

4.7.1 Physical constraints to the depth estimation procedure

There are certain parameters that have to be defined for the implementation of the depth estimation procedure; it is necessary to define what range of depths is needed given the task that we have in mind, and how precise those measurements should be. Once the depth range has been defined, we can find, given the camera internal parameters, how much disparity has to be induced in order to estimate the depth map within the required range, and how much should we move the camera for that purpose.

The depth in the scene is needed in order to find constraints for the search for faces in the scene in front of the camera, therefore we are interested in the range of depths where a face might be found. Knowing that a human face has an average size of 14.5 ± 3 cm(Jürgens et al., 1990), and that the smallest face pattern detected by the face classifier described in chapter 3 has a size of 20×20 pixels, we can substitute those parameters (the largest face that we might expect to find and the size of the window used by the face classifier) into the pinhole camera equation 4.1 and define the maximum depth as

$$Z_{max} = \frac{fY_{face}}{y_{face}} = \frac{(477.71 \text{ px})(0.175 \text{ m})}{20 \text{ px}} = 4.18 \text{ m}. \tag{4.19}$$

The minimum depth depends on the maximum face size that could be seen by the camera. Given that the image obtained by the camera has a resolution of 640×480 pixels, the minimum distance between the face and the camera is $Z_{min} \approx 0.17$m.

Now the range of depths has been defined we need to find the minimum necessary camera translation to induce the flow in the scene that would allows us to recover the depth in the predetermined range. The disparity was calculated using an evenly spaced grid of 320×240 points on a 640×480 image. Substituting the minimum disparity of $\Delta y = 1$ pixel and the maximum distance Z_{min} in the projective equation 4.3 we find the minimum required translation

$$\Delta Y_{min} = \frac{Z_{max}\,\Delta y_{min}}{f} = \frac{(4.18 \text{ m})(1 \text{ px})}{477.71 \text{ px}} \simeq 1 \text{ cm}.$$

Three levels were used for the disparity estimation; in order to achieve a higher accuracy while moving the robot a larger translation than required was used. The camera was uniformly translated and the images were grabbed at regular intervals such that the translational difference between frames was 2.5 cm; with this, the displacements between the central frame and the lateral frames were 2.5, 5.0 and 7.5 cm for each level respectively. The exact position of the camera is recorded at the moment each image is grabbed and is obtained using the robot odometry. An algorithmic description of the method was given in table 4.1.

4.7.2 Moving the Camera

The camera is mounted on top of a Real World Interface B21 mobile robot; the robot has a cylindrical shape, it can either move forwards or backwards, and it can rotate around its vertical axis. On the top of the robot, there is a pan-tilt unit with a CCD camera attached to it, at approximately 15 cm from the robot rotation axis, and 1.25 metres from the ground plane. The pan-tilt unit gives two degrees of freedom to the camera, rotating around its x–axis and y–axis (see image (**a**) in figure 4.4).

It is necessary to move the robot's body in order to change the position of the camera; as explained before, it was decided to use a movement direction parallel to the image plane (the XY plane in the camera coordinate system). Two movement strategies were tested. For the first one a quasi-lateral movement of the camera was achieved by rotating the robot body by an angle α while rotating the camera around the X axis (panning) by the same angle in the opposite direction; if the angle is small, the difference in the camera position along the Z axis is minuscule and can be ignored. The B21 robot and camera control units are independent and theoretically it is not

possible to coordinate them in order to achieve a smooth continuous movement while keeping the camera body perpendicular to the XY plane. For this reason a discrete approach was used, where the images are captured with the robot body and camera static and set at a particular set of configurations which lies within the ideal robot-camera continuous trajectory. The set of configurations are calculated in such way that the robot-camera original position corresponds to the central frame, while the rest of images are taken at symmetrical positions with respect to the original position, keeping the difference in camera displacement with respect to the central position along the Y axis as a multiple of ΔY_{min}.

The second strategy first rotates the robot body 90 degrees in one direction while panning the camera 90 degrees in the opposite direction; with this arrangement the robot direction of movement is orthogonal to the camera orientation, and by translating the robot (forward or backward), a smooth continuous lateral movement of the camera is achieved. The images are captured while the robot is translating; the desired displacement between frames is achieved by controlling the translational speed and the inter-frame capture time (see image (c) in figure 4.4).

The main advantage of the rotational strategy is that it is not necessary to translate the robot, and the depth in the scene can be estimated on the spot, something that is particularly useful when the area around the robot is occupied. On the other hand, the main disadvantage is that the camera position estimation is not very reliable; although using the robot odometry we can have a good estimate of the rotation of the robot body, the pan-tilt unit does not have any kind of feedback mechanism that would allow us to determine the actual orientation of the camera (during the implementation it was noted that the pan-tilt unit has a fair amount of play).

In the case of the translational movement, the main advantage is the simplicity of the approach, and the fact that good estimations of the position of the robot at any time can be obtained. The main disadvantage is that it requires a certain amount of space around the robot. Results obtained with each approach are analysed and discussed in section 4.8.

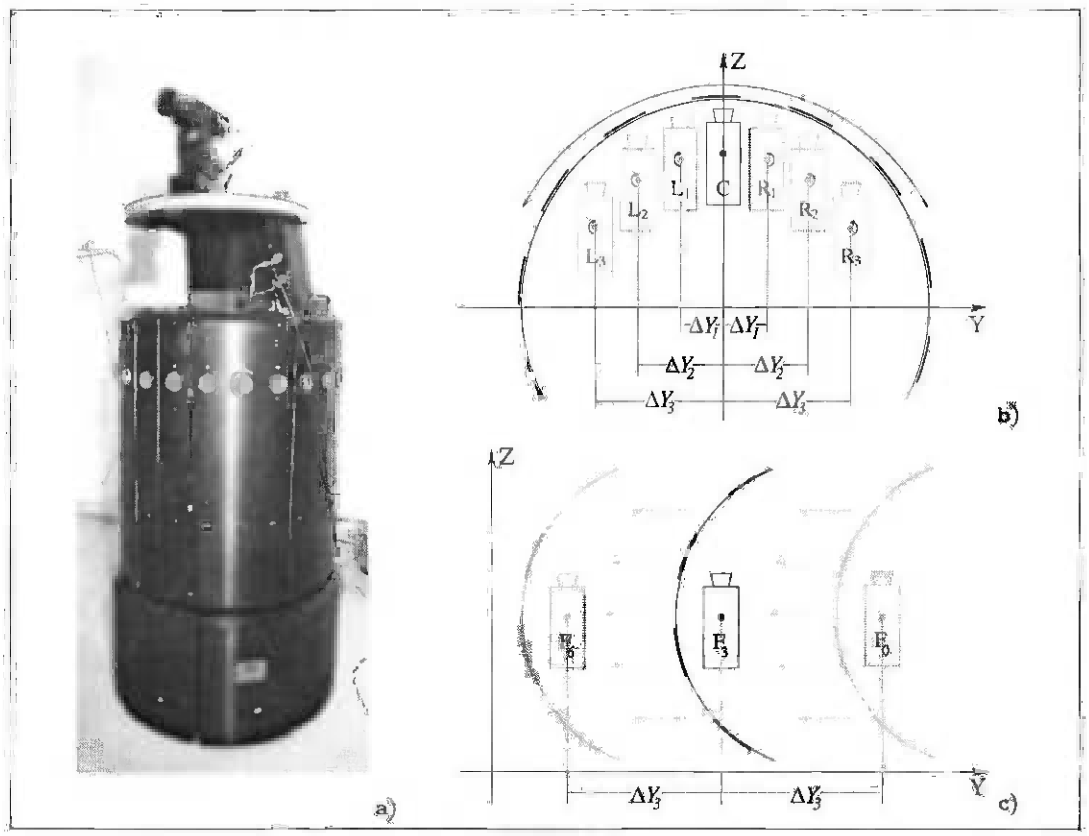


Figure 4.4: Robot motion: a) the B21 robot "Gillespie". Image (b) shows how the robot moves in the rotational strategy for camera translation (the rotation angles are exaggerated for clarity). Image (c) shows the robot configuration with the translational strategy.

4.7.3 Image noise quantisation

The amount of Gaussian white noise present in the images was estimated by obtaining some statistics from a set of images captured using the robot's camera.

Thirty 640×480 images were captured; the scene in front of the camera was static and special care was taken to avoid illumination changes in the scene. The pixel-wise mean of the images was calculated and then its difference with respect to each image was used as a set of samples of the image noise; the sample mean and variance of the noise images was then calculated, and from them the mean and variance were used to characterise the image noise.

Two cases were considered: in the first case the statistics were obtained from the raw images, in the second one the optical distortion correction process described in appendix A was applied to each of the images beforehand in order to consider the pos-

sible disturbances induced by the imaging process (a bilinear interpolation mapping). The variance estimated was 2.64 and 3.40 squared intensity units[6] respectively.

4.8 Experimental results

Two different experiments were tried. The first experiment uses a sequence of artificial images in order to evaluate the performance of the disparity estimator under controlled conditions. The second one evaluates the quality of the depth estimator as the distance between the objects in the scene and the observer is increased, using real images.

4.8.1 Depth estimation results on artificial images

The artificial images are composed by projecting a set of flat geometric figures placed into an 3-D scene, into the image plane. Two different kinds of artificial images were generated: in the first kind the background as well as the figures are "painted" with random dots, while in the second kind a grey-level texture is used. The random-dot images can be considered ideal, since they are rich in high spatial frequencies which allow us to make very good matches for every part of the image. On the other hand, textured images are more similar to real images, where there is a wider variety of spatial frequencies. Examples of the artificial images can be seen in figure 4.5, images (**a**) and (**d**).

The camera movement is simulated by changing the object positions between frames; in order to avoid quantisation errors due to the fact that only discrete object movements can be simulated, images with a high horizontal resolution were generated, and then smoothed with a Gaussian filter and down-scaled to form a 640×480 image. Finally Gaussian noise with a variance similar to the one estimated using the real camera is added to each of the images of the sequence.

Four objects are painted on each frame: a circle, a triangle, an irregular blob and a square, and their distance to the camera is respectively 1,2,3 and 4 meters. The background is set at infinity and the camera displacement between frames is 1.88 cm.

[6] The intensity of a pixel is a given as number between 0 and 255.

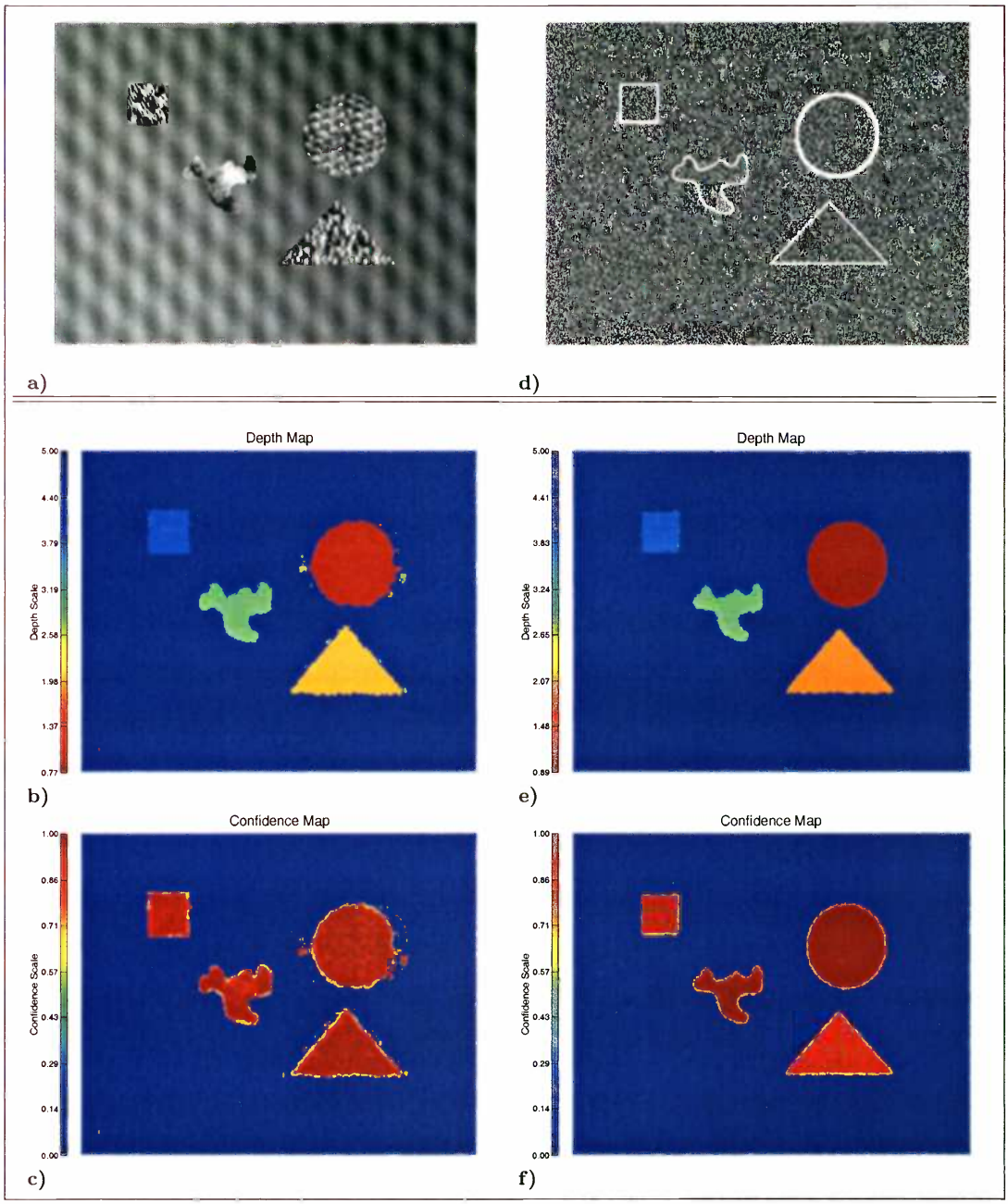


Figure 4.5: Depth estimation examples with artificial images. Two examples are shown. The left column shows the results of the depth estimation procedure using a sequence of seven artificial textured images. The right column shows a similar example, using instead random-dot images. The top row (images (**a**) and (**d**)) shows examples of the central frames of each sequence (for clarity in (**d**), the silhouette of the figures has been super-imposed). The second row (images (**b**) and (**e**)) shows the estimated depth map for each case, and the third row (images (**c**) and (**f**)) shows the respective confidence maps. The depth maps shown have already been thresholded, eliminating those measurements whose confidence is lower than 0.5 and setting them to the maximum detectable distance (5 m). 640 × 480 images were used, with the focal length set to 477.71 px and Gaussian noise with mean zero and variance 3.4 was added to each image. A 7×7 window was used for the disparity estimation.

Object	Random-dot Sequence			Textured Sequence		
	Estimated Depth	Estimated Std. error	Depth Std. error	Estimated Depth	Estimated Std. error	Depth Std. error
Circle	0.999 m	0 0 m	$0\ 139 \times 10^{-3}$ m	0 999 m	0.0	2.029×10^{-3} m
Triangle	1 998 m	$1\ 414 \times 10^{-3}$ m	$1\ 372 \times 10^{-3}$ m	1 999 m	$1\ 414 \times 10^{-3}$ m	4.707×10^{-3} m
Blob	2 999 m	$3\ 462 \times 10^{-3}$ m	$2\ 101 \times 10^{-3}$ m	2 999 m	$3\ 443 \times 10^{-3}$ m	$13\ 373 \times 10^{-3}$ m
Square	3 943 m	$5\ 890 \times 10^{-3}$ m	$13\ 717 \times 10^{-3}$ m	3 982 m	$6\ 004 \times 10^{-3}$ m	$18\ 893 \times 10^{-3}$ m

Table 4.2: Artificial image sequence depth estimation results. The three columns on each category show the mean of the estimated depth, the mean of the estimated standard error, and the actual standard error of the depth estimation.

Once the image sequence was generated, the depth and confidence maps were estimated, producing depth, confidence and variance maps. Each figure in the image was cropped by hand and, after discarding the background, unreliable measures (those whose confidence measure is lower than 0.5) and outliers (by trimming the 2.5 % of the measures at the top and bottom of the histogram), the mean depth and depth standard error for each figure was computed; the same procedure was used with the variance map in order to obtain an approximation of the estimated standard error for each figure. The results are shown in table 4.2.

As was expected, the results obtained from the random-dot sequence are generally better than the ones obtained from the textured sequence; for the random-dot sequence the depth standard error is (for the three closest figures) smaller than the estimated standard error, while for the textured sequence, all the standard errors are larger than expected. This can be explained by the fact that, given the uncorrelated nature of the random-dot images, perfect matches can be made at every position of the scene, while with the textured images it is more likely to obtain mismatches since there is a smaller intensity difference between adjacent parts of the image; also the spatial frequencies of the random dot images are high and even, which gives the advantage that the size of the matching window used to estimate the disparity does not affect the quality of the measurement. In the textured sequence there is a wider range of frequencies which limits the amount of matching features available, given a fixed-size window. These phenomena are made clear in the confidence map shown in figure 4.5 where it can be seen that, for the random-dot sequence, the confidence measure of each figure has little variance and the figures' boundaries are well defined, while the same can not be said about the results obtained from the textured sequence.

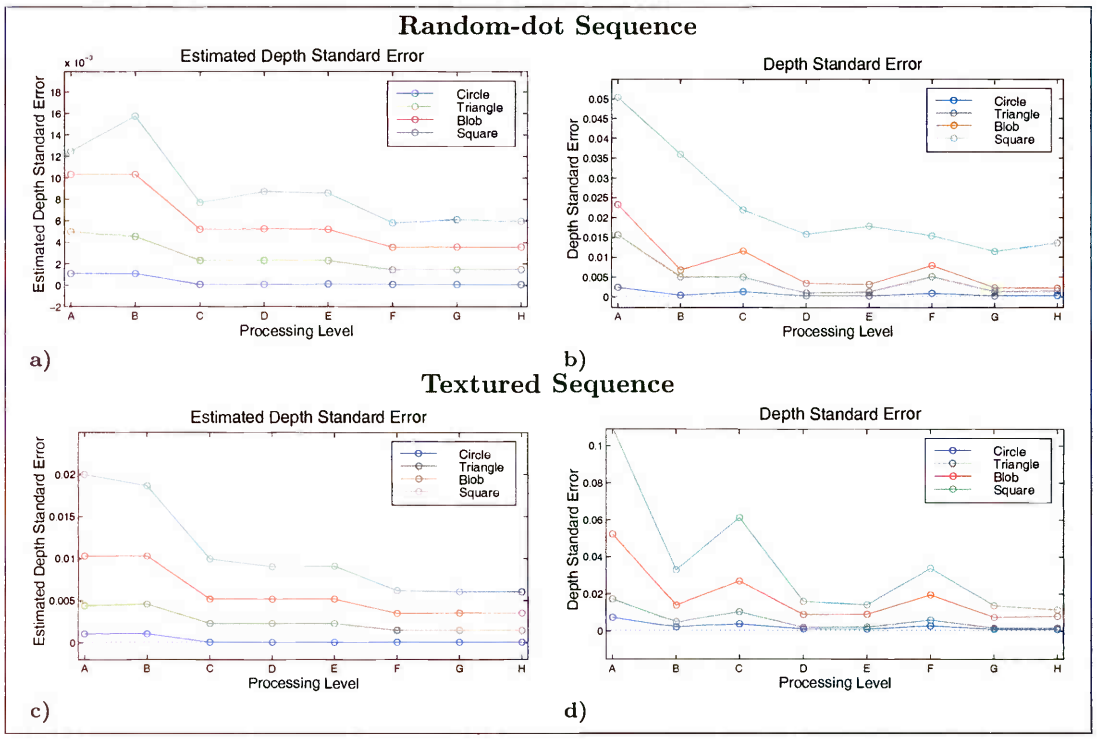


Figure 4.6: Depth estimation sensitivity on artificial images: Each row shows the estimated and actual depth standard error obtained from the random-dot and textured sequences respectively. The positions along the abscissa on each graph indicate the step within the recursive depth estimation procedure: **A**, **C** and **F** correspond to the estimation from a single pair of images at each level (no mixing). **B**, **D** and **G** correspond to the mixture of the two symmetric maps at each level and **E** and **H** to the recursive mixing of the maps produced at the current and previous levels.

Figure 4.6 shows how the estimated and actual depth standard error changes as the multilevel process advances. The different steps within the process can be classified in three different states, which are shown on each graph: the "single" state corresponds to those maps obtained from a single pair of images at each level (*e.g.* **A**, **C** and **F**); the intralevel states, where the results correspond to the mixture of the two symmetric maps estimated at each level (*e.g.* **B**, **D** and **G**), and the interlevel states, where disparity maps from different levels are mixed — **E** corresponds to the mixture of the first and second intralevel disparity maps, while **H** is the final map and corresponds to the mixture of **E** and **G**.

The general tendency, as expected, is that the estimated and depth standard error diminish as the process advances and most of the improvement is made with the intralevel mixing. It should be noted that, as expected, as the camera displacement

is increased the estimated and depth standard error diminishes, especially for objects that are not close the camera; a direct consequence of this is that little improvement, in terms of minimising the error, is achieved with the intralevel mixing, since the variance measures to be mixed tend to be uneven.

4.8.2 Depth estimation results on real images

The depth estimation method proposed in this chapter was also tested with real images obtained from the laboratory. Four sequences of images were obtained from the scene, using the translational and rotational strategies described in section 4.7 and small and large camera displacements. Each image is 640×480, and the mapping described in appendix A was used to eliminate optical distortion.

The reason for using camera displacements of different magnitude in each strategy is to evaluate how much it affects the performance of the depth estimation process. In section 4.7 it was mentioned that the minimum displacement in order to obtain a disparity of at least 1 pixel for the maximum distance allowed ($\simeq$ 4.5 m), is approximately 1 cm; in order to obtain a steady camera movement, larger displacements than strictly needed were used. In the case of the rotational strategy we are restricted by the camera position within the robot body, and also smaller movements have to be used in order to minimise the camera displacement along the Z axis.

For the translational trajectory, inter-frame camera displacements of 2.5 and 4 cm are used in each case, while for the rotational trajectory, the robot was rotated an angle such that the inter-frame camera translations along the Y axis were 3.4 and 4 cm. Given inaccuracies in the robot and camera control motion, the displacements are not perfect, but a good estimation of the camera position was recorded at the instant each frame was grabbed. The positions of the camera with respect to the central frame position for each sequence are given in table 4.3.

In the scene were set four different targets, at different distances from the camera. All of them are cardboard figures with a pattern printed on them; the first one is at a distance of 1.38 m and is located close to the upper-left corner of the scene, the second one is at a distance of 2.37 m and is located in the lower-left part of the scene, the third

Movement	Displacement		L3	L2	L1	C	R1	R2	R3
Translational	small	(x)	0 073 m	0.037 m	0.019 m	0 000 m	-0.027 m	-0 045 m	-0 073 m
		(y)	–	–	–	–	–	–	–
	large	(x)	0 110 m	0.064 m	0 028 m	0 000 m	-0 036 m	-0 063 m	-0.100 m
		(y)	–	–	–	–	–	–	–
Rotational	small	(x)	0.110 m	0.076 m	0 035 m	0 000 m	-0 035 m	-0 075 m	-0 111 m
		(y)	-0 046 m	-0 020 m	-0 004 m	0.000 m	-0 004 m	-0.019 m	-0 046 m
	large	(x)	0 119 m	0 080 m	0 039 m	0 000 m	-0 041 m	-0.081 m	-0 121 m
		(y)	-0 058 m	-0 023 m	-0 005 m	0 000 m	-0 005 m	-0.022 m	-0 055 m

Table 4.3: Camera displacements with respect to the central frame for the real image sequences.

target is situated at 3.14 m from the camera and is situated in the middle-right part of the scene, and finally the last target, which has a circular shape and is situated roughly at the centre of the scene, is 3.97 m away from the camera. These measurements are for the translational trajectory sequence; in the case of the rotational trajectory 0.19 m has to be subtracted from the previous figures, given the difference in the camera position with respect to the frontal part of the robot. In the rest of this chapter each of the figures in the scene will be referred by their closeness to the camera as "first", "second", "third" and "fourth". The central frames of the sequences can be seen in the top row of figures 4.7 or 4.8.

Once the sequence of images were obtained, the related depth, confidence and variance maps are calculated using the procedure described in this chapter. As with the artificial images, each figure in the image was cropped by hand and, after discarding the background, unreliable measures (those whose confidence measure is lower than 0.5) and outliers (by trimming the 2.5 % of the measures at the top and bottom of the histogram), the mean depth and depth standard error for each figure were computed; the same procedure was used on the variance map in order to obtain an approximation of the estimated standard error of each figure. The results for the small and large displacement of the translational trajectory are shown in table 4.4, and the results for the rotational trajectory are shown in table 4.5. The depth and variance maps for the small and large displacements can be seen in figures 4.7 and 4.8 respectively.

The first fact to note in both cases (the translational and rotational strategies) is that the difference between the estimated and computed standard error is much bigger than obtained from the artificial image sequences; this can be explained by the lack of homogeneity present in the real images, which allows a wider range of quality of the matches made by the disparity estimator, as can be seen in the confidence maps shown

Object	Small Displacement Sequence			Large Displacement Sequence		
	Estimated Depth	Estimated Std. error	Depth Std. error	Estimated Depth	Estimated Std. error	Depth Std. error
First	0.999 m.	0 0 m	64.5×10^{-3} m	1 069 m	0 0 m	$52\ 8 \times 10^{-3}$ m.
Second	1.833 m.	$0\ 9 \times 10^{-3} m$	171.8×10^{-3} m	1 990 m	$0\ 9 \times 10^{-3} m$	$101\ 2 \times 10^{-3}$ m.
Third	2 496 m	$1\ 8 \times 10^{-3}$ m.	$229\ 2 \times 10^{-3}$ m	2 636 m	$1\ 3 \times 10^{-3}$ m	$165\ 4 \times 10^{-3}$ m
Fourth	3.007 m.	2.7×10^{-3} m	$91\ 1 \times 10^{-3}$ m.	3 182	2.0×10^{-3} m.	$47\ 0 \times 10^{-3}$ m

Table 4.4: Real image sequence depth estimation results for the translational trajectory. Two categories are shown, corresponding each to small and large camera displacements respectively. The three columns in each category show the mean of the estimated depth, the mean of the estimated standard error, and the actual standard error of the depth estimation.

in figures 4.7 and 4.8.

The difference between the small and large displacement results, in both cases, proves that better results can be obtained by using large displacements, in terms of a smaller depth variance and marginally in terms of greater accuracy, with the main drawback that it is more computationally expensive, since a larger area in the SSD surface has to be analysed to find the minimum error. The general trend, as expected, is that the depth standard error is directly proportional to the distance between the object and the camera, although apparent contradictory results are observed for the fourth object which, being the most distant one, should have had a greater standard error than any of the other cases. This can be explained by the fact that the fourth object is the one with the largest area, and therefore it holds a larger number of features which ensures a better matching quality, and it is also the only one with non-regular patterns printed on it, which also helps to ensure the uniqueness of the matching process.

In the same fashion as for the artificial images, the evolution of the estimated and real depth standard error, as the depth estimation procedure advances, was performed on the real image sequences; also, a similar analysis was applied to the mean of the estimated depth, and the percentage of the reliable measures for each figure in the scene (see figures 4.9 and 4.10).

In general, the evolution of the depth standard error tends to be similar to the one obtained with the artificial images, except for the third figure, which does not quite follow the same pattern. In the case of the depth mean, graphs (a) and (e) in both figures show that, as the process estimation advances, the mean depth of each scene figure tends to diminish, and that this effect is more accentuated as the

Object	Small Displacement Sequence			Large Displacement Sequence		
	Estimated Depth	Estimated Std. error	Depth Std. error	Estimated Depth	Estimated Std. error	Depth Std. error
First	0.927 m	0.0 m	49.6×10^{-3} m	0.924 m	0.0 m	59.7×10^{-3} m
Second	1.588 m	0.0 m	143.0×10^{-3} m	1.737 m	0.0 m	192.2×10^{-3} m
Third	2.272 m	0.9×10^{-3} m	172.5×10^{-3} m	2.687 m	1.3×10^{-3} m	217.7×10^{-3} m
Fourth	2.568 m	1.3×10^{-3} m	91.8×10^{-3} m	3.324 m	2.0×10^{-3} m	129.5×10^{-3} m

Table 4.5: Real image sequence depth estimation results for the rotation trajectory. Two categories are shown, corresponding each to small and large camera displacements respectively. The three columns in each category show the mean of the estimated depth, the mean of the estimated standard error, and the actual standard error of the depth estimation.

distance between the object and the camera increases. This happens because when two independent measurements with similar disparity variance, are mixed, the one that is closest to the camera will have a smaller variance, thus biasing the depth mixture towards zero (see equation 4.9).

Finally, graphs (d) and (**h**) in both figures show how the quality of the estimated depth map, in terms of number of reliable measures per figure, increases as the process advances. Each plot shows the percentage of the reliable measurements obtained from each figure. It is clear how the percentage increases in two levels, first as the single maps on each level are mixed, and in a second level, as the interlevel maps are mixed.

4.9 Conclusion

In this chapter a depth estimation method was proposed. It uses a multi-level hierarchical method to iteratively obtain a depth map from a set of disparity maps, estimated from a sequence of images captured from a moving camera.

The proposed technique was tested with artificial and real image sequences, and reliable depth maps were obtained. Although only an iconic representation of the scene structures were obtained, *i.e.* the surfaces on the scene are not explicitly represented, which as a consequence produced uneven depth maps, where the distance between the observer and the scene was only accurately estimated on textured surfaces, it proved adequate for the work presented in this thesis, as will be seen in the following chapter.

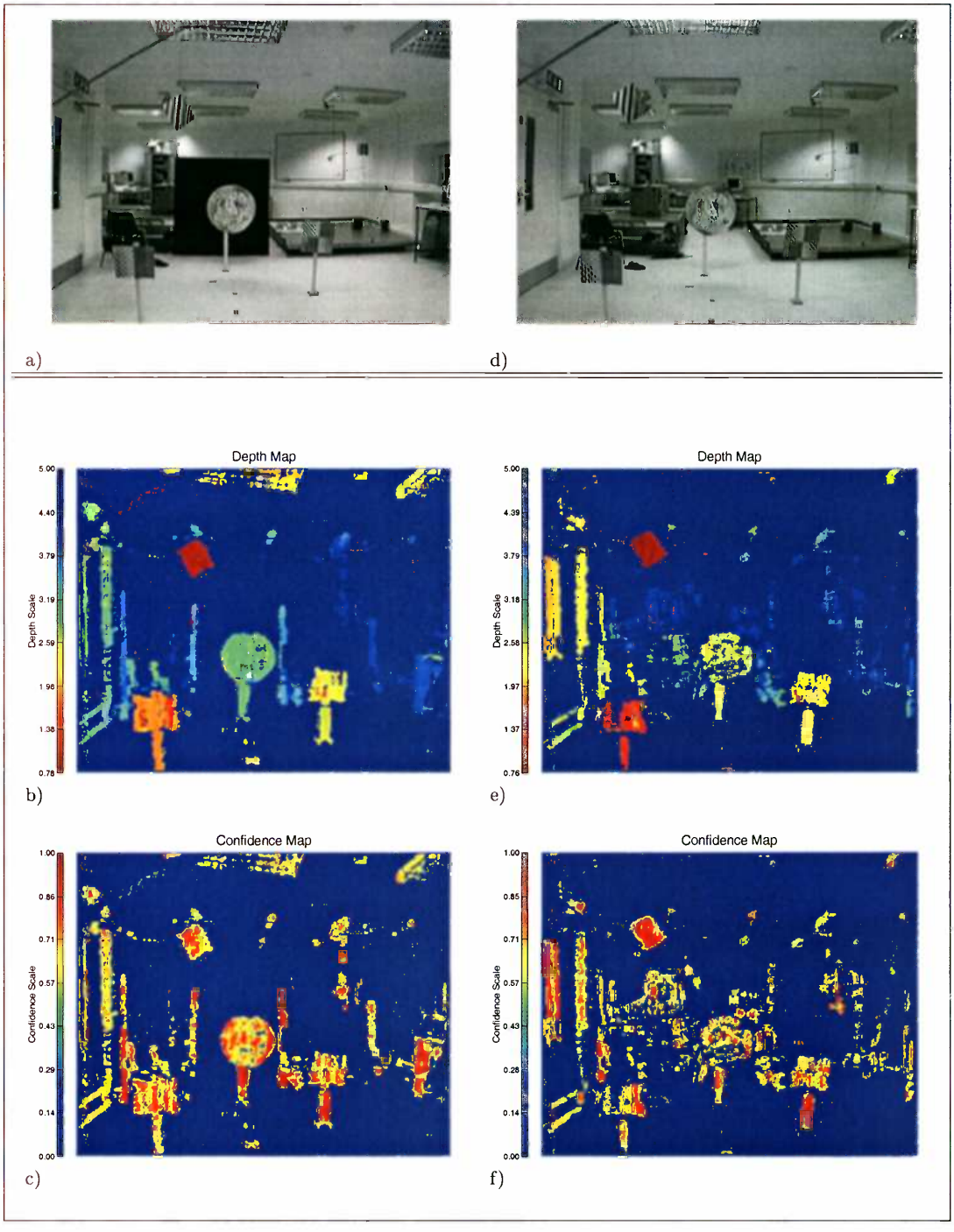


Figure 4.7: Depth estimation examples with real images (small displacement). Two examples are shown. The left column shows the results of the depth estimation procedure for the translational movement strategy. The right column shows the results of the depth estimation procedure for the rotational movement strategy. The top row (images **a** and **d**) shows examples of the central frames of each sequence. The second row (images **b** and **e**) shows the estimated depth map for each case, and the third row (images **c** and **f**) shows the respective confidence maps. The depth maps shown have already been thresholded, eliminating those measurements whose confidence is lower than 0.5 by setting them to the maximum detectable distance (5 m). 640×480 images were used.

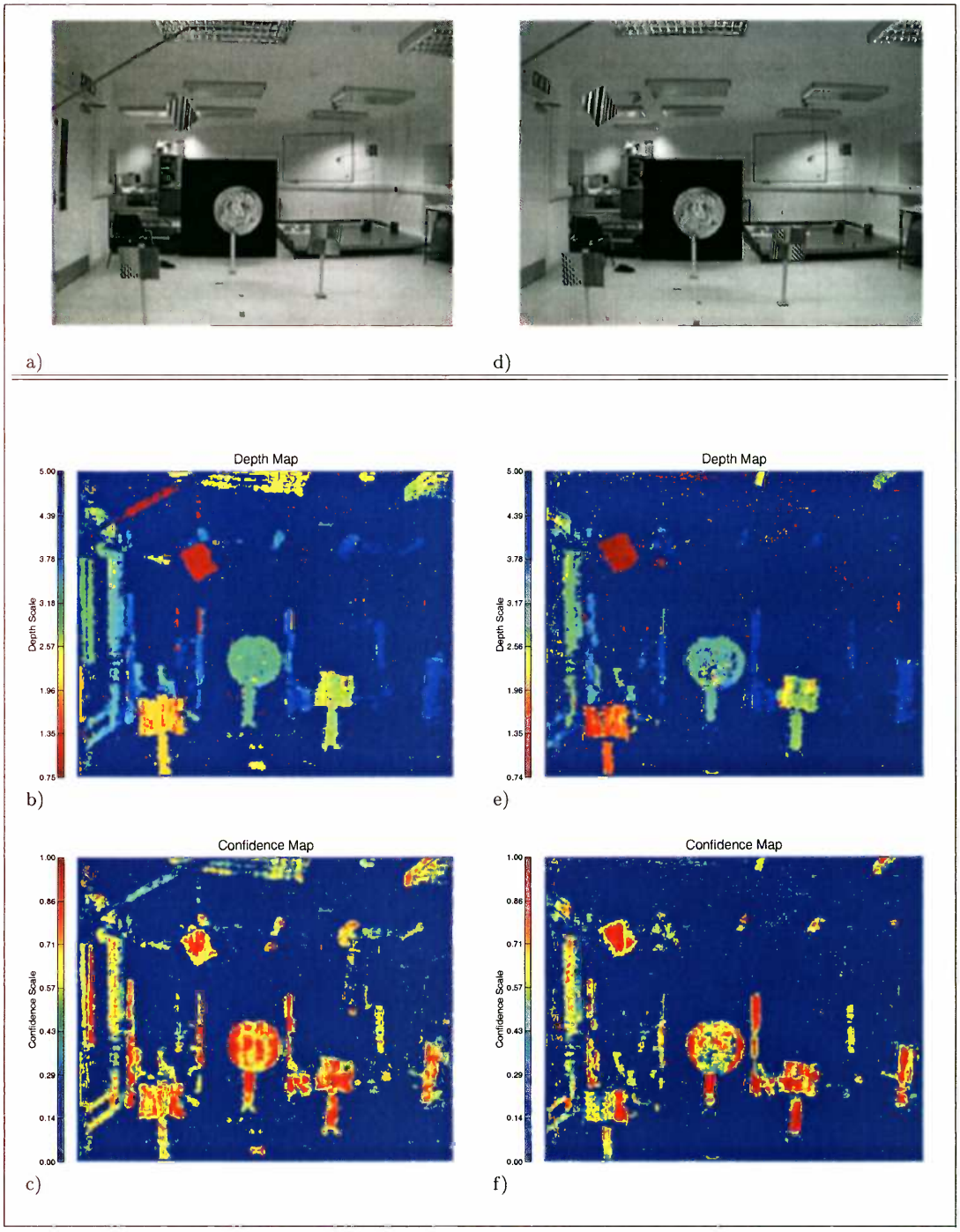


Figure 4.8: Depth estimation examples with real images (large displacement). Two examples are shown. The left column shows the results of the depth estimation procedure for the translational movement strategy. The right column shows the results of the depth estimation procedure for the rotational movement strategy. The top row (images **a** and **d**) shows examples of the central frames of each sequence. The second row (images **b** and **e**) shows estimated the depth map for each case, and the third row (images **c** and **f**) shows the respective confidence maps. The depth maps shown have already been thresholded, eliminating those measurements whose confidence is lower that 0.5 by setting them to the maximum detectable distance (5 m). 640×480 images were used.

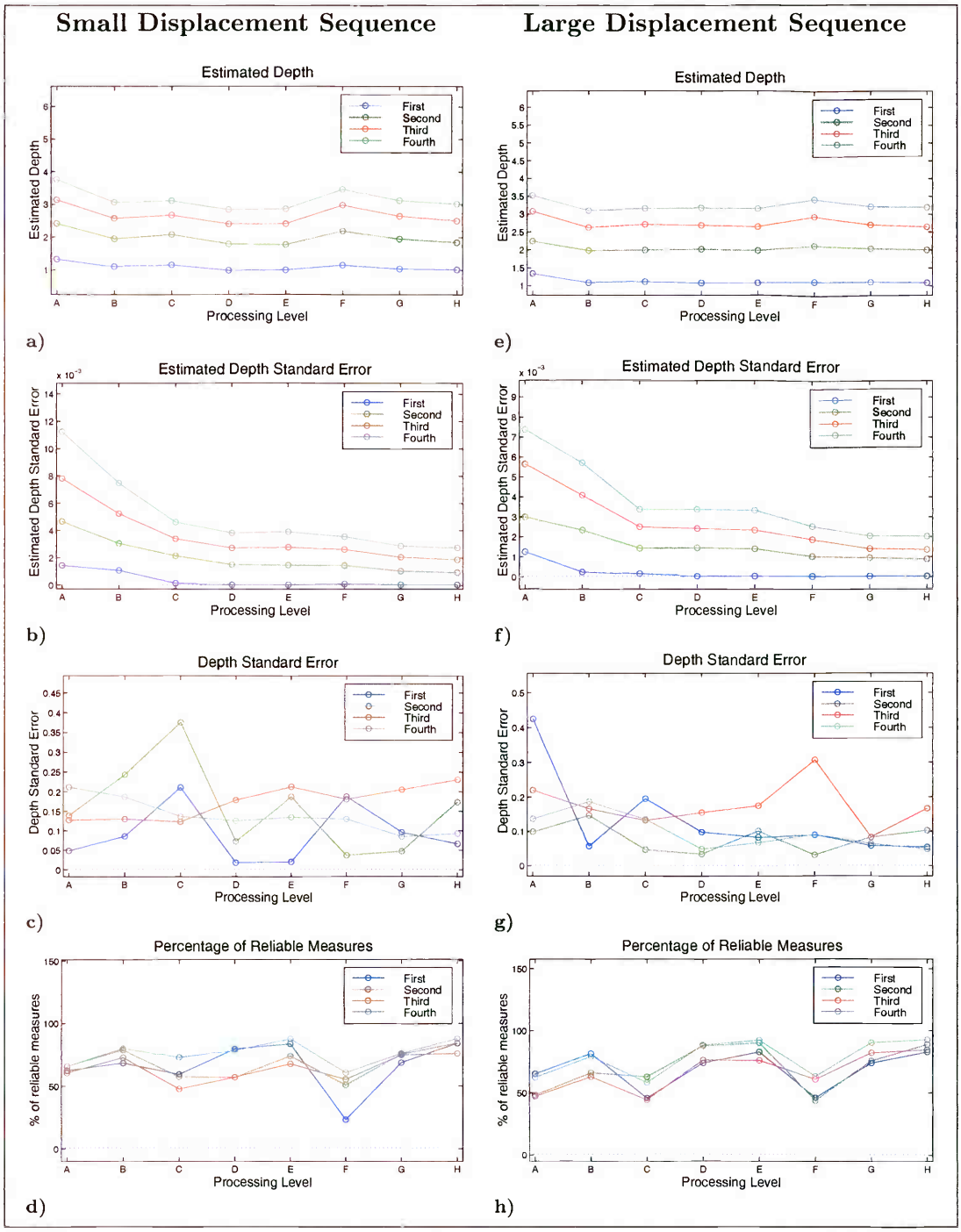


Figure 4.9: Depth estimation sensitivity on real images, translational case: Each row shows the mean estimated depth, the estimated and actual depth standard error, and the percentage of reliable measures obtained from a sequence of images, using a translational camera movement. The positions along the abscissa on each graph indicated the step within the recursive depth estimation procedure: **A**, **C** and **F** correspond to the estimation from a single pair of images on each level (no mixing). **B**, **D** and **G** correspond to the mixture of the two symmetric maps on each level and **E** and **H** to the recursive mixing of the maps produced on the current and previous levels.

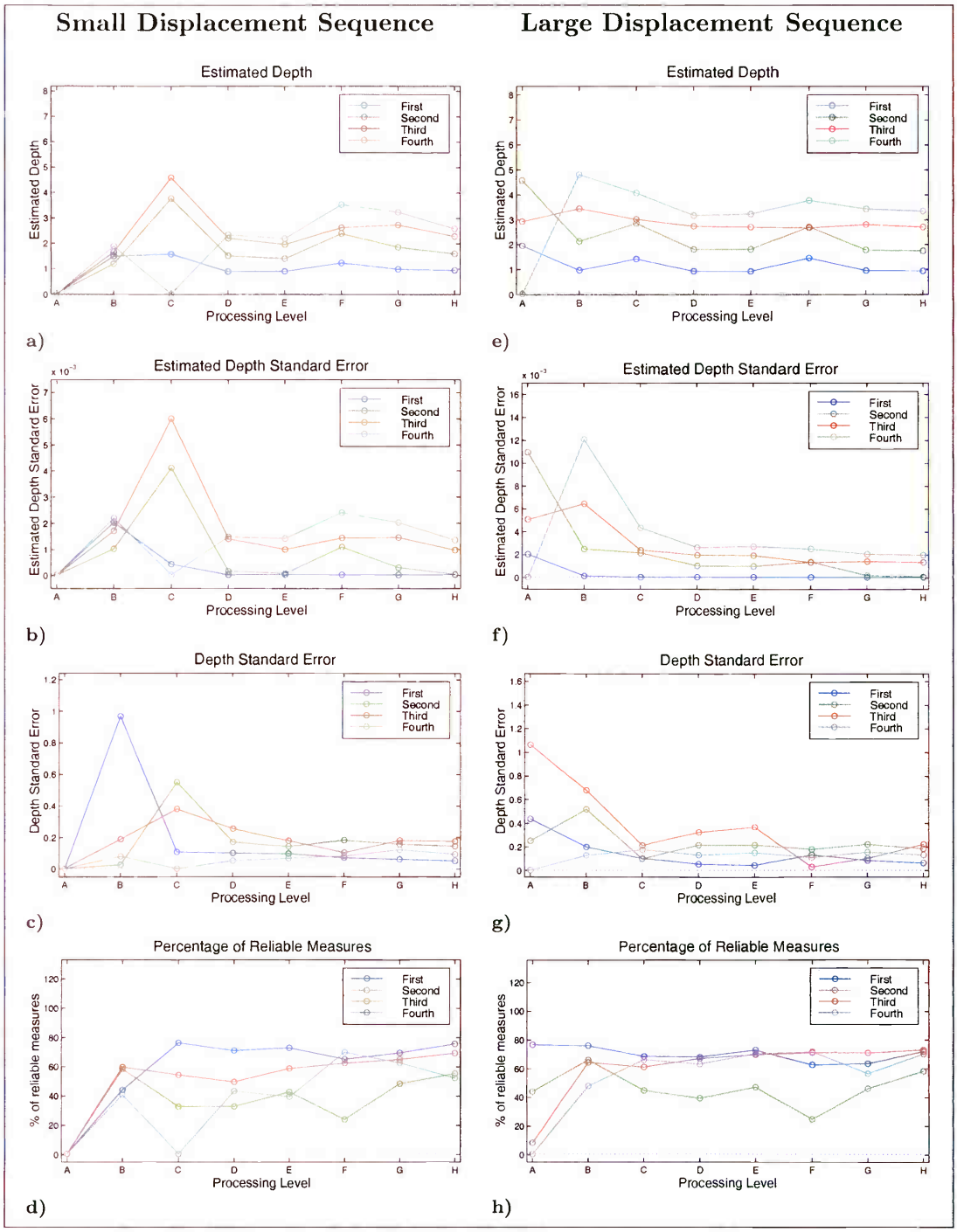


Figure 4.10: Depth estimation sensitivity on real images, rotational case: Each row shows the mean estimated depth, the estimated and actual depth standard error, and the percentage of reliable measures obtained from a sequence of images, using a rotational camera movement. The positions along the abscissa on each graph indicated the step within the recursive depth estimation procedure: **A**, **C** and **F** correspond to the estimation from a single pair of images on each level (no mixing). **B**, **D** and **G** correspond to the mixture of the two symmetric maps on each level and **E** and **H** to the recursive mixing of the maps produced on the current and previous levels.

Chapter 5

Searching for faces:
Constraints from situatedness

5.1 Introduction

In traditional face detection implementations an image exists only as a 2-D grid of intensity values, where certain patterns present in the image are classified as being faces. Although by experience we know that under normal circumstances human faces in an image can only be found attached to persons present in the scene or as pictorial representation of human faces (paintings, photographs or sculptures), most of the algorithms rely only on the face classifier for the location process, ignoring the wealth of clues — given the structured nature of the information found in natural images — that could be used to simplify the task, mainly by guiding the search; in the traditional approach the algorithm's behaviour would be the same if the image is uncorrelated (noise) or if the information in it is structured, as happens in real images.

The main difference between the methods for face detection considered earlier and the one proposed here is the fact that we consider the observer to have a direct relationship with the analysed scene. This relationship is described in two ways:

- A raw 3-D description of the scene inhabited by the observer (depth map). Given that we have a monocular observer, the depth map arises from the interaction of the observer with the world.
- The relationship between the scene and the observer is explicitly known *i.e.*

the orientation and position of the ground plane is known with respect to the observer. In our case the estimation of these parameters is easy, since the observer is a camera mounted on top of a mobile robot, which maintains a constant posture with respect to the ground plane.

Once the relationships between the observer and the scene are known, a model of the subject and its relationship with the world is used to define a set of constraints for the search space.

This chapter is organised as follows: first a model of the subject (adult human beings) is defined, and a formal definition of the search space is given. The next two sections describe how we constrain the search space. The next section deals with how the search is actually performed. Finally, some experiments are carried out followed by a discussion of the proposed approach.

5.2 Preliminaries

This section deals with two main issues: how we represent the scene and the search space, and which is the most appropriate description of the subject and its relationship with the environment.

5.2.1 Scene representation

As was explained in chapter 3, in order to achieve scale independence for the face detection process, the scene is represented with a Gaussian pyramid, where the lowest level represents the original image, and the subsequent levels are scaled-down copies of that image.

Each level in the pyramid is then represented by an image $\mathbf{I}$, which is defined as a set of "pixels", each of them characterised by a quadruple (x, y, i, d), where (x, y) are the pixel coordinates in the image plane, while (i, d) are measurements of the light intensity and depth of the scene "point" which is projected at (x, y) on the image plane. The search space is then composed of the union of the pixel sets of each level. In order to include the scale information in the definition of the set, we define it as the

set **S** of quintuples (x, y, l, i, d) where l is the pyramid level.

Pattern recognition methods for face detection do not deal with face occlusion, and it is not expected that a face would be detected if it is partially occluded; because of this the search space can be constrained in such way that the patterns that fall outside the image boundaries at each pyramid level should not be analysed. Therefore the number of positions to analyse on each level is given by $(X_i - P_{sz} + 1)(Y_i - P_{sz} + 1)$ where (X_i, Y_i) are the dimensions of the image on level i of the pyramid and P_{sz} is the pattern size.

The number of levels of the pyramid depends on the scaling factor between levels, the pattern size and the smallest side of the original image. The image on top of the pyramid (the smallest one) should have its smallest side no smaller than the side of the pattern we are looking for, in our case the face pattern. The number of levels is then given by:

$$n = \log_f \frac{P_{sz}}{s_d} + 1, \quad s_d = \min(X_0, Y_0)$$

where f is the scaling factor between levels in the image pyramid. Since the number of pyramid levels should be a natural number, in practice the number of levels defined above is rounded towards zero.

The size of the search space depends only on the scaling factor f and the size of the original image. The size of the search space is given by

$$\begin{aligned} N &= \sum_{i=0}^{n-1} (f^i X_0 - P_{sz} + 1)(f^i Y_0 - P_{sz} + 1) \\ &= \sum_{i=0}^{n-1} (X_0 Y_0 f^{2i} - (P_{sz} - 1) f^i (X_0 + Y_0) + (P_{sz} - 1)^2). \end{aligned}$$

Since we know that $\sum_{i=0}^{n-1} ak^i = a(1-k^n)/(1-k)$ we can express the previous equation as:

$$N = \frac{X_0 Y_0 (1 - f^{2n})}{1 - f^2} - \frac{(P_{sz} - 1)(X_0 + Y_0)(1 - f^n)}{1 - f} + n(P_{sz} - 1)^2 \qquad (5.1)$$

As expected, the size of the search space N holds a linear relationship with respect to the image size, and it extends in three dimensions, *i.e.* for a face to be located in the image, three parameters have to be found: the (x, y) coordinates in the image plane and the scale of the face.

5.2.2 Subject Model

A model of the subject is necessary in order to constrain the search for faces in the image plane, for example, the location of a face in the image can be constrained by the knowledge of how the observer and the scene and the scene and the subject relate. The relationship between the camera (observer) and the world is given in terms of its relationship with the ground plane, *i.e.* the external camera parameters, while the relationship between the subject and the world is defined by two basic assumptions:

- The subject will always be found standing/supported on the ground plane.
- The subject orientation is perpendicular to the ground plane.

The ground is the only reliable medium through which the observer and the subject relate, and constraints can be derived from this fact.

On the other hand, since we know that the relationship between the face scale in the image plane and a face in the scene is ruled by the projective equation, a model of the size of a human face can be used to constrain the range of scales where a face might be found in the scene.

The subject model then consists of the characterisation of the height and the face size of the adult subjects, and it is based on anthropometric data.

Jürgens et al. (1990) published an international overview of body measurements from an ergonomic point of view, based on a compilation of different anthropometric studies published worldwide.

In their study they found that in order to properly represent the wide variation of body measurements it is necessary to classify the subjects into large and small types, each type composed of different ethnic and gender groups[1].

In order to include the extreme measurements of the world population, each type was characterised by the limiting values (the 5^{th} and 95^{th} percentiles) of the extreme populations of each type, defining the mean of each type as the midpoint between

[1] *e.g.* the small type is composed of the male and female population from South America, and the female population from Southern Europe, Northern Asia, Japan, Africa, India and South-East Asia.

these extreme values. The figure 5.1 shows an example of the distribution of the world population height for the small and large types.

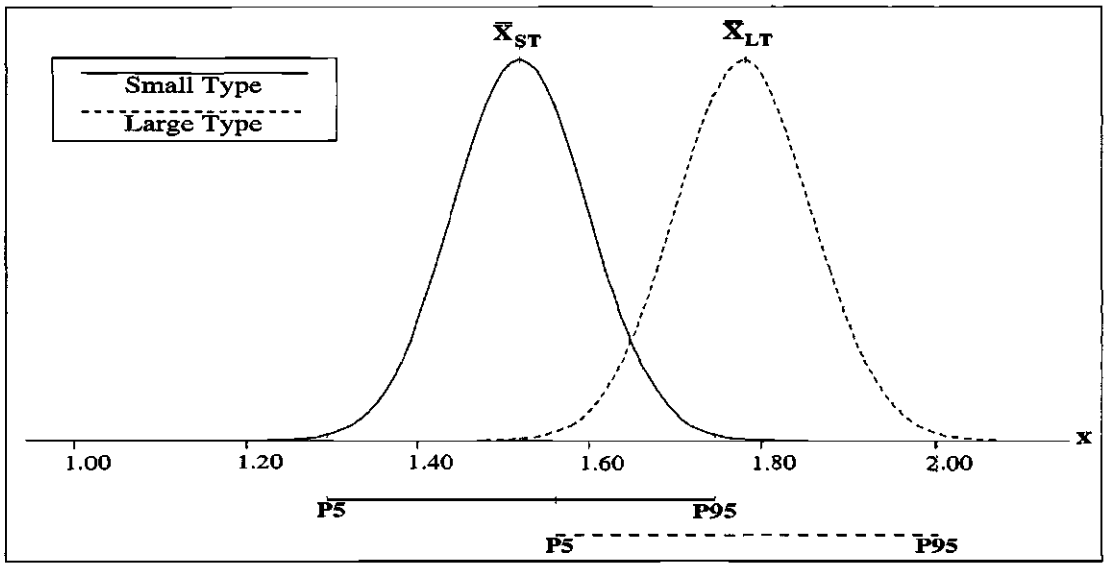


Figure 5.1: World population stature model for large and small type showing mean and extreme values [*Adapted from* Jürgens et al. (1990)].

The closest measurements presented in their study that would help us to characterise the face size are the head measurements: head length, breadth and circumference. In the work presented here, it was decided to use the human head breadth as a good approximation to the size of the human face: as was pointed out before, the face patterns used to train the classifiers are represented as a 20×20 intensity values grid, where external head features, such as hair or ears, are eliminated from the sides of the pattern.

The mean and standard deviation of the small and large type distributions for height and head breadth are given in the following table.

Feature	Small Type		Large Type	
	Mean	Std. Deviation	Mean	Std. Deviation
Height	1.50 m.	0.079 m.	1.78 m.	0.079 m.
Head Breadth	0.135 m.	0.009 m.	0.16 m.	0.006 m.

5.3 Constraining the search space

In this section a definition of the different constraints that can be set upon the search space, given a subject model and the explicit relationship between the observer and the scene, will be discussed.

The kind of constraints that we expect to find can be classified into two kinds: constraints of scale, which rely on the scene depth, and constraints in location, which emerge from the geometric relationship between the observer, subject and the scene.

5.3.1 Constraints due to scene depth

Depth in the scene can be recovered by different methods, as explained in the brief review given in chapter 4; in our case we chose a kinetic depth algorithm that estimates the depth from a sequence of images grabbed while the camera is moved in a controlled way. The proposed algorithm works under the assumption that the only source of change on the image plane is produced by the camera's movement, therefore we are only able to obtain reliable depth information from static scenes. The main consequence of this is that we are forced to estimate the depth of the scene under controlled conditions, *i.e.* not allowing anything to move in the scene while capturing the sequence of images. This is an important drawback, since we will not be able to estimate the depth in real-time. In the approach presented here depth estimation is done off-line, under the assumption that there will be no changes in the position of the camera or in the general structure of scene. Nevertheless, using a similar approach to the one described in the previous chapter it would be possible to estimate the depth in the scene on a real-time basis if a binocular system is used.

The main difference between these two implementations is the availability of the depth information; this determines the extent of the constraints imposed on the search space due to depth information. Two cases can be then defined:

- The first case is the general approach, where the depth information can be estimated at a similar rate to the capture rate of grey-level images. The depth map can then apply to both the static scene's structure and any moving objects.

- The second case is a particular case of the general problem, under the constraint that the scene has to be static. The depth map is estimated at the beginning of the search process and only represents the depth of the static structures of the scene.

The relationship between scale and depth

Since we are interested in the localisation of faces of individuals present in the scene, and given the small variation of human face sizes ($\pm$ 0.03 m, see page 104), it is clear that the range of scale of the face patterns that can be found in the scene depends mainly on the distance between the camera and the subject. The distance along the **Z** axis between a face in the scene and the camera, and the size of the projected pattern on the camera's image plane, is described by the projective equation 4.1.

The optimal detection of a face pattern will happen on a single level of the Gaussian pyramid, the one that is closest to representing the face as a 20×20 pattern. Since the face classifier accepts certain degree of variance in the input pattern, we can say that a face in the scene at a particular distance from the scene will only be detectable at a restricted set of levels of the image pyramid, or that each level in the image pyramid covers a certain depth range in the scene where a human face might be found.

The relationship between the face scale (expressed in term of pyramid levels) and the distance between the camera and the subject can be expressed as a scaled version of equation 4.19, which expresses the depth range where a face can be detected on each level of the image pyramid. In chapter 4 we were only interested in the maximum distance at which a face might be found in a full scale image, while in this case it is necessary to describe the range of distances covered by each level. Taking into account the variance in the face size, $\sigma^2_{f_s}$, we can define the scale-depth relationship as:

$$\bar{Z}(n) = f^n F \frac{\bar{f}_s}{P_{sz}} \quad , \quad \sigma^2_{Z(n)} = f^{2n} F^2 \frac{\sigma^2_{f_s}}{P^2_{sz}} \tag{5.2}$$

where n is the pyramid level, $\bar{f}_s$ is the average face size, F is the focal length of the camera, f is the Gaussian pyramid scaling factor[2].

[2] in our case equal to 5/6.

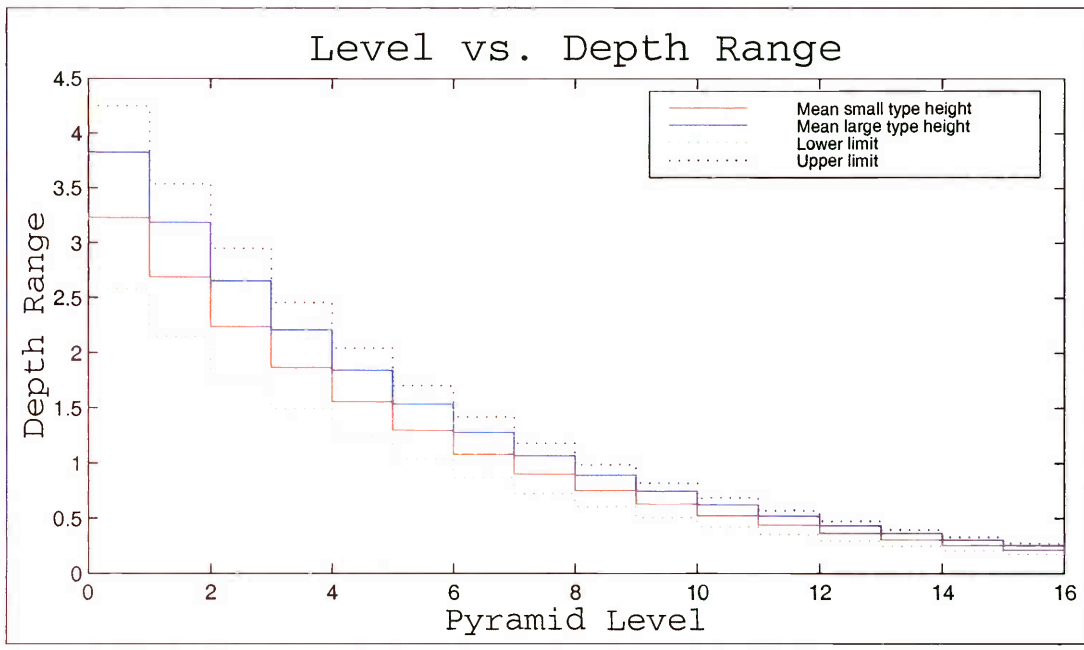


Figure 5.2: Depth range *vs.* scale: this plot shows the maximum and minimum distance a human face might be found on a particular level on the image pyramid.

In order to consider the extreme cases of the population the range of camera-face distances for a particular level can be defined by the interval

$$[Z_s(n) - 3\sigma_{zs}, Z_l(n) + 3\sigma_{zl}]$$

where $Z_s(n)$ and $Z_l(n)$ are the mean distances at the level n for the small and large human type, and σ_{zs} and σ_{zl} are their respective standard deviations. A plot that shows the mean and extreme depth values for different human types and scales is shown in figure 5.2.

It is important to note that there is a certain degree of overlap between the depth ranges covered by each pyramid level; this overlap together with the face pattern variance inherent to the pattern classifier, helps to explain the multiple detections reported in the literature for this kind of classifier.

Using depth information to constrain the search space: the general case

In the general case the depth map not only gives information about the static structures present in the scene, but also depth information about the subjects in it. The

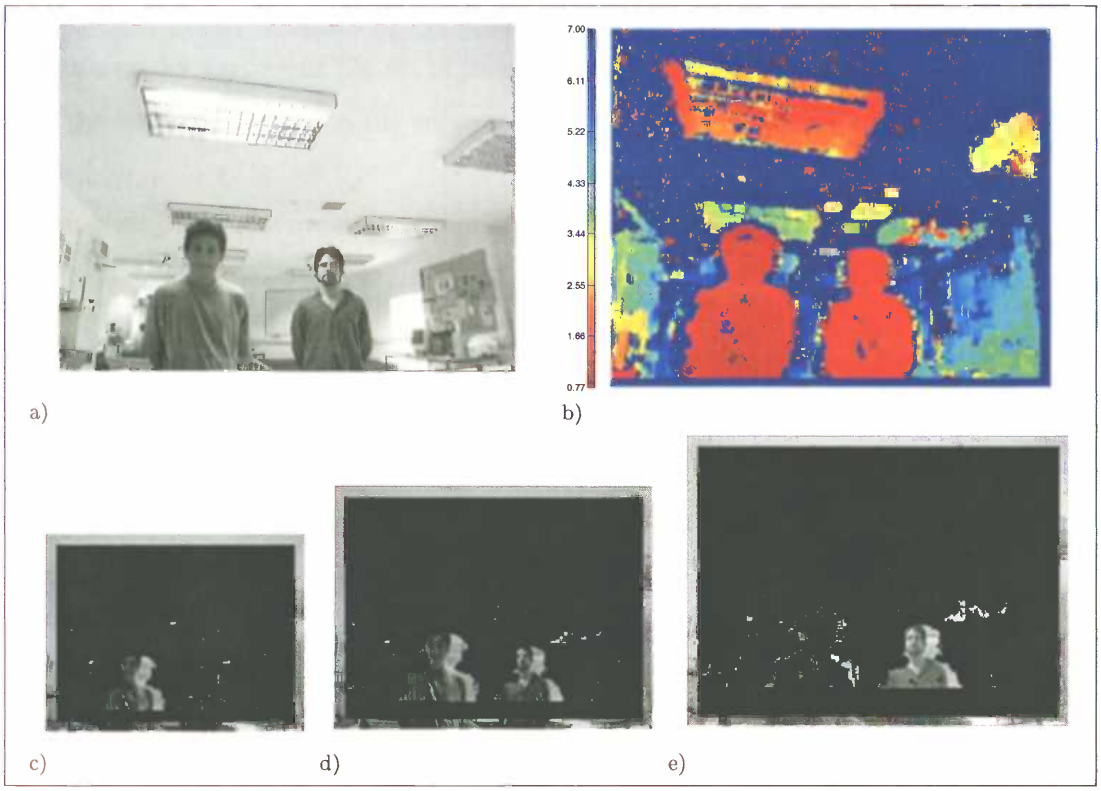


Figure 5.3: Constraining the search space using scene depth information: The general case. The first two images show the grey-level (**a**) and the depth (**b**) information of a scene. Images (**c**), (**d**) and (**e**) show the pyramid levels 13, 12 and 11 respectively, where the blacked-out areas corresponds to the elements eliminated from the search space.

immediate consequence of this is that the uncertainty in depth is reduced since we have an estimation of how far every object in the scene is from the camera, and also how large human faces tend to be; this information is useful for discarding those elements of the search space where a face can not be found.

The search space S can be then constrained by eliminating those elements that are outside the level depth interval defined above. The constrained search space S_{dg} is defined as follows:

$$S_{dg} = \{p \in S \mid d(p) \in [Z_s(l(p)) - 3\sigma_{zs}, Z_l(l(p)) + 3\sigma_{zl}]\} \tag{5.3}$$

where $d(p)$ represents the depth information contained in the quintuple p, and $l(p)$ is the pyramid level of the quintuple p[3]. An example of how the search space can be constrained is shown in figure 5.3[4].

[3] In order to refer to the component $\mathcal{Q}$ of the tuple p, the notation $\mathcal{Q}(p)$ is used.

[4] The examples of the general case shown here were obtained by asking the subjects to remain static.

As can be seen in figure 5.3, the main effect of constraint using the scene depth is to enable us, by narrowing the set of possible scales where a face can be found, to "focus" on the surfaces present in the scene.

Using depth information to constrain the search space: the particular case

In the particular case the depth map is static, it gives a rough representation of the different structures present in the scene, and it is estimated from a scene which only contains static objects, such as furniture, walls, floor and ceiling. Since the subjects will only be visible if they are situated in the space between the scene's structures and the observer, we can eliminate those areas of the pyramid where it is unlikely that a face will appear (the areas behind the scene's structures); in other words, we are making explicit the areas in the image where a face will be occluded.

The scene depth information can be used now to detect those positions in the image pyramid where an occlusion will happen, and eliminate them from the search space. The constrained search space S_{dp} is then defined as follows:

$$S_{dp} = \{p \in S \mid d(p) > Z_l(l(p)) + 3\sigma_{zl}\}. \tag{5.4}$$

In the particular case, the areas in the search space that are constrained by the depth information are more limited than in the general case since the depth interval where a face might be found is open on one side. An example of the constrained search space can be seen in figure 5.4.

5.3.2 Geometrical constraints

Another other source of constraint is based on the relationship between the scene and the people who inhabit it. Two main aspects of this relationship are useful for us and can be defined with two main assumptions:

- People in the scene maintain an upright position.
- People in the scene are supported by a surface underneath them.

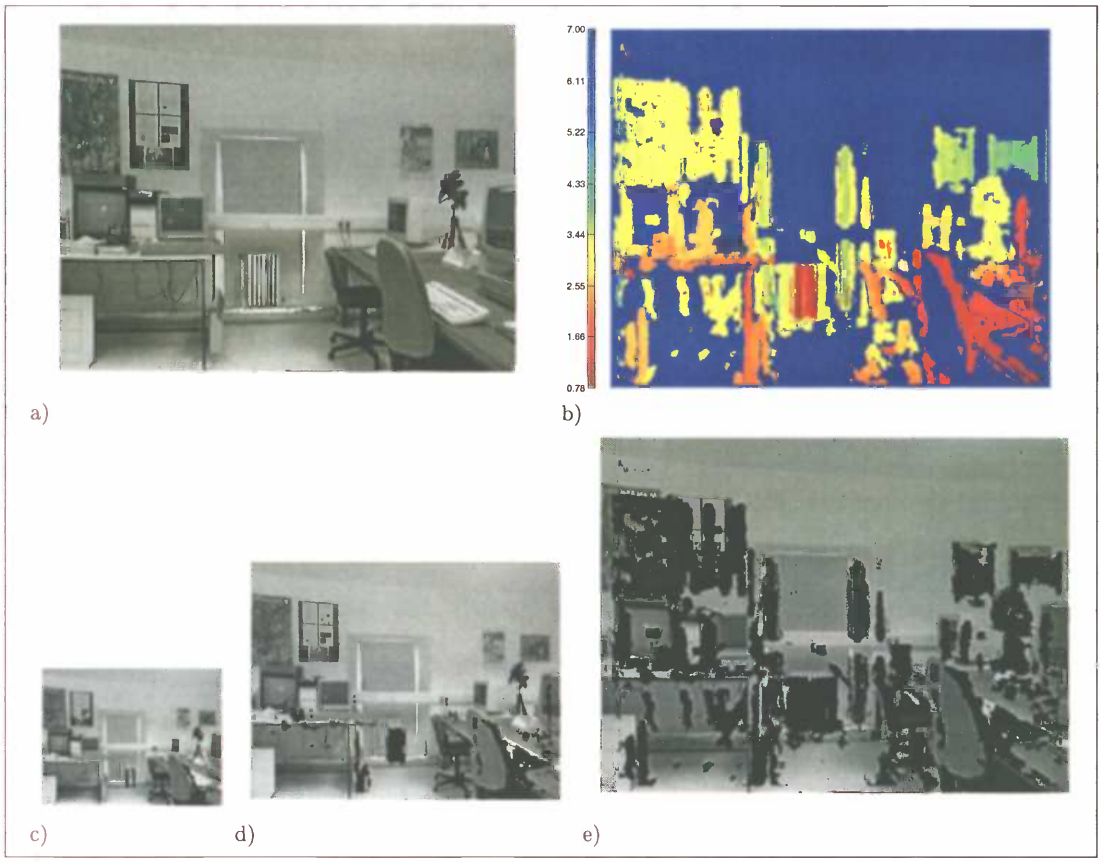


Figure 5.4: Constraining the search space using scene depth information: The particular case. The first two images show the grey-level (**a**) and the depth (**b**) information of a scene. Images (**c**), (**d**) and (**e**) show the pyramid levels 6, 3 and 0 respectively, where the blacked-out areas correspond to the elements eliminated from the search space.

A direct consequence of the first assumption is that the faces to be found in the scene also have an upright position, which is a common requirement in face detection algorithms; most of the classifiers reviewed in chapter 2 only consider a quite limited variation on the face rotation angle.

The second assumption is useful because it allows us to constrain the range positions where a face might be found in the vertical direction, using an anthropometric model and horizontal surfaces model of the scene.

The ground plane

We model the ground plane as an horizontal flat surface, a reasonable assumption for an indoor environment. The surface is expressed by the following equation:

$$aX + bY + cZ = d, \quad d > 0 \tag{5.5}$$

where $\mathbf{n} = [a, b, c]^\top$ is an orthonormal vector to the surface, which is pointing away from the camera, and d is the distance from the surface to the camera viewpoint, where the coordinate frame is centred on the camera viewpoint as in figure 4.1.

Since our representation of the scene is through the two-dimensional image captured by the camera, we need to express the previous equation in terms of its projection on the image plane. Substituting the projective equation 4.1 into 5.5 we obtain:

$$\frac{Z}{F}\Big(ax + by\Big) + cZ = d. \tag{5.6}$$

We are interested in finding those areas of the image that correspond to a position in the scene that are inside the interval where it is likely to find a face given the anthropometric model. For this reason we need to find the correspondence between a position in the image plane and its minimum distance to the ground plane at a particular distance Z from the camera.

Under the assumption that the floor is represented by an horizontal surface, the minimal distance between a point in the scene and the plane is the length of a vector parallel to the orthonormal vector $\mathbf{n}$ drawn between the point and the floor; this minimal distance is expressed as follows:

$$H = d - \frac{Z}{F}\Big(ax + by\Big) - cZ \tag{5.7}$$

Taking into account that we represent a portion of the scene in terms of elements of the search space, we can express the same equation as a function of the element p of the search space. The distance between a point in the scene, characterised by the element $p \mid p \in S$, and the floor is therefore given by the following equation:

$$H(p) = d - \frac{Z_{mx}(p)}{f^{l(p)}F}\Big(ax(p) + by(p)\Big) - cZ_{mx}(p) \tag{5.8}$$

where $Z_{mx}(p)$ is the maximum range at which a face can be found on the level $l(p)$ of the image pyramid (namely $Z_{mx}(p) = Z_l(l(p)) + 3\sigma_{zl}$).

The interval where a face might be found is defined in terms of the anthropometric model described before. Taking into account the small and large types, the interval is

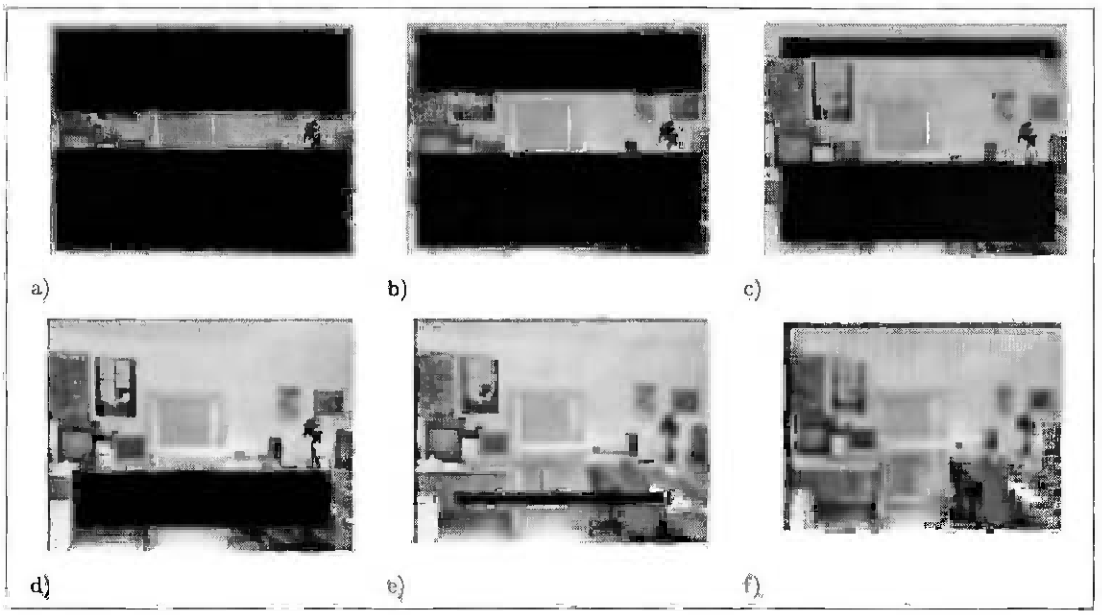


Figure 5.5: Constraining the search space using geometric information: images (a)-(f) respectively show the levels 0, 3, 6, 9, 12 and 15 of the image pyramid where the blacked-out areas correspond to the elements eliminated from the search space using geometric constraints.

given by

$$[h_s - 3\sigma_{hs},\ h_l + 3\sigma_{hl}]$$

where h_s, h_l are the mean height of the small and large human types respectively, and σ_{hs}, σ_{hl} their respective standard deviations.

Once we have defined the interval, we can constrain the search space by eliminating those elements that lie outside the interval. The constrained search space S_h is then defined as follows:

$$S_h = \{p \in S \mid H(p) \in [h_s - 3\sigma_{hs},\ h_l + 3\sigma_{hl}]\} \tag{5.9}$$

The final search space S_s is defined then by the intersection of the sets S_{dg} and S_h for the general case, and S_{dp} and S_h in the particular case. Examples of the fully constrained search space are shown in figures 5.6–5.8. Figures 5.6 and 5.7 are examples of the particular case with different camera orientations (a tilt angle of 0 and 20 degrees respectively), while figure 5.8 is an example of the general case with a camera tilt angle of 10 degrees.

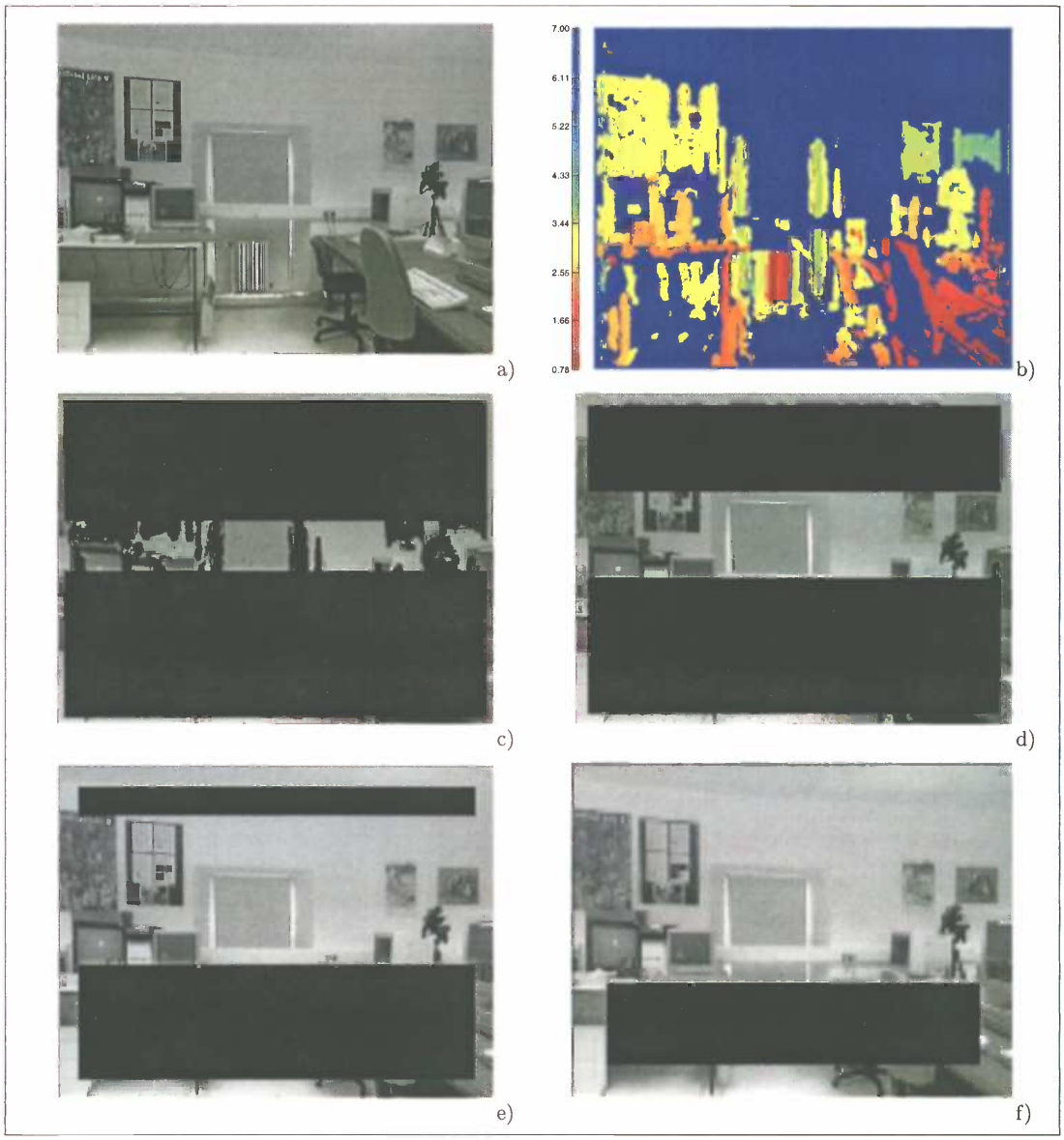


Figure 5.6: Constrained search space using both depth and geometric information (particular case): images (**a**) and (**b**) show the grey-level and depth information of the scene while images (**c**)-(**f**) show respectively the levels 0, 3, 6, and 9 of the image pyramid where the blacked-out areas corresponds to the elements eliminated from the search space; the camera orientation is parallel to the ground plane.

5.4 The Search

Our main assumption is that face location is carried out by a situated observer, which has a direct relationship with the world, and it is therefore expected that the process should have a real-time response. In traditional face detection approaches the search is considered to be completed when the whole search space has been tested. For example, some approaches (Vaillant et al., 1994; Rowley et al., 1998) including the one described

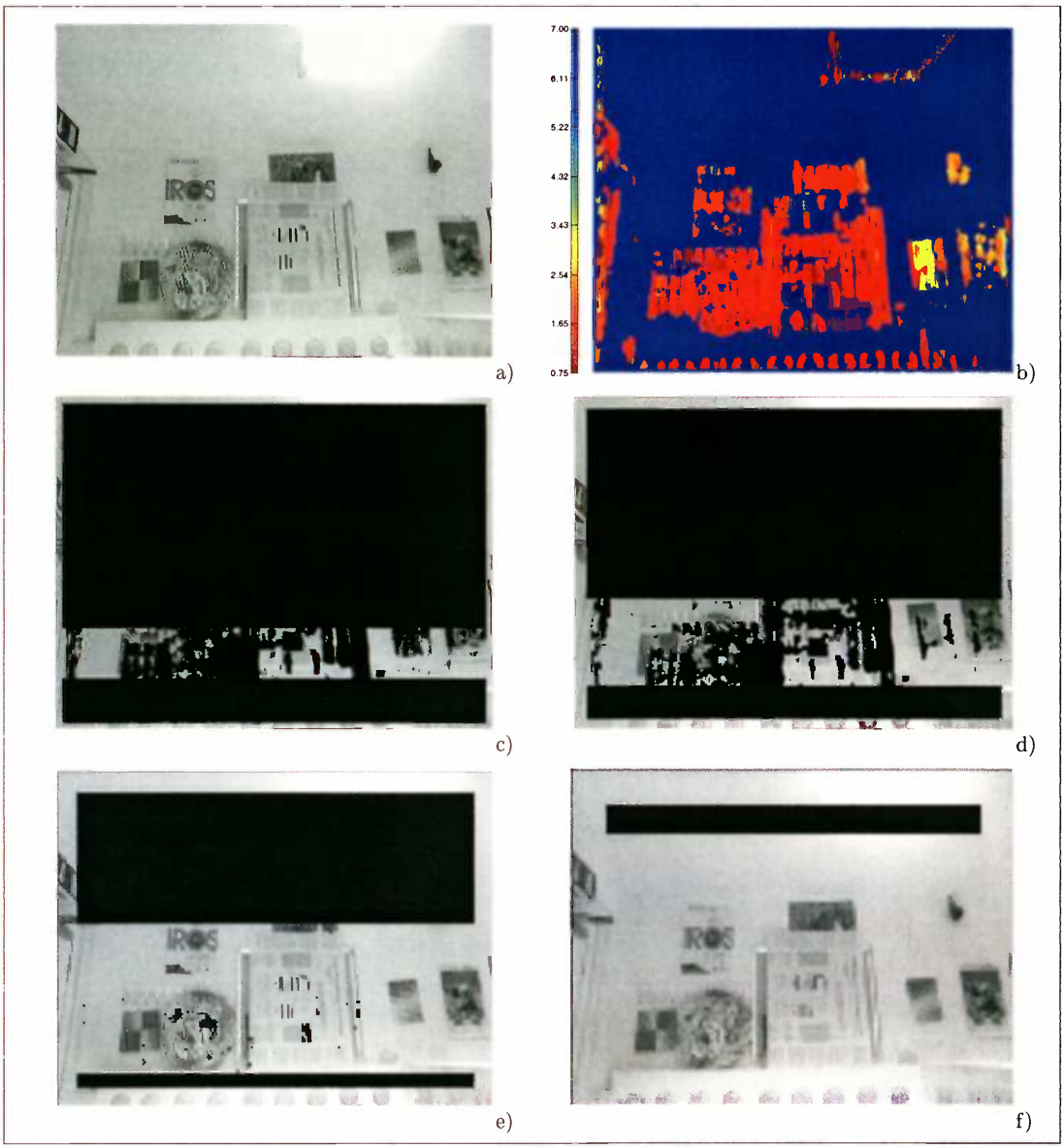


Figure 5.7: Constrained search space using both depth and geometric information (particular case): images (**a**) and (**b**) show the grey-level and depth information of the scene while images (**c**)-(**f**) show respectively the levels 0, 3, 6, and 9 of the image pyramid where the blacked-out areas corresponds to the elements eliminated from the search space; the camera's tilt angle is 20° above the ground plane.

in chapter 3 have an extra stage where multiple detections are merged after the search process has been finished, which implies that in order to obtain definitive results every element of the search space has to be tested.

While this is not a big problem for off-line face detection systems, in real-time systems it would be advantageous to use a more intelligent kind of process — one that allows us to direct the search to those relevant portions of the image and to obtain definite

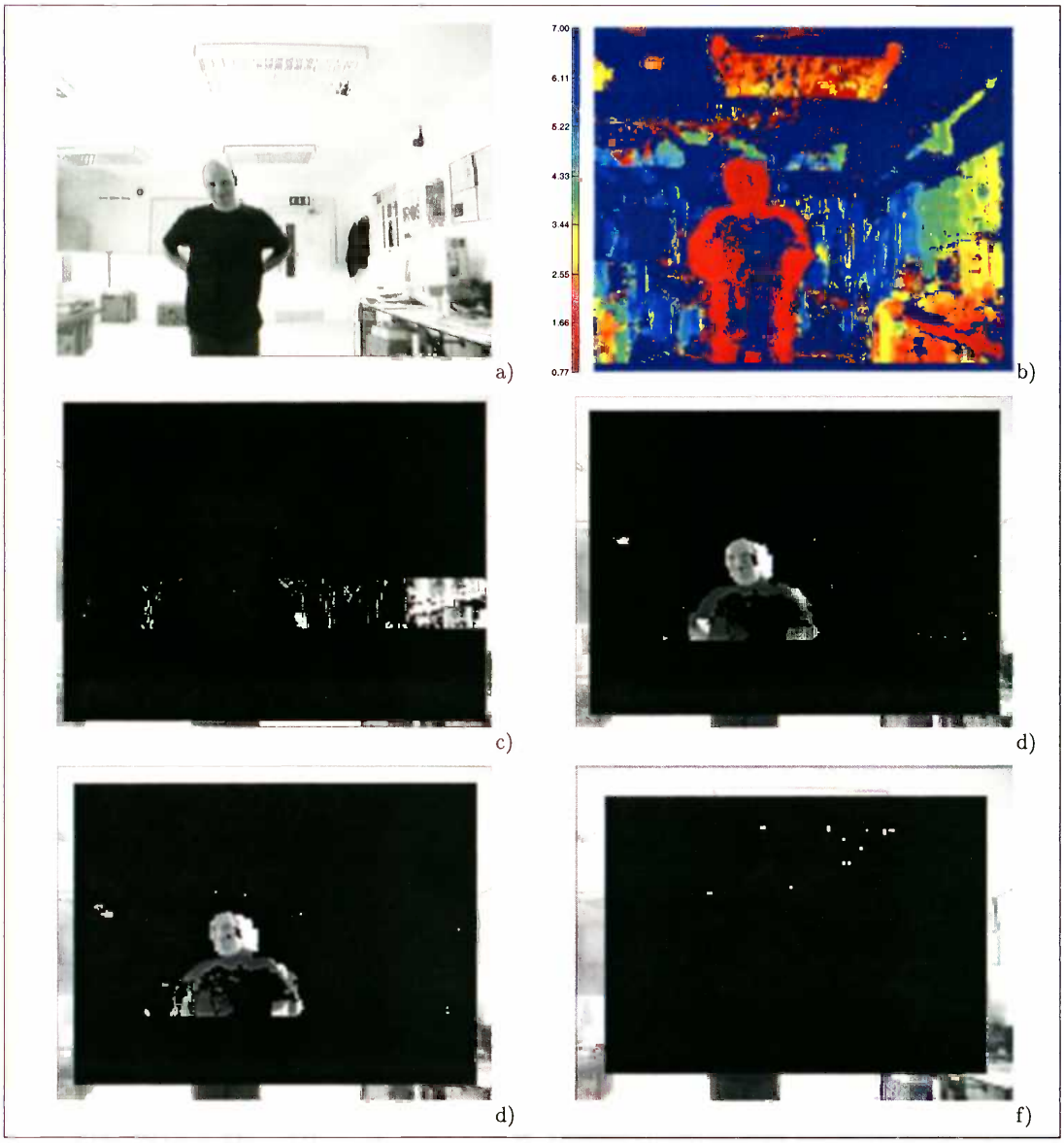


Figure 5.8: Constrained search space using both depth and geometric information (general case): images (**a**) and (**b**) show the grey -level and depth information of the scene while images (c)-(**f**) show respectively the levels 0, 4, 5, and 8 of the image pyramid where the blacked-out areas corresponds to the elements eliminated from the search space; the camera's tilt angle is 10° above the ground plane.

partial results while the search is still under way.

In this section we describe a strategy for performing a guided search. It relies mainly on knowledge of the observer-scene-subject relationship and on the way faces in the scene tend to generate a cluster of hypothetical detections in the image pyramid, as described in section 3.5.

5.4.1 The search sequence

In non-situated approaches for face detection the easiest way to implement the search for faces in the image pyramid is a sequential approach because, given that the search is not considered complete until every element of the search space has been tested, the order of the search does not affect the final result. The use of a sequential search eliminates the need to explicitly represent the search space as in section 5.2.1, since the location and scale of each pattern to be tested are represented implicitly by the indices that control the algorithm loops, and the intensity and depth measures are stored in the image pyramid.

In section 5.3.2, the subject's height probability distribution was used to find those areas of the image plane at different scales where it is unlikely that there is a face; the same model can now be used to direct the search by using such a probability distribution to construct an index to control the search, *i.e.* the positions on the image pyramid with the highest probability will be tested first.

The probability function $\mathcal{P}(H)$ used to rate the elements of the search space is defined by a bimodal Gaussian distribution depending on the parameters of the small and large type height model, and is given by:

$$\mathcal{P}(H) = K_p(e^{-\frac{1}{2}(\frac{H-h_s}{\sigma_{hs}})^2} + e^{-\frac{1}{2}(\frac{H-h_l}{\sigma_{hl}})^2}) \tag{5.10}$$

where K_p is a normalisation factor to ensure that the area under $\mathcal{P}(H)$ is equal to one.

The probability corresponding to each element is obtained by calculating the area under the distribution interval $[H(p) - D_h(p), H(p) + D_h(p)]$, where $2D_h(p)$ is the size of the portion of the scene represented by the element p of the search space, defined by:

$$D_h(p) = \frac{Z_{mx}(p)}{2F}. \tag{5.11}$$

The probability assigned to each element depends solely on the height of the scene portion represented by such an element with respect to the ground plane ($H(p)$), and the area of the scene represented by the element ($D_h(p)$); because of this it is expected that there will be some areas of the image that will have equal probability. In our case — since the camera baseline is parallel to the ground plane — pixels on any given

horizontal line will be equiprobable. This implies that if we use the probability alone as the basis of the search index, all the elements belonging to a particular row of an image pyramid level will be evaluated sequentially.

This is not the most convenient way to perform the search, for the following reason: elements in the search space that represent the same portion of the scene at different scales will have slightly different probabilities assigned to them. The main consequence of this is that a whole row of elements at a particular level will be tested before proceeding to the next most likely place to find a face, which in turn will be another row. Thus, the way in which the search is conducted will be non-homogeneous, analysing certain portions of the search space before others for no reason whatsoever.

In order to obtain a more homogeneous search, an additional random term is added to the probability measure. The magnitude of this random term is determined in such way that it does not outweigh the probability distribution as the main force driving the search. For this reason it was decided to define it as a fraction of the probability calculated using a value which is the mean of the large and small population mean heights. Therefore the maximum magnitude of the random term $\mathcal{R}(p)$ is defined as follows:

$$\mathcal{R}(p) = 2K_r K_p D_h(p)\mathcal{P}(H_{mean}), \quad H_{mean} = \frac{h_s + h_l}{2} \tag{5.12}$$

where K_r is the scaling factor that determines the importance given to the random term.

The rating value $\mathcal{I}(p)$ assigned to the element p of the search space S is then defined as follows:

$$\mathcal{I}(p) = \mathcal{P}(H(p)) + \mathcal{R}(p)\rho \tag{5.13}$$

where ρ is a uniformly distributed random number in the interval $[-1, 1]$. Figure 5.9 shows the rate values assigned to the search space elements in pyramid level 0.

Once the rating values corresponding to each element of the search space have been calculated, the elements of the search space are sorted in such way that those with the largest rating value will be the first to be analysed. The ordered subspace S_o is then defined as follows:

$$S_o = \{p_i \mid i \in \{1 \cdots n\},\ p_i \in S_s,\ \forall\, j \in \{1 \cdots n-1\},\ \mathcal{I}(p_j) \geq \mathcal{I}(p_{j+1})\} \tag{5.14}$$

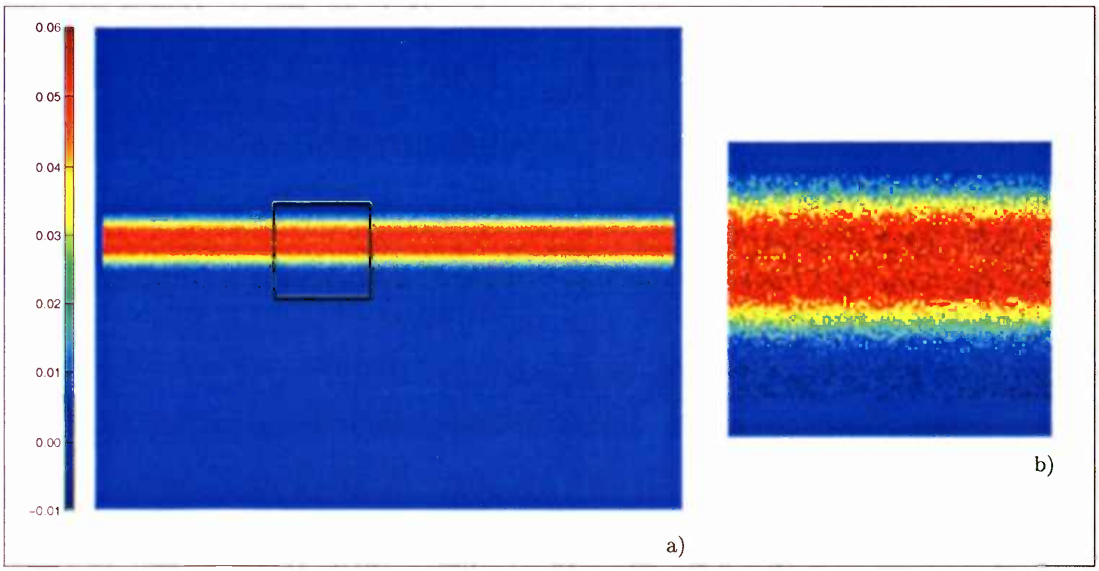


Figure 5.9: Example of the rated search map: image (**a**) shows the rating values used for every position of the search space at the pyramid level 0, image (**b**) shows the zoomed out area marked in image (**a**) (K_r is equal to 0.15).

where n is the number of elements in set S_s.

Another possible way to generate the search index is by using non-replacement sampling on the probability distribution described in equation 5.10: a value is sampled from the constrained height distribution in order to obtain an hypothetical vertical position, while the horizontal and scale values are sampled from two uniform probability distributions. The scale and horizontal values have to be checked first against the constraints derived from the depth information; if they are within the constraints, the corresponding search space element space is estimated, and finally checked with respect to the search index in order to ensure a non-replacement sampling. If such a search element has not been tested, its position is added to search index. The previous procedure is repeated until the search index is complete.

The main problem with the non-replacement sampling procedure described above is that it is computationally expensive, since the sampling is done from a set of three continuous distributions in order to generate a well defined discrete search index, and does not explicitly take into account the constraints derived from depth information; every sample has to be discretised and compared first with the depth constraints – which are by nature discrete – and with the search index. It should be noted that the scheme described in the previous paragraph was successfully used to generate the

test images used to test the approach describe in this chapter (see subsection 5.5.1), a feasible task giving the small number of faces generated for each image.

It is important to note that the method used to generate the search index is equivalent to a partial non-replacement sampling of those elements of the search space which overlap in scale (*i.e.* overlapping image pyramids rows), while using the distribution that describes the subject height as the main source driving the search. Adding a random term to the probability value used to sort the search index and then using this value to sort the search index is equivalent to classifying the search space on groups which overlap in scale and then applying a non-replacement sampling on each group to find the order used to test the search space; each group is ordered according to its mean probability value.

The estimation of the rating values and the ordering of elements of the search space elements is a computationally expensive process, but it can be calculated off-line and it only has to be recomputed in case of translation or rotation of the camera.

5.4.2 Directing the search with evidence

One of the features of the pattern classification approaches to face detection is that faces in the scene tend to be detected more than once; as mentioned in chapter 3, a cluster of detections is normally found in the vicinity of the face. The element p_v is said to be in the vicinity of the optimal detection p_i if the Euclidean distance between the (x, y) components of the quintuplets p_i and p_v is smaller than a threshold τ_d, and the difference between their corresponding scale components (l) is smaller than a threshold τ_l.

In order to obtain a single detection per face in the image, a merging procedure is used; it determines the best detection (in terms of the classifier output) from each cluster as the real detection. As described in chapter 3, the clusters are identified by comparing the detections found in the scene with each other and grouping those which fulfil the vicinity conditions described above. This merging procedure is normally applied once the whole search space has been analysed and, while this is not a disadvantage for off-line face detection systems, it is infeasible for an on-line face

detection system which is expected to work in real-time.

Instead of waiting for the whole search space to be tested in order to start looking for detection clusters, it is proposed in this section to purposefully look for clusters while the search is still under way. For this reason, we divide the search into two modes: the first one uses the search index to determine which is the next element to test, while the second, which is activated when an element is classified as being a face, looks for evidence in the vicinity of the previously detected element to confirm the detection; once the elements belonging to the cluster have been analysed, the first mode resumes, and the search continues until every element of the set S_s has been tested.

The size and shape of the cluster around the optimal detection is determined by the parameters τ_d, τ_l and the Euclidean metric used to compare the position between two elements. In traditional approaches, the use of the Euclidean distance as the criterion to group different detections into clusters during the merging procedure is not a computationally expensive process, since the number of detections to compare tends to be small. Also, the complexity of the merging procedure constitutes only a small overhead when compared with the computational effort needed for the search.

On the other hand, the use of the Euclidean distance for the inverse procedure — the determination of the area covered by the cluster at a particular level of the image pyramid — was considered to add unnecessary complexity to the search process, especially given the importance of the speed of the search process to an on-line face detection system. For practical reasons, it was decided to approximate the circular area ideally drawn by the use of a Euclidean metric with the square in which that circle would be inscribed.

The subset S_v of the search space composed of those elements which are in the vicinity of the optimal detection p_i, is defined as follows:

$$S_v(p_i) = \{p_v \mid p_v \in S_s,\ x(p_v) \in \mathcal{X}(p_i, p_v),\ y(p_v) \in \mathcal{Y}(p_i, p_v),\ l(p_v) \in \mathcal{L}(p_i)\} \quad (5.15)$$

where

$$\begin{aligned} \mathcal{X}(p_i, p_v) &= [x(p_i) f^{l(p_i)-l(p_v)} - \tau_d f^{-l(p_v)}, x(p_i) f^{l(p_i)-l(p_v)} + \tau_d f^{-l(p_v)}] \\ \mathcal{Y}(p_i, p_v) &= [y(p_i) f^{l(p_i)-l(p_v)} - \tau_d f^{-l(p_v)}, y(p_i) f^{l(p_i)-l(p_v)} + \tau_d f^{-l(p_v)}] \end{aligned}$$

$$\mathcal{L}(p_i) = [l(p_i) - \tau_l, l(p_i) + \tau_l]$$

and f is the scaling factor between pyramid levels, which is used in the definition of the intervals $\mathcal{X}(p_i, p_v)$ and $\mathcal{X}(p_i, p_v)$ to scale the optimal detection position $[x(p_i), y(p_i)]$ and the size of the interval determined by τ_d.

Using the search index to control the testing order of the elements in the search space S_s, as described in the last subsection, guarantees that every element in the search space will be tested, and that no element will be tested more than once; when we combine the two search strategies (indexed and the evidence-guided search), the last statement loses its validity, *i.e.* an element tested by the indexed search can be tested again by the evidence-guided search and vice versa.

In order to avoid redundancy during the search process, two new components were added to each of the quintuplets that compose the search space. The role of these elements is to function as binary flags: the first flag (f_t), indicates whether the element has been already tested, and the second one (f_{st}), which is meaningful only when the element has been tested, indicates the outcome of the test.

Before analysing an element of the search space, either during the indexed search mode or the evidence-guided search mode, the flag f_t corresponding to the element being tested is checked: if the flag is unset, which means that the element has not been tested, the algorithm proceeds to extract the pattern, classify it and set the flag f_t. If the flag is set, the algorithm checks the flag f_{st} to know if the pattern corresponding to that element was previously classified as a face; if this is the case, the algorithm ignores the flag f_t and proceeds to extract and test the corresponding pattern in order to obtain the classifier output value.

If the pattern was classified as belonging to the non-face class the search continues its course. On the other hand if the pattern is classified as a face, the path followed depends on whether the face pattern was detected during indexed search mode or evidence guided search. In the former case the evidence-guided search is triggered, which then starts searching for another detection in the vicinity of the element p_i, (*i.e.* $S_v(p_i)$). In the latter case, the location components of the element p_v ($[l(p_v), l(p_v), l(p_v)]$) and the output of the classifier (O_{net}) are stored in a temporary list ($\mathcal{L}$).

```
for every p_i ∈ S_o
begin
   if f_t(p_i) = 0 or (f_t(p_i) = 1 and f_st(p_i) = 1)
   begin
      if f_t(p_i) = 0
         f_t(p_i) = 1.
      Extract and test the pattern corresponding to the pyramid
        position determined by p_i, and store result as O_net.
      if O_net ≥ τ_o
      begin
         f_st(p_i) = 1.
         Find the set of patterns on the vicinity of p_i (i.e. S_v(p_i)).
         for every p_v ∈ S_v(p_i)
         begin
            if f_t(p_v) = 0 or (f_t(p_v) = 1 and f_st(p_v) = 1)
            begin
               if f_t(p_v) = 0
                  f_t(p_v) = 1.
               Extract and test the pattern corresponding to the pyramid
                 position determined by p_v, and store result as O_net.
               if O_net ≥ τ_o
               begin
                  f_st(p_v) = 1.
                  Add p_v and O_net to the list L.
               end
            end
         end
         Merge the multiple detections stored in L and calculate the
           associated detection counter of the optimal detection p_o
           as N_v.
         if N_v ≥ τ_v
         begin
            Find the set of patterns that overlaps with the face
              defined at p_o. (i.e. S_f(p_o)).
            for every p_f ∈ S_f
               f_t(p_f) = 1.
         end
      end
   end
```

Table 5.1: The search algorithm.

In the evidence-guided search case, after every element of the $S_v(p_i)$ has been tested, the detections stored in $\mathcal{L}$ are merged into a single optimal detection p_o. In a similar fashion to the merging procedure described in chapter 3, the number of merged detections N_v is used to evaluate the plausibility of the detection being a real face by

comparing it with a threshold τ_v; if the number of merged detections is greater than or equal to τ_v a face is said to have been found.

When we are certain that a face has been found, we can dynamically reduce the size of the search space by marking as tested all those elements of the search space that would overlap with the face just found; this subset of the search space, which is similar to the subspace used for the guided search given in equation 5.15, is defined as follows:

$$S_f(p_o) = \{p_r \mid p_r \in S_s,\ x(p_r) \in \mathcal{X}(p_o, p_r),\ y(p_f) \in \mathcal{Y}(p_o, p_f),\ l(p_f) \in \mathcal{L}(x_i)\} \quad (5.16)$$

where

$$\begin{aligned}
\mathcal{X}(p_o, p_f) &= [x(p_o) f^{l(p_o)-l(p_f)} - \frac{P_{sz}}{2} f^{-l(p_f)}, x(p_o) f^{l(p_o)-l(p_f)} + \frac{P_{sz}}{2} f^{-l(p_f)}] \\
\mathcal{Y}(p_o, p_f) &= [y(p_o) f^{l(p_o)-l(p_f)} - \frac{P_{sz}}{2} f^{-l(p_f)}, y(p_o) f^{l(p_o)-l(p_f)} + \frac{P_{sz}}{2} f^{-l(p_f)}] \\
\mathcal{L}(p_o) &= [0, l(p_o)]
\end{aligned}$$

where P_{sz} is the size of the pattern side. The pseudocode of the search algorithm described in this section is given in table 5.1.

5.5 Experimental results

The main objective of this section is to compare the performance of the constrained search method described in this chapter with an exhaustive method, such as the one described in chapter 3.

The performance of the search method can be described in terms of two main features: the first one is the complexity of the task, in terms of the total number of elements tested in the search and the minimum number of analysed elements needed to detect all the faces in the scene, while the second one involves the general system performance, in terms of detected faces and false detections, and how it compares with respect to an unconstrained method.

The algorithm performance was tested by applying it to a set of images with human faces on them, at different distances and positions within the scene. In order to obtain a greater variety of subjects, in terms of physical measurements and faces, a mixture of real images and hand-cropped faces were used to compose 50 test images.

5.5.1 The test images

Ten different image sequences were obtained using the procedure described in chapter 4, and their depth and confidence maps were also estimated; the central frame of each sequence was used to compose an example figure. For five of the image sequences the camera tilt angle is 20 degrees, while for the other five it is 0 degrees; in all cases there is no rotation around the Z or X axes. The distances between the camera and the different structures in the scene are different for each sequence, and vary from a few meters to mostly unobstructed open space.

Ninety two faces were obtained from public face databases listed in appendix B, and are different to the ones used to train the classifiers used.

The images were composed by pasting the cropped scaled face images into the scene image. The procedure to decide the position and scale of the face was determined by using the subject model, the observer-scene relationship, and the image depth map to situate an hypothetical person in the scene. The procedure acts iteratively in order to determine the location and scale of the face in the image; while some of the parameters can be set at random (*e.g.* the horizontal position of the face), the dependencies between some of the other parameters is not simple; the face vertical position in the image plane depends on the height of the person, their distance from the camera, and the orientation of the camera with respect to the ground plane, while the scale of the face depends on the distance from the camera and the subject face size. The procedure described next solves this problem by setting the subject model first, and then computing the plausible regions of the scene where a face belonging to a real person could appear.

First, the horizontal position of an hypothetical person in the image is randomly determined. The human type to which such person belongs was determined by picking a number from a uniformly distributed random distribution in the interval $[0, 1]$, and deciding if the person belong to the large or small class by comparing it with a threshold set at 0.5. The stature and face size was determined by a random number picked from the corresponding Gaussian distribution used to model the subject; although the correlation between human height and face size is not clearly stated in the work of

Jürgens et al. (1990), for simplicity it was decided to use the same human model, either large or small, for the definition of the person face size and height.

The next step is to define the distance from the camera to the subject Z. For this we estimate first the range of distances at which it would be possible that the face would show in the image. The maximum distance is defined in terms of face scale *vs.* depth relationship expressed in equation 5.2 (given on page 112, $Z_{mx} = Z_l(0) + 3\sigma_{zl}$).

The minimum distance between the subject and the camera depends on two main conditions: the person should not be so close to the camera that the face does not fit in the image, and the person should not be so close to the camera that the face is not projected into the camera sensor. In our case the second condition is the more important, since the "height" of the camera mounted on the robot (1.25 m) is lower than most of the population considered in the model, with the main consequence that if a person gets too close to the camera the face, or a portion of it, will be projected above the upper limit of the portion of the image plane sensed by the camera. Because of this, we define the minimum distance by solving for Z in equation 5.7 (page 112)

$$Z = F\frac{d - H}{ax + by + cF} \tag{5.17}$$

where x is set to the vertical position of the top row in the image plane (X_0, estimated in appendix A), y is equal to the horizontal position already defined, and $[a, b, c]$ depend on the camera orientation with respect to the ground plane[5]. The actual distance along the Z axis where we will situate the hypothetical subject is obtained as a random number picked from the uniform distribution defined over the obtained depth interval.

The projection of the subject face into the image plane is obtained by solving for x in equation 5.7. Once we have obtained the coordinates of the face in the image plane, we obtain the distance along the Z axis from the camera to the closest object in the scene from the depth map, and we compare it with the position along the Z axis of the subject; if the distance to the subject is larger than that of the nearest object, *i.e.* the face will be occluded by that object if set in the real scene, we recalculate the camera-subject distance as a random number in the interval defined by the minimum

[5] In our case the value assigned to y does not matter: b is equal to zero since there are no rotations around the X or Z axis of the camera centred coordinate frame.

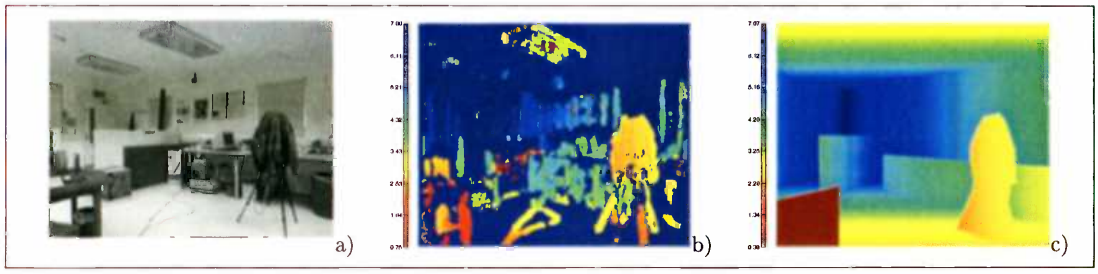


Figure 5.10: Real and ideal depth maps: images (**a**) shows an example of scene from which the depth map was extracted, image (**b**) shows the computed depth map, while (**c**) shows the artificial ideal depth map.

distance and the distance to the nearest object, and then we recompute the vertical position of the face as before.

It is important to note that the depth maps obtained are not perfect, since the depth can not be estimated on non-textured surfaces as mentioned in the previous chapter. For this reason, and in order to obtain plausible test images, the depth maps were "fixed", by fitting surfaces to clusters of reliable depth estimation points belonging to the same surface, under the least square criterion; the reliable depth measurements were selected manually. Once the main surfaces in the scene were estimated, an ideal depth map was generated. Examples of the real and ideal depth maps are shown in figure 5.10.

Finally, using the subject face size and the distance from the camera, the face image is scaled and pasted into the scene image. The number of faces contained in each image varies between two and five. Because it could happen that the faces pasted into the image overlap, an overlap check is done before pasting the face into the image, (similar to the one used for the merging process described in chapter 3). If the new face overlaps with any of the ones already pasted in the image, it is discarded and the procedure described before is repeated. Fifty test images containing a total of 182 faces were generated; some examples of test images are shown in figure 5.11.

5.5.2 The method

Three different tests were tried on the set of images, and each one uses the search algorithm described in this chapter. For the first test, which is used as a control, every element of the search space was tested, and a random value was used to determine the

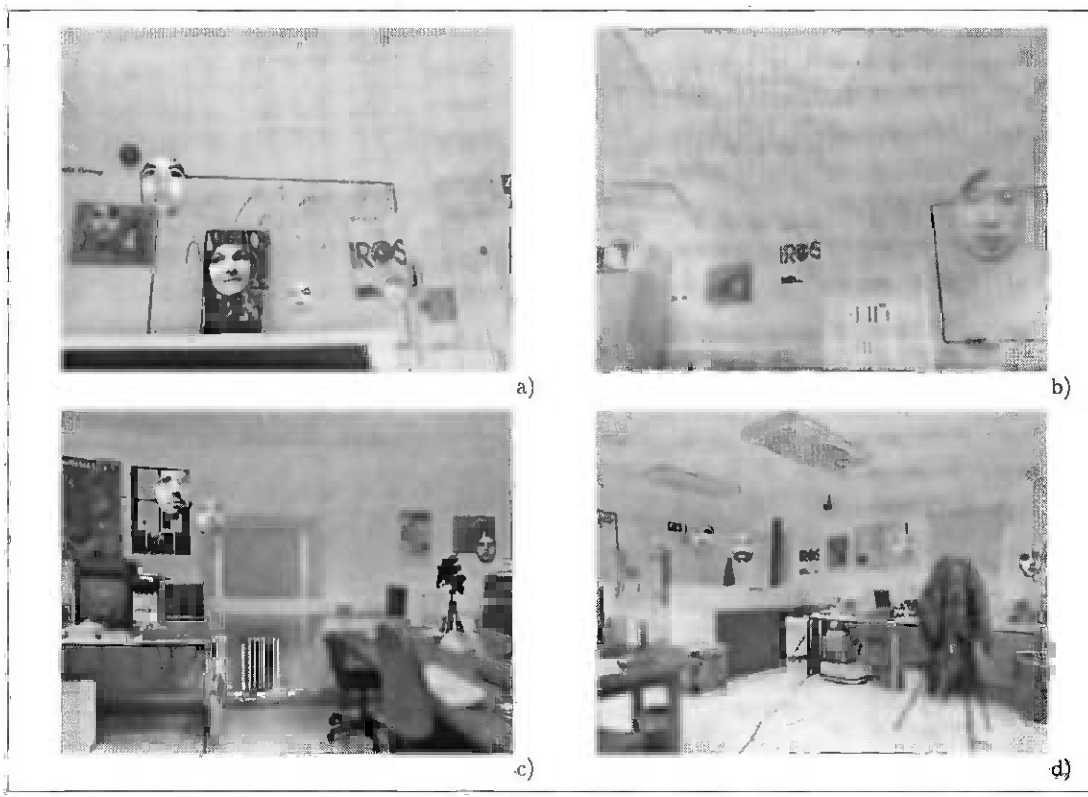


Figure 5.11: Examples of the test images: images (a), (b) were obtained using a camera tilt angle of 20 degrees, while in images (c) and (d) the orientation of the camera is parallel to the ground plane.

test order instead of the probability measure derived from the subject model. For the second and third tests, the search space is constrained, and the order of the search is determined by the probability measure drawn from the subject height model. The difference between them is that the second test uses the depth map obtained with the procedure described in the previous chapter in order to constrain the search space, while the third test uses an ideal map (like the one used to construct the test images) for the same purpose.

In the presentation of the mechanisms used to constrain the search space earlier in this chapter, we considered that it was convenient to define the boundaries that constrain the range of human height and face size measurements in terms of the standard deviations of the Gaussian distributions used for the subject's model. In general, the lower bound for the intervals was equal to three times the standard deviation below the mean of the small human type, while the upper bound was equal to three times the standard deviation above the mean of the large human type. While three times

the standard deviation is a reasonable value since more than 99% of the population would be considered, it is nevertheless an arbitrary value; in order to analyse how a change in the range of the interval will affect the size of the constrained search space and the performance of the classifier, the second and third tests were tried using five different spread values: 2, 2.5, 3, 3.5 and 4 times the standard deviation.

Two of the classifiers used for the experiments of chapter 3 were used for each test: the first one is the classifier labeled as FM1, which is composed of the mixture of the networks labeled F1 and F2, which were trained with the original training set; the second classifier is the one labeled NFM1, which is composed of the mixture of the networks labeled NF1 and NF2, which were trained with the extended training set. The parameters used in the search algorithm were set as follows: The vicinity parameters τ_d and τ_l were set to 3 pixels and 3 pyramid levels respectively, the classifier output threshold τ_o was set to 0 and the merging counter threshold τ_v was set to 8.

5.5.3 Results

The results obtained for the three experiments with different spread values are shown in table 5.2, which shows the detection rate, the total number of elements analysed in each case, and the false alarm rate with respect to the complete search space (general false alarm rate)[6] and the restricted search space (particular false alarm rate).

These results give an indication of how the performance of the system changes with respect to the classifier and search parameters used. Next, an analysis of the results is given, where the system performance is discussed first in terms of the detection rate, and second in terms of the false alarms produced.

It is important to note important differences between the set of images used for the evaluation of the system in this chapter and the set used in chapter 3: while the images used in chapter 3 were obtained from different sources, which implies that the quality of the image — in terms of illumination conditions, resolution and quantisation values used — varies from image to image, the scene images used in this chapter were captured under more or less even conditions. Also the face images pasted on the scene images

[6] Equivalent to the search space size of the test A.

Test	Spread [$\times\sigma$] (Space Size)	Classifiers: FM1 Classifier Detected Faces	FM1 Classifier False Alarms	NFM1 Classifier Detected Faces	NFM1 Classifier False Alarms
TEST B	2.0 (7,906,227)	170 (93.41%)	265 167,400 | 29,835	140 (76.92%)	31 1,431,002 | 255,040
	2.5 (8,985,523)	173 (95.05%)	289 153,498 | 31,092	144 (79.12%)	42 1,056,215 | 213,941
	3.0 (9,985,674)	176 (96 70%)	347 127,842 | 28,777	147 (80.77%)	53 837,001 | 188,409
	3.5 (10,898,410)	177 (97.25%)	380 116,740 | 28,680	151 (82.97%)	52 853,097 | 209,585
	4.0 (11,751,417)	177 (97.25%)	405 109,533 | 29,016	151 (82.97%)	53 837,001 | 221,725
TEST C	2.0 (4,682,867)	151 (82.97%)	97 457,330 | 48,277	112 (61.54%)	26 1,706,194 | 180,110
	2.5 (5,256,959)	152 (83.52%)	104 426,549 | 50,548	112 (61.54%)	36 1,232,251 | 146,027
	3.0 (5,722,155)	153 (84.07%)	123 360,659 | 46,522	112 (61.54%)	46 964,371 | 124,395
	3.5 (6,111,064)	150 (82.42%)	126 352,072 | 48,501	110 (60.44%)	46 964,371 | 132,849
	4.0 (6,421,620)	145 (79.67%)	131 338,634 | 49,020	112 (61.54%)	40 1,109,026 | 160,540
TEST A	— — (44,361,050)	180 (98.90%)	2156 20,576	156 (85.71%)	211 209,250

Table 5.2: Face detection results with the constrained search algorithm: in columns 3 and 5 the number of detected faces and the detection rate is given, while columns 4 and 6 show the number of false alarms and the false alarm rate, which is computed using as a reference the size of the unconstrained search space (bottom-left in the corresponding cell), and the size of the constrained search space (bottom-right in the cell). The total number of elements analysed in each experiment is given at the bottom of each cell in the second column.

have similar characteristics as the ones used to train the network; the resolution is adequate, and they have an even quality. The effect of this is that the results obtained will be better than the ones obtained in chapter 3. In fact, as can be seen in table 5.2 the results for the unconstrained search space (Test A) is excellent in terms of detected faces, and very similar in terms of the false alarm rate to the results obtained earlier for the same networks (see tables 3.4 and 3.7).

The most noticeable aspect of the results of experiments B and C is the positive correlation between the spread value and the detection rate; the detection performance

of the system tends to improve as we relax the subject model. The improvement is not just noticeable in terms of the number of detected faces, but also in the quality of detections, measured as the number of merged detections corresponding to a detected face; after analysing the number of detections merged per face[7] it was found that predominant behaviour is that as we increase the spread value the number of detections per face stays the same or increases its value. This behaviour is more stable in the case of the results obtained with the classifier NFM1.

It is important to note the exceptional case of the test C with the FM1 classifier and spread value 4σ , whose performance is worse than all the other spread values in the same class. The reason for this can be explained in terms of how the constraints on the search space change as we change the spread value.

As the spread value is increased, the constraints due to the observer-scene-subject geometrical relationship are relaxed, thus increasing the number of elements to be tested in the search space. On the other hand, the effect of an increase of the spread value on the constraints due to scene depth depends on the way the depth information is used: in the general case, where depth information about the subjects in the scene is available, an increase of the spread value implies a relaxation of the constraints, while in the particular case, when only depth information about the static structures in the scene is available, the constraints imposed on the search space will tighten as the spread value is increased.

The reason for this opposite behaviour lies in the role of depth information in the definition of the constraints used in each case. For the general case, depth information is used to "focus" the observer attention in the surfaces present on the scene; in this case the spread value is a measure of the uncertainty of the depth where a face might be found. For the particular case, depth information is used to make explicit the occlusions in the scene; an increase in the spread value implies an increase in the depth range where a face might be found, and therefore an increase in the likelihood of a face being occluded.

As explained before, test C uses an ideal depth map where the distance between every

[7] A table with the results is shown in appendix C.

scene point projected in the image and the observer is available; because of this the constraints derived from depth information will be more tightly enforced than in the experiments using the real depth map. The main effect of this in the results obtained is that the increasing effect of the depth constraints due to an increase of the spread value became evident, provoking a reduction of the detection rate.

The false alarm rate was on average smaller than the one reported for the same classifiers in chapter 3, while the total number of false detections is much smaller by at least an order of magnitude; this is a direct consequence of constraining the search space, since there are fewer oportunities for the classifier to fail.

Another important aspect of the search algorithm is how much the task of locating faces in the scene is simplified by constraining the search space. As can be seen in table 5.2, the total size of the search space for the images that compose the test set is reduced, in the case, to approximately a fourth of the complete space size for test B, and a seventh for test C.

While this gives us a good idea of how much the task of analysing a whole image is simplified, we also need to evalute how the indexing of the search space by using the human height model improves the performance of the system in terms of the time needed to locate the faces in the scene. Since the computation of the time varied depending on the computer used to execute the experiment, and the processor load during the execution, it was decided to use instead the number of analysed search space elements as a measure of time.

During the execution of the different experiments, the number of elements analysed until the first and last succesful detections in every image were recorded and for each of the three tests tried, the median and the smallest and largest values of this "time" needed for detection was computed. The results are shown in table 5.3.

For the tests B and C, using either the FM1 or NFM1 classifiers, as we increase the spread value, we note two main aspects on the results shown in table 5.3: there is no correlation between the spread value and the time needed to find the first face in the scene, and as the spread value increases, the time needed to find the last face in a scene also increases. This can be explained as follows: for the test images used in

Test	Spread ($\times\sigma$)	Classifiers			
		FM1 Classifier		NFM1 Classifier	
		Median First Detection	Median Last Detetction	Median First Detection	Median Last Detetction
TEST B	2.0	2983 [530 , 39814]	50277 [2698 , 137885]	6923 [631 , 114254]	77155 [1466 , 163910]
	2.5	3350 [573 , 60638]	51845 [4732 , 161197]	9072 [869 , 106948]	73614 [869 , 167964]
	3.0	2997 [374 , 85430]	71699 [3153 , 154674]	7786 [724 , 118086]	82062 [4837 , 180616]
	3.5	2918 [240 , 80360]	66443 [1318 , 158559]	6628 [693 , 117638]	84148 [4742 , 183458]
	4.0	3974 [564 , 89722]	62053 [2005 , 170946]	9923 [342 , 189241]	88483 [2642 , 194842]
TEST C	2.0	4010 [523 , 50912]	26243 [2611 , 111091]	5167 [818 , 68871]	30051 [1825 , 120826]
	2.5	2722 [245 , 47811]	27078 [2872 , 118958]	7058 [831 , 64343]	35669 [2575 , 121660]
	3.0	2852 [503 , 67377]	30322 [1770 , 118959]	7017 [891 , 75308]	30334 [2257 , 135777]
	3.5	3017 [453 , 66905]	24340 [2393 , 130359]	5675 [722 , 79597]	33307 [1810 , 145768]
	4.0	3005 [670 , 71483]	22077 [2969 , 117782]	5357 [1112 , 73752]	33895 [1398 , 167184]
TEST A	— —	13005 [1321 , 77106]	94549 [14167 , 379527]	43329 [1598 , 371629]	272892 [21228 , 834717]

Table 5.3: Search statistics: The statistics of the number of search space elements that need to be tested in order to find the first and the last face in the scene are shown. Each cell contains three numbers: the top value is the median of the number of elements needed to find the first or last face in the scene over the 50 test images, while the bottom-left and bottom-right values are the respective minimum and maximum values.

the experiments[8], an increase in the spread value will increase the size of the search space; the effect of this on the time needed to find the first face is negligible, since such elements added to the search space will have a low probability, and therefore will be the last to be tested. On the other hand this affects the time needed to detect the last face in the image, since more elements have to be tested before the last face can be detected —which should be located in a low-probability area anyway.

[8] This is particular to the images used, where the constraints obtained from the geometrical model weight more than the ones obtained from the scene structure.

The time needed to detect a face is also affected by the classifier used; for the classifier FM1, which in general is more sensitive than the classifier NFM1 (meaning that it is more likely to classify a non-face as a face), the time needed to find the first face and the last face is considerably smaller than the the time needed for the classifier NFM1. The quality of depth map also affects the speed of the system; a better quality depth map will allows us to constrain more tightly the search space, and therefore reducing in general the number of elements that need to be tested. This can be observed by comparing the results from tests B and C, where the latter, which has an ideal depth map, needs considerably less time to locate the faces in the scene, in particular the location of the last face in the scene.

The comparison between the experiments (tests B and C) with respect to the unconstrained experiment (test A), shows an important reduction in the time needed to find the faces.

The proportion of the search time needed to locate the first face for the classifier NFM1, compared to the results obtained with test A, is on average less than a fifth and less than a seventh for the test B and C respectively, while in the case of the time needed to find the last face, for test B the proportion is on average less than a third of the search time of test A, and less than an eighth for test C. In the case of the classifier FM1, for tests B and C, the average proportion of the search time for the first face is slightly less than the fourth compared with the results obtained with test A, while the last face average proportion of the search time for test B is aproximately less than five eighths and two sevenths for test C. A table that shows the average proportion of time saved on each case in terms of the exact search time reduction factor is shown in table 5.4 together with search space size reduction ratio.

	FM1 Classifier		NFM1 Classifier		Space
	First detection	Last detection	First detection	Last detection	size
Test B	4.00	1.56	5.37	3.37	4.48
Test C	4.16	3.63	7.16	8.36	7.87

Table 5.4: Reduction factor of the average search time *vs.* the reduction factor of the search space size: the first four columns show the time reduction factor for the first and last detected faces of tests B and C with respect to the equivalent detection time of the unconstrained example (test A), while the last column shows the average search space size reduction factor for the tests B and C with respect to the the size of the unconstrained search space.

The reduction in the search time is quite remarkable in all cases and quite similar in most cases to the reduction in the size of the search space, which as mentioned earlier, is about a fourth of the original space size for the test B and a seventh for the test C. In general we can say that for the classifier NFM1 the ratio is more stable, resembling more the ratio of the reduction of the search space size, while this is not the case for the classifier FM1. This can be explained in terms of the sensitivity of the classifiers, as it is more likely that an unconstrained search will find the faces faster using a sensitive classifier, than a less sensitive one. The smaller average search time reduction ratio of the classifier FM1 with respect to the similar reduction ratio of the other classifier can be explained by a faster unconstrained detection (the classifier FM1 is still faster that the classifier NFM1 in all cases).

5.6 Conclusions

In this chapter a framework for situated face detection was presented, which, by explicitly stating the geometric relationships between the observer, the scene and the subject, which emerge from the fact that the observer is situated in the scene, and by using raw structural scene information (the depth map), obtained from the interaction between the observer and the world, a set of constraints can be imposed into the search space. An algorithm was also proposed to perform a guided search, which allows us to obtain definite results before the whole search space has been tested.

The results show an important reduction in the search space size, which implies a faster processing time per image, and an even faster processing time before a subject is detected.

More thorough conclusions will be drawn in the next chapter where final conclusions of the work presented in this thesis are given.

Chapter 6

Conclusions

In this last chapter the final conclusions of this thesis are presented. The partial results obtained in the chapters on the individual modules that compose the system framework will be reviewed, and their main characteristics, advantages and disadvantages will be brought into the context of the general system.

The chapter is organised as follows: first the thesis is summarised in section 6.1. Achievements and contributions are presented in section 6.2, followed by section 6.3, where a discussion of the limitations of the proposed framework and ideas for future work are given.

6.1 Thesis summary

This thesis presents a novel framework for a real-time system for detection and location of faces in a scene. In the first chapter, a brief summary of the psychological basis of face processing is given together with a discussion of automatic face processing systems, and the role of automatic face detection as the very first stage in the process. Also a distinction is made between the face detection and face location problems, identifying the search problem as the main obstacle for the implementation of a real-time face detection system. The objective and scope of this thesis is presented with a description of how a method for automatic face detection can benefit from environmental constraints to simplify the task, and finally how these constraints fit into the artificial intelligence paradigm proposed by Brooks (1991) is explained.

A review of the approaches for automatic face detection is presented in the second chapter, classifying the different proposed approaches as feature-based methods and pattern recognition based methods. It was shown why pure feature-based detection methods are NP-complete problems, and that the most common way of simplifying the problem (labeling), involves solving a pattern recognition method problem. In opposition, the location problem, inherent in pattern recognition methods for face detection, while being computationally expensive, has a linear complexity, and therefore can be solved efficiently.

The next two chapters are devoted to describing the implementation of the two main modules of the framework: a face/non-face pattern classifier and a depth estimation module.

The face classifier is based on an approach proposed by Rowley et al. (1998), a multilayer perceptron with retinal connections. Weight sharing was used, transforming the classifier into a fully convolutional neural network. With the idea of increasing the range of face orientations that could be detected by the classifier, two similar networks were trained using different training sets: the first training set was composed of front view face patterns, while side view face patterns were used in the second training set. The system was tested on a standard set of test images; the front view face classifier succeeded at its goal, and weight-sharing was proved a useful technique to increase the generalisation capabilities of the classifier. The classification of side view patterns, on the other hand, proved to be a much more complex problem, given the large variance induced in the patterns by changes in the face orientation, and failed to produce acceptable results. Also, and for comparison reasons a DFFS-DIFS classifier, similar to the one proposed by Sung and Poggio (1998) was implemented, and tested.

A depth estimation module was proposed in order to obtain structural information about the scene inhabited by the observer. The module estimates depth from the disparity estimated by comparing individual frames from a sequence of images captured from a camera moving in a predetermined path. A novel hierarchical method is proposed to mix the different disparity maps, allowing us to obtain fast results, while increasing the quality of the map as times goes by.

Finally, in chapter 5 a geometrical method that describes the relationship between the observer, the scene and the subject is used together with scene depth information to define a set of constraints on the search space.

Constraints on the search space can be derived from the scene depth information and, based on the availability of this information, two cases were identified: the general case, where the depth information can be obtained at a similar rate to the capture rate of gray level images, and a particular case where the system works under the premise that the scene should be static. For the general case, a bounded scale interval where a face might be found is defined, while for the particular case an open interval is defined, constraining the maximum distance from the observer at which a face might be found in the scene. In the work presented here since the depth information is obtained from a moving camera, only the particular case was tested.

The other source of constraint is based in the observer-scene-subject geometrical relationship. For that reason, an anthropometric model of the subject is defined, and by combining it with the known position of the observer with respect to the scene (camera position and orientation with respect to the ground), and the projective model that describes how the scene is projected into the camera's image plane a set of constraints on the search space are defined.

Next a search algorithm, which uses the observer-scene-subject geometrical relationship to rank the elements of the search space, was also proposed. The ranking of the search space allowed us to have an ordered face search, where the places where a face is more likely to be found are tested first (using the classifiers described in chapter 3). The search algorithm also allowed us to obtain definite location of detected faces before the whole search space has been tested, by exploiting the known behaviour of the face classifier, namely multiple detection in the vicinity of faces in the image.

The whole system was tested using sequences of real images mixed with cropped front-view face images. An important reduction of the search space size and detection time were achieved.

6.2 Achievements and contributions

The framework presented in this thesis depends on a set of modules, whose viability, in terms of simplicity of the implementation, and amount of resources needed, was considered to be critical for the practical implementation of the approach. Because of this, the achievements and contributions of the work presented in this thesis will be first listed in terms of the new ideas and improvements to known technologies used in the individual modules that compose the framework, and finally the main contribution of this thesis will be stated.

- A front view face detector was successfully implemented, and the original technology was improved by increasing its generalisation capabilities, managing to obtain better results than an un-improved classifier using a smaller training set, and in general achieving similar results as those published.
- An extra insight was obtained in the complexity of the detection of complex three dimensional objects from their two dimensional representations, in particular the necessity of dividing the problem by using a larger number of viewer-centred representations, to minimise the pattern variation.
- A method for fast depth estimation was implemented, defining a new method suitable for real-time operation.
- The different relationships between an observer and the scene were properly determined, allowing us to define a set of constraints on the search space.
- A novel search algorithm, suitable for real-time operation, was proposed and tested, obtaining a considerable increase with respect to traditional methods for face detection.

The main contribution is the definition of a new framework that allows us to exploit the situatedness of the observer to reduce the complexity of the search in a face detection system. It was proved experimentally that an important reduction of the computational time needed for the analysis of the image is obtained by applying the proposed framework.

6.3 Future Work

Given the many different topics involved in the definition of the framework presented in this thesis, there are many problems that should be addressed, or features of the framework that can be improved.

One of the main issues that should be addressed is the implementation of a system where constraints from depth information are derived using the general case. That would imply, first, a new depth estimation module that would allow us to obtain depth information at a similar rate to the one used to capture the images. That can be achieved by using two or more cameras in order to eliminate the need of camera motion to induce image disparity. The algorithm presented in chapter 4 can be used if more than two cameras are used, or if the cameras can be translated.

A problem that was not solved in this thesis is the implementation of a side view classifier. The main identified problem is the large variance induced in the face pattern by small changes in face orientation. In order to solve this problem, a subdivision into smaller classes in terms of orientation of side view faces is necessary. In order to properly normalise and classify side view faces, information about their three dimensional orientation would be needed, and since most face databases are composed only of two dimensional representations of faces, either a database of three dimensional faces, or the use of a method to extract the three dimensional orientation of a face from two dimensional images, would be needed.

A limitation of the approach is the simplicity of the subject model, and the assumptions made about the scene-subject relationship. In the framework presented, only the adult population is considered, with the main drawback that small children will not be detected by the system. This can be partially solved by relaxing the geometric constraints imposed in the search space, by extending the lower boundary of the interval towards the ground plane, keeping the geometric constraints only on the top of the image. This solution is not satisfactory though, since the whole population model is not being used, and during the search procedure, the areas of the scene where childrens' faces might be located will be the last to be tested. Also the assumption about the subject-scene relationship is quite limited; we assume that all the people

found in the scene will be standing, and that the ground is always a flat surface. Cases common in indoor environments, like people sitting in chairs, tables, or the floor are not considered, as well as common un-eveness of the ground surface, such as ramps or stairs.

One possible solution to this problem would be the definition of a more complex model of the scene-subject relationship. Such a model would, for example, estimate the geometric constraints and the search ranking index using a more thorough definition of what is the ground plane (*e.g.* the ground plane is any flat or nearly flat surface on the image), and a wider variety of relationships between the subject and the ground plane would be allowed (*e.g.* sitting, laying on the ground, laying on one side, etc.). This approach to solving the problem can be considered to belong to the traditional approach of artificial intelligence, *i.e.* "good old fashioned artificial intelligence" or GOFAI. Such paradigm relies heavily on complex models of the world, and would require to some extent an explicit, high level understanding of the world[1], a complex and largely unresolved problem. This is inappropriate for the solution of the face detection problem, which is in humans an ability that is very likely to be innate, and does not require any kind of reasoning. Also, such a solution to the problem would contradict the spirit of the work presented in this thesis, where assumptions about the world have been kept to a minimum level.

A more adequate solution would be to use the simple model proposed in this thesis as minimal functional model, and then dynamically improve it by learning the probabilistic distribution where faces are likely to be found in a particular environment.

[1] *e.g.* the system should be able to extract surfaces from the depth map, and find those where a person could be found sitting or standing.

Bibliography

Adiv, G. (1985). Determining three-dimensional motion and structure from optical flow generated by several moving objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 7(4):384–401.

Aloimonos, J., Weiss, I., and Bandyopadhyay, A. (1988). Active vision. *International Journal of Computer Vision*, 2(3):333–356.

Anandan, P. (1984). Computing dense displacement fields with confidence measures in scenes containing occlusion. In *Intelligent Robots and computer vision*, volume 521 of *Proceedings of SPIE-The international Society for Optical Engineering*, pages 184–194. Society of Photo-Optical Instrumentation Engineers.

Anderson, B. and Moore, J. (1979). *Optimal Filtering*. Prentice-Hall.

Ballard, D. H., Becker, T. G., Brown, C. M., Gans, R. F., Martin, N. G., Olson, T. J., Potter, R. D., Rimey, R. D., Tilley, D. G., and Whitehead, S. D. (1988). The rochester robot. Technical report, University of Rochester. Department of Computer Science.

Beauchemin, S. S. and Barron, J. L. (1995). The computation of optical flow. *ACM Computing Surveys*, 27(3):433–467.

Ben-Yacoub, S. (1997). Fast object detection using MLP and FFT. IDIAP-RR 11, IDIAP.

Berkeley, G. (1709). *Essay Towards A New Theory of Vision*, volume 1 of *The works of George Berkeley, Bishop of Cloyne; edited by A. A. Luce and Jessop, T. E.* Thomas Nelson and Sons Ltd (1948), London. Bibliotheca Britannica Philosophica.

Blake, A. and Yuille, A., editors (1992). *Active vision*. MIT Press.

Brooks, R. A. (1991). Intelligence without reason. In Myopoulos, John; Reiter, R., editor, *Proceedings of the 12th International Joint Conference on Artificial Intelligence*, pages 569–595, Sydney, Australia. Morgan Kaufmann.

Bruce, V., Green, P. R., and Georgeson, M. A. (1996). *Visual Perception: Physiology, Psychology, and Ecology*. Psychology Press, 3rd edition.

Bruce, V. and Young, A. (1986). Understanding face recognition. *British Journal of Psychology*, 77:305–327.

Burt, P. J. (1988). Smart sensing within a pyramid vision machine. *Proceedings of the IEEE*, 76(8):1006–1015.

Buxton, B. F. and Buxton, H. (1983). Monocular depth perception from optical flow by space time signal processing. *Proceedings of the Royal Society of London, Series B*, 218:27–47.

Canny, J. (1986). A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6):679–698.

Colmenarez, A. J. and Huang, T. S. (1997). Face detection with information based maximum discrimination. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 782–787.

Ellis, H. D. (1981). Theoretical aspects of face recognition. In G. Davies, H. Ellis, J. S., editor, *Perceiving and Remembering Faces*, chapter 8, pages 171–197. London Academic Press.

Ellis, H. D. (1986). Process underlying face recognition. In Bruyer, R., editor, *The Neuropsychology of face perception and facial expression*, chapter 1, pages 1–27. Lawrence Erlbaum Associates.

Ellis, H. D. and Young, A. W. (1989). Are faces special ? In *Handbook of Research on Face Processing*, chapter 1, pages 1–19. Elsevier Science Publishers B.V. (North Holland).

Fasel, B. (1998). Fast multi-scale face detection. IDIAP-COM 4, IDIAP.

Faugeras, O. (1993). *Three Dimensional Computer Vision: A Geometric Viewpoint.* Artificial Intelligence Series. The MIT Press.

Fukunaga, K. (1990). *Introduction to statistical pattern recognition.* Harcourt Brace Jovanovich, 2 edition.

Govindaraju, V., Srihari, S. N., and Sher, D. B. (1990). A computational model for face location. In *Procedings of the Third International Conference on computer vision.*

Gray, R. (1990). *Entropy and Information Theory.* Springer-Verlag.

Guyon, I., Albrecht, P., Cun, Y. L., Denker, J., and Hubbard., W. (1991). Design of a neural network character recognizer for a touch terminal. *Pattern Recognition*, 24(2):105–119.

Hallam, J. C. T. and Malcolm, C. (1994). Behaviour: Perception, action and intelligence — the view from situated robotics. *Proceedings of the Royal Society of London, Series A*, 349:29–42.

Han, C.-C., Liao, H.-Y. M., Yu, K.-C., and Chen, L.-H. (1997). Fast face detection via morphology-based pre-processing. Technical Report TR-IIS-97-001, Institute of Information Science, Academia Sinica, Taiwan, Nankang 115, Taipei, Taiwan, ROC.

Hayes, G. (1989). A read-time kinetic depth system. M.sc. dissertation, Department of Artificial Intelligence, University of Edinburgh.

Heeger, D. J. and Jepson, A. D. (1992). Subspace methods for recovering rigid motion i: Algorithm and implementation. *International Journal of Computer Vision*, 7(2):95–117.

Horn, B. K. P. and Schunck, B. G. (1981). Determining optical flow. *Artificial Intelligence*, 17:185–203.

Huang, L. and Aloimonos, Y. (1991). Relative depth from motion using normal flow: an active and purposive solution. Technical Report CAR-TR-535, Computer Vision Laboratory, Center for Automation Research, University of Maryland, College Park, MD 20742-3411.

Jarvis, R. A. (1983). A perspective on range-finding techniques for computer vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5:122–139.

Jolliffe, I. T. (1986). *Principal Component Analysis.* Springer series in statistics. Springer-Verlag.

Jürgens, H. W., Aune, I. A., and Pieper, U. (1990). *International data on anthropometry.* Number 65 in Occupational safety and health series. International Labour Office.

Kanatani, K. (1993). *Geometric Computation for Machine Vision.* Number 37 in Oxford engineering Science Series. Oxford Science Publications.

Kirby, M. and Sirovich, L. (1990). Application of the kurhunen-loeve procedure for the characterization of human faces. *IEEE Transactions on pattern analysis and machine intelligence*, 12(1):103–108.

Koenderink, J. J. and van Doorn, A. J. (1976). Local structure of movement parallax of the plane. *Journal of the Optical Society of America*, 66(7):717–723.

Lang, K., Waibel, A., and Hinton, G. (1990). A time-delay neural network architecture for isolated word recognition. *Neural Networks*, 3:23–43.

LeCun, Y., Boser, B., Denker, J. D., Henderson, D., Howard, R., Hubbard, W., and Jackel, L. D. (1989). Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1:541–551.

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbar, W., and Jackel., L. D. (1990). Handwritten digit recognition with a back-propagation network. In Touretzky, D., editor, *Advances in Neural Information Processing Systems 2, (NIPS*89).*

Leung, T., Burl, M., and Perona, P. (1995). Finding faces in cluttered scenes using labeled random graph matching. In *Proceedings of the 5th International Conference on Computer Vision*, pages 637–644.

Longuet-Higgins, H. and Prazdny, K. (1980). The interpretation of a moving retinal image. *Proceedings of the Royal Society of London, Series B*, 208:385–397.

Longuet-Higgins, H. C. (1981). A computer algorithm for reconstructing a scene from two projections. *Nature*, 293:133–135.

Lucas, B. D. and Kanade, T. (1981). An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence.*, volume 7, pages 674–67.

Marr, D. and Hildreth, E. (1980). Theory of edge detection. *Proceedings of the Royal Society of London, Series B*, 207:187–217.

Marr, D. and Poggio, T. (1976). Cooperative computation of stereo disparity. *Science*, 194:283–287.

Matthies, L., Kanade, T., and Szeliski, R. (1989). Kalman filter-based algorithms for estimating depth from image sequences. *International Journal of Computer Vision*, 3:209–236.

Moghaddam, B. and Pentland, A. P. (1995a). An automatic system for model-based coding of faces. Technical Report 317, M.I.T. Media Laboratory Perceptual Computing Section. Appears in: IEEE Data Compresion Conference, Snowbird Utah, March 1995.

Moghaddam, B. and Pentland, A. P. (1995b). Probabilistic visual learning for object detection. Technical Report 317, M.I.T. Media Laboratory Perceptual Computing Section. Appears in: The 5th International Conference on Computer Vision, Cambridge MA, 1995.

Osuna, E., R, F., and Girosi, F. (1997). Training support vector machines: An application to face detection. In *Proceedings to the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 130–136.

Pentland, A. P. (1987). A new sense for depth of field. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(3):523–531.

Perkins, S. J. and Hayes, G. (1994). Real time optical flow based range sensing on mobile robots. Technical Report 675, Department of Artificial Intelligence, University of Edinburgh.

Perrett, D. I., Smith, P. A. J., Potter, D. D., Mistlin, A. J., Head, A. S., Milner, A. D., and Jeeves, M. A. (1985). Visual cells in the temporal cortex sensitive to face view and gaze direction. *Proceedings of the Royal Society of London*, B223:293–317.

Phillips, P., Moon, H., Rauss, P., and Rizvi, S. (1997). The feret evaluation methodology for face-recognition algorithms. *Computer Vision and Pattern Recognition*, pages 137–143.

Plaut, D. C., Nowlan, S. J., and Hinton, G. E. (1986). Experiments on learning back propagation. Technical Report CMU–CS–86–126, Carnegie–Mellon University, Pittsburgh, PA.

Rowley, H. A., Baluja, S., and Kanade, T. (1995). Human face detection in visual scenes. Technical Report CMU-CS-95-158R, Carnegie Mellon University.

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

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). *Learning Internal Representations by Error Propagation*, volume 1 of *Parallel Distributed Processing: Exploration in the Microstructure of Cognition*, chapter 8, pages 318–362. MIT Press.

Sergent, J. (1989). Structural processing. In *Handbook of Research on Faces Processing*, chapter 2, pages 57–91. Elsevier Science Publishers B.V. (North Holland).

Serra, J. (1982). *Image Analysis and Mathematical Morphology*. Academic Press, London.

Simoncelli, E. P. and Adelson, E. H. (1991). Computing opting flow distributions using spato-temporal filters. Technical Report 165, MIT Media Laboratory, Massachusetts Institute of Technology, Cambridge Massachusetts 02139.

Sonka, M., Hlavac, V., and Boyle, R. (1993). *Image Processing, Analysis and Machine Vision*. Chapman & Hall.

Sung, K.-K. and Poggio, T. (1994). Example-based learning for view-based human face detection. Technical Report 1521, M.I.T. Artificial Intelligence Laboratory and Center for Biological and Computational Learning. A.I Memo No. 1521 C.B.C.L. Paper No. 112.

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

Tanimoto, S. L. and Pavlidis, T. (1975). A hierarchical data structure for picture processing. *Computer Graphics and Image Processing*, 4(2):104–119.

Tsotsos, J. K. (1989). The complexity of perceptual search tasks. In Sridharan, N. S., editor, *Proceedings of the 11th International Joint Conference on Artificial Intelligence*, pages 1571–1577, Detroit, MI, USA. Morgan Kaufmann.

Tsotsos, J. K. (1990). Analyzing vision at the complexity level. *Behavioral and Brain Sciences*, 13(3):423–469.

Turk, M. and Pentland, A. P. (1991). Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1):71–86.

Ullman, S. (1979). *The interpretation of visual motion*. Artificial intelligence Series. M.I.T. Press.

Vaillant, R., Monrocq, C., and Cun, Y. L. (1994). Original approach for the localisation of object in images. *IEE Proceedings . Vision, image and signal processing*, 141(4).

Vapnik, V. (1995). *The nature of statistical learning theory*. Springer, New York.

Williams, T. D. (1980). Depth from camera motion in a real world scene. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2(6):511–516.

Yow, K. C. and Cipolla, R. (1997). Feature-based human face detection. *Image and Vision Computing*, 15(9):713–735.

Appendix A

Camera Calibration.

Camera calibration is the process of finding the internal or/and external camera parameters; the internal parameters determine the geometric and optical characteristics of the camera, while the external parameters determine the orientation and position of the camera with reference to a certain coordinate frame. The internal and external parameters are independent. In this appendix only the calibration of internal parameters will be treated.

The parameters that define the geometric characteristics of the camera are basically the focal length f, the coordinates of the optical centre of the camera (X_0, Y_0), and the image aspect ratio[1] (k_x, k_y). The relationship that defines the camera model is described in the following linear equation:

$$\begin{bmatrix} x \\ y \end{bmatrix} = \frac{1}{Z} \begin{bmatrix} \alpha_x & 0 & x_0 \\ 0 & \alpha_y & y_0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \tag{A.1}$$

where (x, y) is the projection of the scene point $P = (X, Y, Z)$ into the image plane I (see figure A.1), and the scale factors (α_x, α_y) combine the focal length and the image aspect ratio ($\alpha_x = k_x f$, $\alpha_y = k_y f$). We assume that the camera lens system is focused at infinity, therefore the focal length represents the actual distance between the image plane and the origin (see figure A.1). The focal length units are normally the same as the ones used to measure the distances in the 3D space (*e.g.* meters), therefore the image coordinate units (x, y) are also given in those units. The image obtained with the camera is a discrete version of the image projected into the camera's sensor, represented as a 2-D array of discrete elements called pixels. Since the information that gives the pixel density ratio of the image sensor is not always available, and since the image measurements are always represented as pixels, it is more convenient to conceive the parameter f as a conversion factor whose units are pixels instead of meters; this convention will be used in the rest of this appendix.

The optical characteristics of the camera model are described as the distortion produced by the camera's lens (radial and tangential distortion). As it was pointed out in

[1] The ratio between horizontal and vertical scales

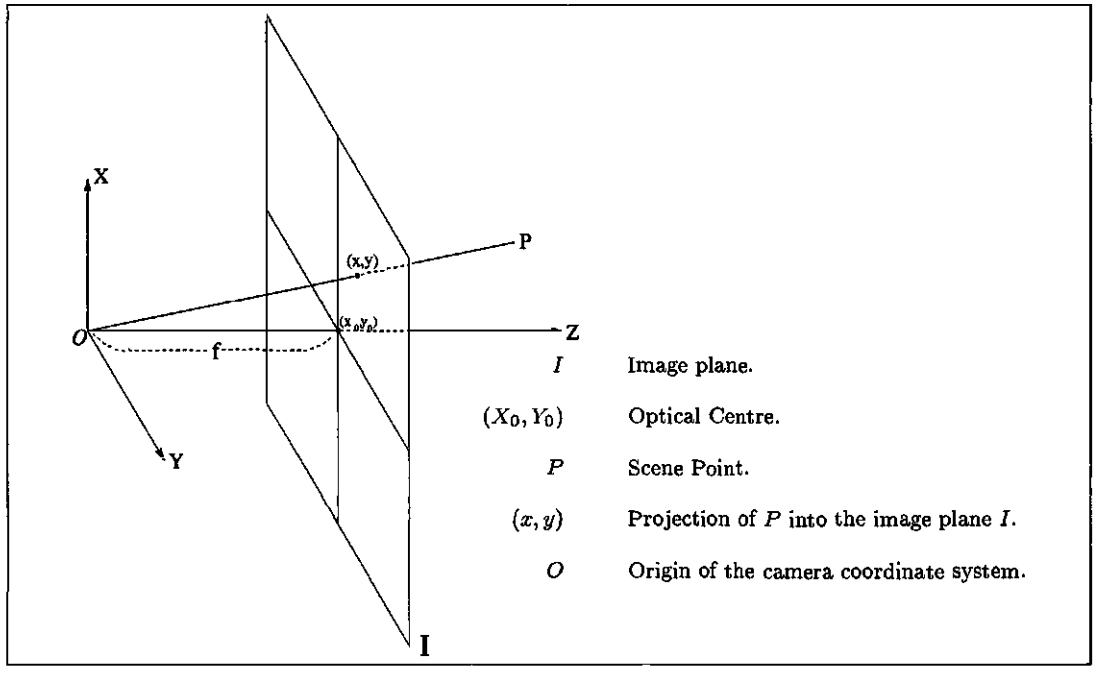


Figure A.1: Perspective projection of the scene.

Tsai (1987) for most industrial machine vision applications only the radial distortion has to be considered.

The approach for estimating the internal parameters of the camera used in this work is mainly based on the procedure described on Kanatani (1993), described on chapters 3 and 10.

A.1 Distortion elimination.

The first step in the calibration process is the elimination of image distortions produced by the camera's lens. For that reason a pair of functions that maps the pixel (x, y) to a new position (x', y') is found. In this case the equations are defined as the polynomial equations

$$x' = \sum_{i=0}^{m} \sum_{j=0}^{m-i} a_{ij} x^i y^j, \quad y' = \sum_{i=0}^{m} \sum_{j=0}^{m-i} b_{ij} x^i y^j, \quad m = 2 \tag{A.2}$$

where (x, y) and (x', y') are set of corresponding points in the original image and the desired image respectively. A number of N coordinates are collected from both images, and by substituting them in the equations A.2 a set of N linear equations with respect to the coefficients $a_{i,j}$ and $b_{i,j}$ are obtained; by solving them under a least squares criterion an approximation of the functions that describe the correct mapping is found.

In our example the original image is a grid of squares taken with the camera perfectly perpendicular to it (image **a** in figure A.2). The corners of the squares were used as reference points; in order to extract them a Canny edge detector (1986) was first

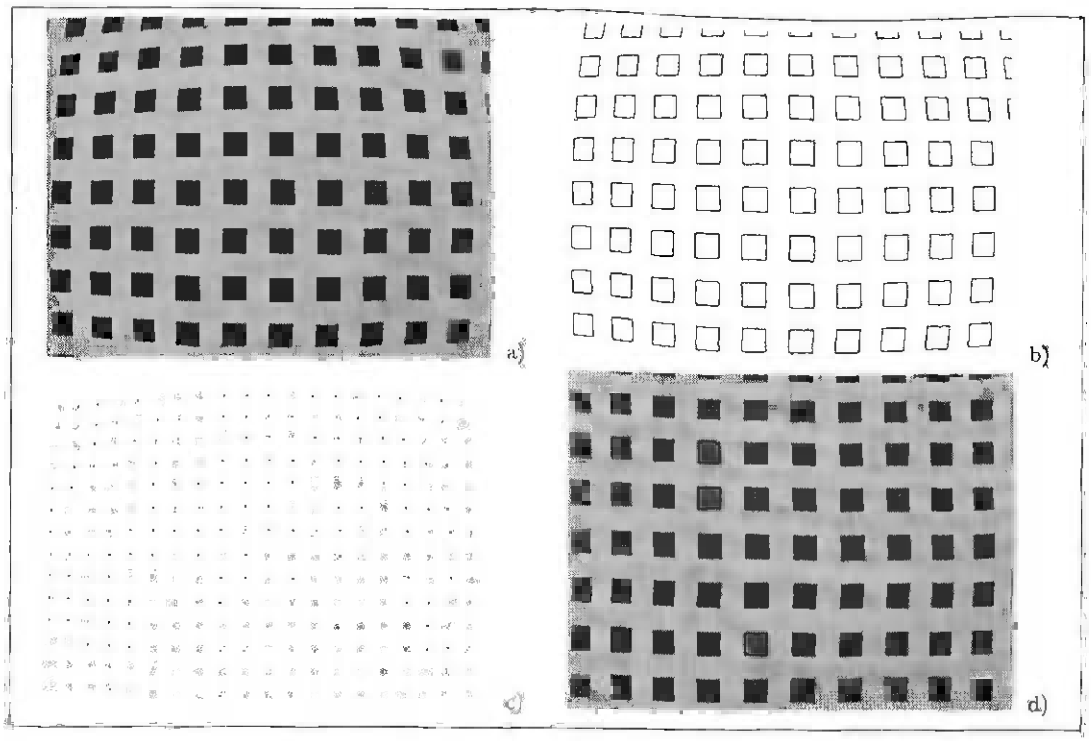


Figure A.2: Distortion elimination process.
Distortion elimination process: the image **a)** shows the original image, **b)** is the edges obtained with the Canny edge detector, **c)** shows the grid of points that match the corners of the squares in **b)** and **d)** shows the undistorted image.

applied to the image (**b** in figure A.2), lines were then fitted to the sides of each square and their intersection was registered as a reference point; 280 uniformly distributed reference points were obtained this way (**c** in figure A.2). Each square measures 1 cm. per side, and the distance between the camera and the grid is 13.5 cm.; a rough estimation of the focal length of the camera was estimated using the squares in the centre of the image, where the image distortion is smaller. A regular grid of 280 points was generated and centred to match the grid obtained from the image. The points were then substituted into the equation A.2, and the set of parameters a_{ij} and b_{ij} were estimated.

The next step is to use the equations A.2 to construct a pair of transformation maps (M_x, M_y) that records the correspondence between every pixel int the original image $I(x, y)$ and the transformed image $I'(x, y)$.

$$I'(x, y) = I(M_x(x, y), M_y(x, y)).$$

The original size of the images obtained from the camera is 640 × 480 pixels; after mapping the image size is 672 × 487.

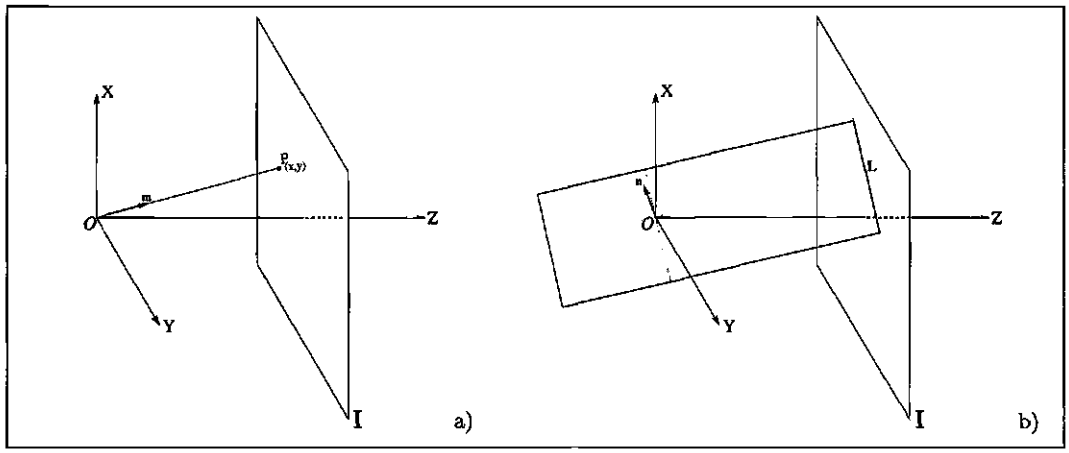


Figure A.3: Homogeneous coordinates for a point and a line in the image plane.

A.2 Preliminary Concepts.

A.2.1 Representation of points and images on the image plane.

Points and lines in the image plane I are represented with homogeneous coordinates, which are defined as a triplet of real numbers $\mathbf{m} = (m_1, m_2, m_3)^\top$, not all of them being 0. The relationship between the homogeneous coordinates $\mathbf{m}$ of a point P in the image plane and its Cartesian coordinates (x, y) is given by:

$$x = f\frac{m_1}{m_3}, \quad y = f\frac{m_2}{m_3} \tag{A.3}$$

A line on the image plane can be defined by a triplet of real numbers $\mathbf{n} = (n_1, n_2, n_3)^\top$, not all of them being 0; The line represented by $\mathbf{n}$ will appear on the image plane at the coordinates defined by

$$n_1 x + n_2 y + n_3 = 0 \tag{A.4}$$

The homogeneous coordinates of the point P on the image plane can be conceived as the vector starting from the origin O and pointing towards the point P (see **a)** on figure A.3). The homogeneous coordinates of a line L on the image plane can be conceived as the vector normal to the plane passing through the origin O and intersecting the line L (see **b)** on figure A.3). Since the relationship between homogeneous coordinates and their equivalent Cartesian representation relies only on the orientation of the vector, they can be multiplied by an arbitrary non-zero number without losing their properties; in order to keep the magnitude of vector in a reasonable range, we normalise them by dividing by their norm. In the rest of this appendix we will refer to points and lines on the image plane by their normalised vector. According to this the point in the image plane with Cartesian coordinates (x, y) will be represented by the normalised vector

$$\mathbf{m} = \pm\mathcal{N}[\begin{pmatrix} x \\ y \\ f \end{pmatrix}] \tag{A.5}$$

and a line $Ax + By + C = 0$ in the image plane will be represented by the normalised vector

$$\mathbf{n} = \pm\mathcal{N}[\begin{pmatrix} A \\ B \\ C/f \end{pmatrix}] \tag{A.6}$$

where f represents the focal length, and $\mathcal{N}$ is the normalisation operator defined by

$$\mathcal{N}[\mathbf{u}] = \frac{\mathbf{u}}{\|\mathbf{u}\|} \tag{A.7}$$

A.2.2 Statistical model of noise

When an image is captured by the camera, a certain amount of noise has to be taken into account in order to estimate the reliability of the measurements (points and lines detected in the image plane) used to estimate the internal camera parameters.

The noise of a image point P whose homogeneous coordinates are defined by the vector $\mathbf{m}$ is defined as a covariance matrix $\mathbf{V}[\mathbf{m}]$ with the following properties:

- It is symmetric and positive semi-definite.
- It is singular with the vector $\mathbf{m}$ itself: $\mathbf{V}[\mathbf{m}]\mathbf{m} = 0$
- Its spectral decomposition is defined by

$$\mathbf{V}[\mathbf{m}] = \sigma_1^2\mathbf{u}\mathbf{u}^\top + \sigma_2^2\mathbf{v}\mathbf{v}^\top + 0\mathbf{m}\mathbf{m}^\top \tag{A.8}$$

  where σ_1^2, σ_2^2 and 0 $(\sigma_1 > \sigma_2 > 0)$ are the three eigenvalues of $\mathbf{V}[\mathbf{m}]$ and $\{\mathbf{u}, \mathbf{v}, \mathbf{m}\}$ are the corresponding eigenvectors.

- the RMS of the orthogonal projection of the noise takes its maximum in the orientation of the vector $\mathbf{u}$ and its minimum in the orientation of the vector $\mathbf{v}$.
- The RMS magnitude of the noise is $\sqrt{tr(\mathbf{V}[\mathbf{m}])} = \sqrt{\sigma_1^2 + \sigma_2^2}$

The model of the noise assumes that noise occurs at each single image point and it is equally likely in all orientations with the same RMS magnitude ϵ (measured in pixels). Given this condition the covariance matrix $\mathbf{V}[\mathbf{m}]$ of the vector $\mathbf{m}$ of an image point P at a distance r from the image origin is given by

$$\mathbf{V}[\mathbf{m}] = \frac{\epsilon/f^2}{2(1 + r^2/f^2)}\left(\mathbf{u}\mathbf{u}^\top + \frac{\mathbf{v}\mathbf{v}^\top}{1 + r^2/f^2}\right) \tag{A.9}$$

where

$$\mathbf{k} = (0, 0, 1)^\top, \quad \mathbf{u} = \pm\sqrt{(1 + \frac{f^2}{r^2})}\mathbf{m} \times \mathbf{k}, \quad \mathbf{v} = \pm\mathbf{u} \times \mathbf{m} \tag{A.10}$$

We define the RMS error ϵ to be equal to one.

A.2.3 Point collinearity and line concurrency.

The vector $\mathbf{m}$ of the intersection point P of two image lines l and l' whose vectors are $\mathbf{n}$ and $\mathbf{n}'$ respectively, is given by:

$$\mathbf{m} = \mathcal{N}[\mathbf{n} \times \mathbf{n}'] \tag{A.11}$$

In the same way, the vector $\mathbf{n}$ of the line L that joins the two image points P and P' whose vectors are $\mathbf{m}$ and $\mathbf{m}'$ respectively, is given by:

$$\mathbf{n} = \mathcal{N}[\mathbf{m} \times \mathbf{m}'] \tag{A.12}$$

When we have a set of three or more points, we say that they are collinear if there is a line passing through all of them; when we have a group of three or more lines, we said that they are concurrent if they share they same intersection point. Formally we say that a set of points are collinear if the rank of their set of vectors is less than three, and a set of lines are concurrent if the rank of their set of vectors is less than three. The problem with this definition is that it is very sensitive to noise; more robust definitions of collinearity and concurrency are:

A set of N image points P_α with homogeneous coordinates $\mathbf{m}_\alpha$, $\alpha = 1, .., N$, are collinear if the smallest eigenvalue of their moment matrix

$$\mathbf{M} = \sum_{\alpha=1}^{N} W_{M\alpha} \mathbf{m}_\alpha \mathbf{m}_\alpha^\top \tag{A.13}$$

where $W_{M\alpha}$ are positive constants, is equal to zero; the associated eigenvector $\mathbf{n}$ is the vector of the line L passing through P_α. A set of N image lines L_α with homogeneous coordinates $\mathbf{n}_\alpha$, $\alpha = 1, .., N$, are concurrent if the smallest eigenvalue of their moment matrix

$$\mathbf{N} = \sum_{\alpha=1}^{N} W_{N\alpha} \mathbf{n}_\alpha \mathbf{n}_\alpha^\top \tag{A.14}$$

where $W_{N\alpha}$ are positive constants, is equal to zero; the associated eigenvector $\mathbf{m}$ is the vector of the point P of the common intersection of the lines L_α.

The set of constants $W_{M\alpha}$ and $W_{N\alpha}$ are weights that should be chosen so that reliable data are given large weights while unreliable data are given small weights. The optimal weights in the maximum likelihood sense for the equations A.13 and A.14 are respectively defined by:

$$W_{M\alpha} = \frac{1}{\mathbf{n} \cdot (V[\mathbf{m}_\alpha]\mathbf{n})}, \quad W_{N\alpha} = \frac{1}{\mathbf{m} \cdot (V[\mathbf{n}_\alpha]\mathbf{m})} \tag{A.15}$$

Since the computation of the optimal weights involves the use of the use of the vector $\mathbf{m}$ or $\mathbf{n}$ that we want to compute, the following approximation is used

$$W_{M\alpha} = \frac{1}{trace(\mathbf{V}[\mathbf{m}_\alpha])}, \quad W_{N\alpha} = \frac{1}{trace(\mathbf{V}[\mathbf{n}_\alpha])} \tag{A.16}$$

Once we have defined the optimal estimation of a line formed out of a set collinear points, or a point as the intersection of concurrent lines, we need to consider that the

noise on each of the points that form the line, or on each of the lines that intersect in a point will propagate into the new line or point. We define the covariance matrix of the optimal estimation of the intersection point vector $\mathbf{m}$ of N concurrent lines by

$$V[\mathbf{m}] = \frac{\mathbf{u}\mathbf{u}^\top}{\lambda_u} + \frac{\mathrm{v}\mathrm{v}^\top}{\lambda_v} \tag{A.17}$$

where $\{\mathbf{m}, \mathbf{u}, \mathrm{v}\}$ is the system of eigenvectors of the optimal moment matrix $\mathbf{N}$, with corresponding eigenvalues $(0, \lambda_u, \lambda_v)$. In the same fashion, the covariance matrix of the optimal estimation of the common line $\mathbf{n}$ of N collinear points is defined by

$$V[\mathbf{n}] = \frac{\mathbf{u}\mathbf{u}^\top}{\lambda_u} + \frac{\mathrm{v}\mathrm{v}^\top}{\lambda_v} \tag{A.18}$$

where $\{\mathbf{n}, \mathbf{u}, \mathrm{v}\}$ is the system of eigenvectors of the optimal moment matrix $\mathbf{M}$, with corresponding eigenvalues $(0, \lambda_u, \lambda_v)$.

A.2.4 Vanishing points.

A vanishing point of a space line is defined as the limit of the projection of the point that moves along the space line; if we situate a space line at the origin of the camera O, the vanishing point of that line is given by the intersection of the line with the image plane. From this interpretation it is clear that parallel space lines, when projected on the image plane, will intersect at a common vanishing point.

A.3 Estimation of Optical Centre.

The optical centre of the camera, or image center is calculated by finding the focus of expansion (FOE) of a set of spatial reference points while moving the camera along the optical axis. The FOE is defined as the vanishing points of the trajectories of points moving on the scene; if we move the camera along the optical axis, the trajectories of the reference scene points will be parallel to it and therefore will intersect the image plane on the optical centre.

The procedure for estimating the optical centre will be described next:

- The camera was mounted in a photographic copy stand, which allows us to control the distance between the camera and the object to be captured. The camera was mounted pointing down with its main axis parallel to the stand post. A grid of squares similar to the one used to eliminate the optical distortions was used as a reference. An image sequence of eight images were captured, each image taken with the camera situated at a different distance from the calibration grid.
- Four corners on the calibration grid were selected as reference points, their coordinates in the image plane were selected by fitting lines to them, and converted to normalised homogeneous coordinates. A tentative optical centre was defined as the centre of the image obtained from the frame grabber after mapping.

- Four lines were fitted to the the four sets of eight collinear points that were obtained from the coordinates of each corner on the image sequence, and their corresponding covariance matrix calculated, by using the method described in section A.2.3 (equations A.13 and A.16).
- The concurrent point of the four lines and its covariance matrix was calculated with the method described in section A.2.3 (equations A.14 and A.16).

The estimated optical centre $(X_0, Y_0)^\top = (320.61, 255.31)^\top$ with the covariance matrix $\mathbf{V}(X_0, Y_0)^\top$ equal to

$$\begin{pmatrix} 17.458451e^{-6} & -6.731742e^{-6} & -7.732514e^{-6} \\ -6.731742e^{-6} & 11.096990e^{-6} & -1.345538e^{-6} \\ -7.732514e^{-6} & -1.345538e^{-6} & 5.627252e^{-6} \end{pmatrix}$$

A.4 Focal length estimation.

The concept of conjugacy is the one we use to to estimate the focal length. We say the two points P and P' with homogeneous coordinates $\mathbf{m}$ and $\mathbf{m}'$ are conjugate to each other if and only if:

$$\mathbf{m} \cdot \mathbf{m}' = 0 \tag{A.19}$$

in other words, two points in the image plane are conjugate to each other if their vectors are mutually orthogonal. We define conjugacy in terms of Cartesian coordinates, applying the equation A.3, as follows: Two points in the image plane with Cartesian coordinates (a, b) and (a', b') are conjugate to each other if and only if

$$aa' + bb' + f^2 = 0 \tag{A.20}$$

From the definition of vanishing point and the concept of conjugacy we can deduce the following statement: Two space lines are orthogonal to each other if and only if their vanishing points are conjugate to each other on the image plane.

The procedure to estimate the focal length involves the estimation of the vanishing points with homogeneous coordinates $\mathbf{m}$ and $\mathbf{m}'$ of two orthogonal space lines projected into the image plane. Since vanishing points do not necessarily fall in the image, the best way of represent them is by using their normalised vectors. Because normalised homogeneous coordinates do not provide an explicit expression for the equivalent Cartesian coordinates on the image plane (if we do not know the value of the focal length), we can not directly use the equation A.20. Instead we express conjugacy in terms of the rate of change of a normalised vector $\mathbf{m}$ as the focal length of the system changes. From the definition of normalised vector A.5 we obtain the following expression: If $\mathbf{m}$ is the vector representing an image point with respect to the focal length f, the point $\mathbf{m}'$ representing the same image point with respect to the focal length f' is given by

$$\mathbf{m}' = \pm\mathcal{N}[\begin{pmatrix} m_1 \\ m_2 \\ (f'/f)m_3 \end{pmatrix}] \tag{A.21}$$

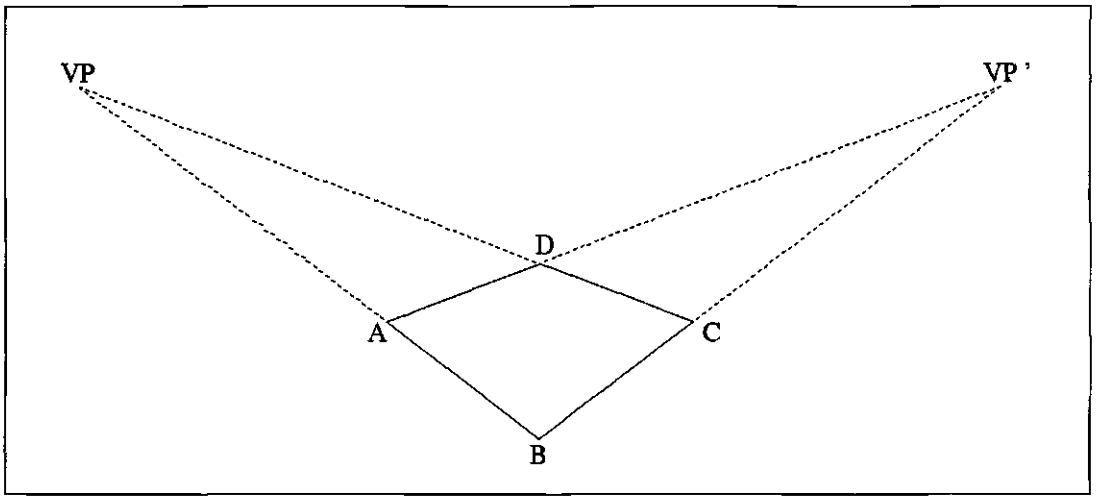


Figure A.4: Diagram of a rectangle placed on the scene. The intersection of the spatial lines $[\overline{AB}, \overline{DC}]$, and $[\overline{AD}, \overline{BC}]$ define the pair of mutually conjugate vanishing points VP and VP'.

By substituting A.21 into A.20 we obtain the following expression

$$m_1 m_1' + m_2 m_2' + \frac{\hat{f}^2}{f^2} m_3 m_3' = 0 \tag{A.22}$$

where $\mathbf{m}$ and $\mathbf{m}'$ are the vectors of the vanishing points of two mutually orthogonal space lines, defined with respect to a tentative focal length f, while $\hat{f}$ is the true focal length. From this it follows that the true focal length can be determined with the following equation

$$\hat{f} = f\sqrt{-\frac{m_1 m_1' + m_2 m_2'}{m_3 m_3'}} \tag{A.23}$$

The pair of conjugate vanishing points can be found by using the image of a rectangular surface placed in the scene (see figure A.4). The vanishing point VP is determined as the intersection of the lines passing through the segments $\overline{AB}$ and $\overline{DC}$, and the vanishing point VP' is determined as the intersection of the lines passing through the segments $\overline{AD}$ and $\overline{BC}$.

Since the vanishing points VP and VP' with homogeneous coordinates $\mathbf{m}$ and $\mathbf{m}'$ are calculated from a real image, a certain amount of noise described by their respective covariance matrices $V[\mathbf{m}]$ and $V[\mathbf{m}']$ should be considered. The variance in the estimation of the focal length is given by

$$V[f] = \frac{f^2}{4} \frac{\mathbf{m}' \cdot (V[\mathbf{m}]\mathbf{m}') + \mathbf{m} \cdot (V[\mathbf{m}']\mathbf{m})}{(m_3 m_3')^2} \tag{A.24}$$

The reliability of the focal length estimation also depends on the 3-D spatial configuration of the space lines used to determine the vanishing points; the accuracy of the estimation is degraded if the vanishing points found are far from the image centre, but because they are by definition conjugate to each other, there is a limited number of

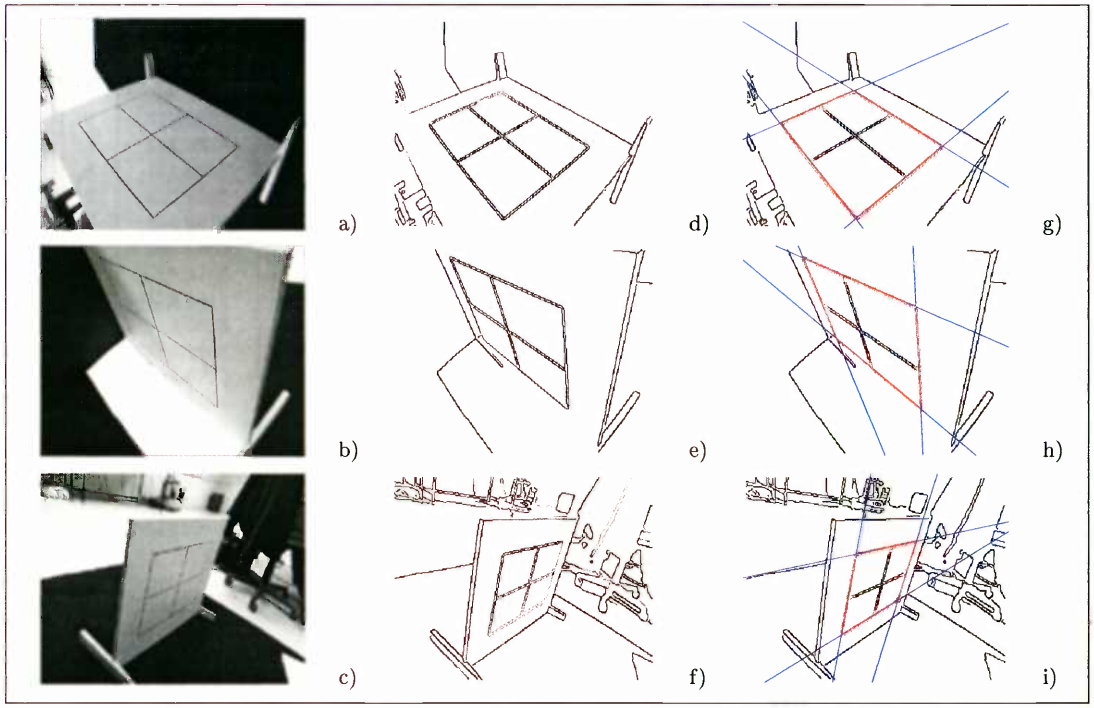


Figure A.5: Images **a)-c)** shows the same calibration frame viewed from different positions and orientations, images **d)-f)** shows the result of applying the Canny edge detector to figures **a)-c)**, images **g)-i)** shows the rectangle fitted to the calibration frame in red, and the two pairs of lines whose vanishing points are used to estimate the focal length in blue.

configurations that would allow both of them to be relatively close to the image centre. Instead of finding an ideal configuration to determine the focal length, we estimated it as the weighted average over N measurements obtained from different images. An optimal estimated focal length $\overline{f}$ is defined by

$$\overline{f} = \sum_{\alpha=1}^{N} W_\alpha f_\alpha \,, \quad \sum_{\alpha=1}^{N} W_\alpha = 1 \tag{A.25}$$

where f_α, $\alpha = 1, .., N$ are the estimates of the focal length from N different images.

The weights are defined in terms of the variance of the independent focal length estimations $V[f_\alpha]$ by

$$W_\alpha = \frac{1}{V[f_\alpha]} \Big/ \sum_{\beta=1}^{N} \frac{1}{V[f_\beta]} \tag{A.26}$$

while the variance of the optimal estimate is given by

$$V[\overline{f}] = 1 \Big/ \sum_{\alpha=1}^{N} \frac{1}{V[f_\alpha]} \tag{A.27}$$

From the computed variance of the optimal estimate of the focal length, a confidence interval can be computed. The statistic $(f - \overline{f})\sqrt{V[\overline{f}]}$ where f is the true focal length, obeys the standard normal distribution. A $(100 - a)\%$ confidence interval can be

α	f_α	$V[f\alpha]$	W_α	α	f_α	$V[f\alpha]$	W_α
1	441.3869	0.4515	0.0169	24	499.7343	0.9923	0.0077
2	543.2228	3.3584	0.0023	25	481.3260	2.2808	0.0033
3	558.4158	0.5050	0.0151	26	549.8075	0.7562	0.0101
4	511.6667	0.3233	0.0236	27	567.8223	0.5759	0.0133
5	513.9518	0.1868	0.0408	28	582.6858	0.7668	0.0100
6	482.6258	0.1022	0.0747	29	607.4887	1.0058	0.0076
7	490.9507	0.1191	0.0640	30	490.1392	0.1942	0.0393
8	494.9979	0.1527	0.0500	31	513.2585	0.2150	0.0355
9	509.5468	0.2796	0.0273	32	520.5131	0.3162	0.0241
10	510.6766	0.2941	0.0259	33	512.4199	0.2344	0.0326
11	522.5512	0.2005	0.0381	34	526.6368	0.4193	0.0182
12	529.6693	0.1451	0.0526	35	546.4528	6.1176	0.0012
13	488.2864	0.1453	0.0525	36	518.0850	3.3674	0.0023
14	503.0772	0.3142	0.0243	37	524.0083	1.8649	0.0041
15	483.1598	1.8686	0.0041	38	534.2519	0.9850	0.0077
16	482.5283	0.1780	0.0429	39	503.1861	2.1142	0.0036
17	492.0573	0.1562	0.0488	40	540.9092	5.4990	0.0014
18	490.4183	0.1835	0.0416	41	451.7808	3.9082	0.0020
19	482.8465	0.1774	0.0430	42	510.7017	7.8092	0.0010
20	508.3463	0.3910	0.0195	43	526.8043	2.8411	0.0027
21	477.6704	0.8332	0.0092	44	449.1148	1.3153	0.0058
22	461.4639	0.2794	0.0273	45	454.4147	2.5502	0.0030
22	465.0214	0.4203	0.0182	46	521.1126	7.3820	0.0010

Table A.1: Focal length estimation from different images: The first column shows the index of image used, the second column is the estimated focal length, the third column is the variance of the estimated focal length, and the last column shows the associated weight value.

defined as

$$\left[\overline{f} - \lambda_a\sqrt{V[\overline{f}]},\ \overline{f} + \lambda_a\sqrt{V[\overline{f}]}\right] \tag{A.28}$$

where λ_a is the $a\%$ point in the standard normal distribution.

A.4.1 Focal Estimation Procedure.

The procedure used to estimate the focal length is summarised in the next lines:

- 46 images from the calibration frame with different spatial configurations were captured, and mapped to eliminate the optical distortion. The calibration frame consist of a 0.30×0.40 m. black rectangle drawn in a white flat surface. Some examples are shown in images **a-c** in figure A.5.
- Edges in the image were extracted by using a Canny edge detector (see images **d-f** in figure A.5).
- Lines were fitted to the edges of the rectangle and their variance were estimated by using the method described in section A.2.3 (equations A.13, A.15 and A.18).
- Vanishing Points and their respective variances are estimated using the equations A.14, A.16 and A.17 (see images **g-i** in figure A.5).
- The focal length is estimated using the method described in section A.4.

The estimated values of the focal length, variance and associated weight obtained from each image are shown in table A.1. Using equations A.25 and A.27 the optimal estimate of the focal length and its variance were estimated as

$$\overline{f} = 501.6053, \quad V[\overline{f}] = 0.0076$$

then according with equation A.28, the true value of the focal length[2], with a 95% confidence should be in the interval [501.4341, 501.7765].

[2] The focal length is given in pixels. The actual distance between the image plane and the origin can be found if we know the pixel/cm. ratio of the image sensor.

Bibliography

Kanatani, K. (1993). *Geometric Computation for Machine Vision.* Number 37 in Oxford engineering Science Series. Oxford Science Publications.

Tsai, R. Y. (1987). *A versatile camera calibration technique for high-accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses.* IEEE Journal of Robotics and Automation, RA-3(4):323–344.

Canny, J. (1986). *A computational approach to edge detection.* IEEE Transactions on Pattern Analysis and Machine Intelligence, 8(6):679–698.

Appendix B

Face Databases.

Face Database	URL
USENIX Faces Database	`http://facesaver.usenix.org/`
MIT Face Database	`ftp://whitechapel.media.mit.edu/pub/images/`
University of Bern face database	`ftp://iamftp.unibe.ch/pub/Images/FaceImages/`
Image Sample from NIST Special Database 18 MID (Mugshot Identification Database)	`file://sequoyah.ncsl.nist.gov/pub/databases/data/mugshots.tar.Z`
DAI face database	`http://www.dai.ed.ac.uk/daidb/people/homes/arturoe/database/images.html`
CMU and MIT face detection test sets (Tests sets A,C and B respectively on (Rowley et al., 1995)).	`http://www.ius.cs.cmu.edu/IUS/eyes_usr17/har/har1/usr0/har/faces/test/`

Appendix C

Summary of detection results.

NETWORK FM1

	REAL DEPTH MAP					IDEAL DEPTH MAP				
	20	25	30	35	40	20	25	30	35	40
TEST 00										
000	04	04	04	04	04	04	04	04	04	04
001	15	20	20	20	13	19	20	18	20	18	18
002	07	12	12	12	12	12	13	13	13	13	13
003	14	13	13	16	16	16	13	16	16	17	16
TEST 01										
004	15	14	14	15	15	12	00	00	00	00	00
005	05	05	05	05	05	05	05	05	05	05	05
006	17	15	15	15	07	05	05	05	05	05	05
TEST 02										
007	26	26	25	22	22	23	07	07	07	06	04
008	19	32	31	32	32	32	31	32	32	32	26
009	08	08	08	08	08	08	00	00	00	00	00
010	23	25	22	26	23	23	20	20	20	20	16
011	06	06	06	06	06	06	00	00	00	00	00
012	15	15	15	15	15	15	02	02	02	02	02
TEST 03										
013	30	19	20	22	24	24	19	20	22	24	24
014	23	13	16	19	19	13	00	00	00	00	00
TEST 04										
015	20	16	16	16	16	16	16	16	16	17	17
016	11	11	11	10	11	11	11	11	10	11	11
017	23	21	21	21	21	21	21	23	21	21	23
018	16	10	16	16	16	16	16	16	16	16	16
TEST 05										
019	11	00	00	00	00	00	00	00	00	00	00
020	18	16	16	16	16	16	18	18	18	18	18
021	12	10	10	12	10	12	10	10	12	11	10
TEST 06										
022	26	17	20	20	20	20	17	20	20	20	21
023	08	05	05	07	09	05	05	07	07	09	09
024	18	18	18	18	18	13	18	18	13	13	13

NETWORK FM1

	REAL DEPTH MAP					IDEAL DEPTH MAP				
	20	25	30	35	40	20	25	30	35	40
TEST 07										
025	17	14	14	17	17	15	00	00	00	00	00
026	17	17	17	17	17	17	17	17	17	17	17
TEST 08										
027	14	00	00	01	06	06	00	00	00	00	00
028	04	02	02	02	02	03	02	02	02	02	02
029	09	09	09	09	09	09	01	01	01	01	01
TEST 09										
030	23	19	21	23	23	23	19	21	23	23	23
031	06	06	06	06	06	06	02	02	02	00	00
032	37	32	32	37	32	37	32	37	37	37	37
TEST 10										
033	20	20	20	20	20	20	03	03	03	03	00
034	17	17	17	17	16	16	00	00	00	00	00
035	09	01	06	06	06	08	01	01	01	01	01
036	15	08	08	08	08	08	06	06	06	06	06
TEST 11										
037	05	05	05	05	05	05	01	01	01	01	01
038	33	31	31	29	27	27	31	31	31	31	31
039	17	08	10	15	15	15	09	10	15	15	15
040	22	06	06	12	14	14	06	06	12	14	14
041	13	14	14	14	14	14	13	13	13	13	14
TEST 12										
042	08	03	03	03	04	03	00	00	00	00	00
043	09	02	04	07	07	07	02	04	07	07	07
044	12	11	11	11	11	11	11	11	11	11	11
045	09	09	13	13	13	09	09	09	09	07	07
TEST 13										
046	18	17	17	19	19	17	03	03	03	03	03
047	12	11	11	11	11	11	03	03	03	03	03
TEST 14										
048	26	18	18	11	23	23	01	01	01	01	01
049	25	21	22	21	21	21	05	05	03	01	00
050	16	16	16	16	16	16	16	16	16	16	16
TEST 15										
051	23	22	22	22	24	24	22	22	22	24	23
052	16	13	12	12	12	12	04	04	04	04	04
053	16	12	18	12	16	14	12	03	03	03	03
054	14	13	13	16	13	14	16	13	16	16	16
055	21	15	13	12	13	09	13	15	15	15	15
056	12	11	12	12	11	12	12	12	12	11	11
TEST 16										
057	12	04	15	22	22	19	00	00	00	00	00
058	18	00	11	11	11	11	00	00	00	00	00
059	19	19	19	23	19	08	12	13	12	12	13
TEST 17										
060	20	20	20	16	14	16	00	00	00	00	00
061	06	06	06	06	06	06	06	06	06	06	06
062	22	22	22	22	22	22	22	23	22	22	09

NETWORK FM1

	REAL DEPTH MAP					IDEAL DEPTH MAP				
	20	25	30	35	40	20	25	30	35	40
TEST 18										
063	07	02	02	02	02	00	02	02	02	02	00
064	17	17	17	17	17	17	10	10	10	10	10
065	14	13	13	13	14	13	13	13	13	13	13
066	18	01	06	09	09	09	01	06	09	09	09
TEST 19										
067	26	15	20	20	20	21	15	20	20	20	20
068	00	00	00	00	00	00	00	00	00	00	00
069	12	00	00	00	00	07	00	00	00	00	07
070	20	00	09	19	12	20	00	06	06	06	06
TEST 20										
071	17	17	17	17	17	17	16	16	16	16	16
072	24	11	15	17	18	10	11	15	17	16	17
073	18	18	18	18	18	18	18	18	18	18	18
TEST 21										
074	17	06	06	06	06	06	03	03	03	03	03
075	11	10	10	10	10	10	00	00	00	00	00
076	26	19	19	19	19	19	00	00	00	00	00
077	15	12	16	16	14	15	00	00	00	00	00
TEST 22										
078	09	00	09	09	09	09	00	00	00	00	00
079	09	09	09	09	09	09	09	09	09	09	09
TEST 23										
080	17	19	19	15	13	19	19	17	15	15	15
081	18	09	16	16	16	16	09	16	16	16	16
082	22	16	20	24	24	24	16	20	24	24	24
TEST 24										
083	11	13	12	12	12	12	00	00	00	00	00
084	19	10	11	11	11	11	10	10	10	10	10
085	12	13	13	13	13	13	03	03	03	03	03
TEST 25										
086	19	21	18	19	21	21	18	18	18	18	18
087	13	16	15	16	16	15	13	16	16	11	16
088	19	19	16	09	16	17	19	19	09	19	19
089	19	18	19	18	19	18	15	19	18	19	19
090	14	16	16	16	16	16	16	16	16	16	16
TEST 26										
091	16	17	18	18	16	16	16	16	16	08	08
092	18	18	18	18	18	18	02	02	02	02	02
093	14	14	14	14	14	14	14	14	14	14	14
094	24	20	22	19	23	25	13	13	10	06	06
TEST 27										
095	13	17	17	17	17	17	04	04	04	00	00
096	13	08	10	10	10	13	08	10	10	10	11
097	11	11	11	11	11	11	03	03	03	03	00
098	29	24	24	24	24	24	29	29	29	29	25
099	28	28	20	28	28	20	05	05	05	05	03
TEST 28										
100	06	06	07	07	07	07	00	00	00	00	00
101	15	22	15	22	12	15	03	03	03	03	03
102	08	03	03	03	05	05	03	03	03	05	05
103	20	16	16	14	14	14	06	06	06	06	06
104	08	08	08	08	08	08	05	05	05	05	05

NETWORK FM1

	REAL DEPTH MAP					IDEAL DEPTH MAP				
	20	25	30	35	40	20	25	30	35	40
TEST 29										
105	23	25	21	15	23	25	13	15	15	13	15
106	04	03	03	04	04	04	01	01	01	01	01
107	18	06	06	06	06	06	05	05	05	05	05
108	21	17	21	21	23	17	17	17	17	17	17
TEST 30										
109	08	11	11	11	11	11	07	07	07	07	07
110	25	15	23	21	23	23	15	23	21	23	21
111	33	24	28	29	26	27	24	25	29	26	26
TEST 31										
112	13	06	06	06	06	08	06	06	06	06	08
113	11	11	11	11	10	08	04	04	04	04	00
TEST 32										
114	17	20	20	16	16	16	20	18	20	20	20
115	14	07	08	10	13	13	07	08	10	13	13
116	10	08	10	09	10	10	05	05	05	05	05
TEST 33										
117	17	17	16	12	16	16	17	17	12	16	17
118	13	13	13	14	13	13	15	14	14	15	14
119	11	11	11	11	11	13	11	11	11	11	12
120	16	16	16	16	16	16	16	16	16	16	16
TEST 34										
121	14	17	16	14	17	17	03	03	02	02	02
122	14	04	10	10	10	13	00	00	00	00	00
123	21	21	21	21	18	21	21	21	21	20	21
124	21	15	16	15	15	15	15	15	15	15	15
TEST 35										
125	14	13	13	13	13	13	14	14	14	14	13
126	26	17	15	13	17	12	10	10	02	02	02
TEST 36										
127	22	14	14	15	16	16	14	14	15	14	16
128	09	09	09	09	09	09	09	09	09	09	09
129	00	00	00	00	00	00	00	00	00	00	00
130	15	02	06	06	06	11	02	06	06	06	11
TEST 37										
131	12	11	11	11	11	11	05	05	05	05	05
132	23	08	14	16	16	20	08	14	16	16	20
TEST 38										
133	19	21	19	14	21	15	09	09	09	09	09
134	22	18	20	24	18	24	18	20	24	24	24
135	23	18	10	23	22	18	18	18	13	18	18
136	17	24	24	24	24	23	17	17	17	14	10
TEST 39										
137	24	21	24	24	24	21	21	24	24	24	25
138	05	05	05	05	05	05	00	00	00	00	00
139	18	15	18	18	15	15	15	18	18	15	15
140	08	17	17	17	17	17	08	09	01	01	01
141	11	07	11	07	07	07	11	07	07	11	07
TEST 40										
142	18	11	11	10	10	14	11	11	10	10	10
143	24	25	26	20	11	26	02	02	03	03	04

NETWORK FM1

	REAL DEPTH MAP					IDEAL DEPTH MAP				
	20	25	30	35	40	20	25	30	35	40
TEST 41										
144	11	11	11	12	12	12	04	04	04	04	04
145	11	09	09	08	09	14	14	08	08	16	08
146	08	00	00	00	00	00	00	00	00	00	00
TEST 42										
147	07	01	01	01	01	01	01	01	01	00	00
148	10	10	10	10	10	10	10	10	10	01	01
149	18	19	19	19	19	19	00	00	00	00	00
150	08	08	08	08	08	08	00	00	00	00	00
TEST 43										
151	16	16	16	17	16	16	08	08	08	08	08
152	26	09	09	09	09	09	06	06	06	06	06
153	11	11	11	11	11	11	10	10	11	11	10
154	10	00	00	03	03	03	00	00	00	00	00
155	07	06	06	06	07	07	06	06	06	07	07
156	15	12	12	08	09	12	05	05	05	05	05
TEST 44										
157	16	00	00	07	14	14	00	00	06	14	14
158	15	08	08	11	12	12	08	08	11	12	12
159	11	13	13	11	13	11	13	13	11	13	13
160	17	09	09	09	09	09	08	08	08	04	04
161	12	12	12	12	12	12	01	01	01	01	00
TEST 46										
162	15	15	15	15	15	15	15	15	15	15	15
163	16	23	25	22	23	25	13	13	13	13	13
164	18	08	11	11	11	11	08	08	08	08	08
165	12	12	11	11	12	12	12	12	12	12	12
TEST 47										
166	13	12	12	13	13	13	12	12	13	13	13
167	23	23	23	23	23	23	23	22	23	23	23
168	17	16	16	16	16	16	15	15	15	15	15
169	18	17	20	20	17	20	20	17	20	20	20
170	05	00	00	00	03	03	00	00	00	00	00
TEST 48										
171	08	08	08	08	08	08	08	08	08	08	08
172	16	09	09	08	10	08	04	04	04	04	04
173	20	20	20	17	17	17	00	00	00	00	00
TEST 49										
174	21	21	21	21	17	17	07	07	07	07	07
175	18	18	18	18	18	18	04	04	04	02	02
176	13	12	12	12	12	12	06	06	06	02	02
177	24	20	24	20	20	20	19	19	19	16	16
178	20	15	14	15	15	14	04	04	04	04	04
179	24	13	13	13	13	16	12	12	12	12	12
TEST 50										
180	07	07	07	07	07	07	07	07	07	07	07
181	21	13	13	09	05	05	04	04	04	04	04

NETWORK NFM1

	REAL DEPTH MAP					IDEAL DEPTH MAP				
	20	25	30	35	40	20	25	30	35	40
TEST 00										
000	00	00	00	00	00	00	00	00	00	00
001	01	01	01	01	01	01	01	01	01	01
002	00	00	00	00	00	00	00	00	00	00
003	07	07	07	07	06	07	07	07	07	07
TEST 01										
004	05	05	05	05	05	00	00	00	00	00
005	00	00	00	00	00	00	00	00	00	00
006	10	07	07	07	02	02	02	02	02	02
TEST 02										
007	10	10	10	10	10	07	07	07	04	01
008	14	14	14	14	14	14	14	14	14	14
009	05	05	05	05	05	00	00	00	00	00
010	12	12	12	12	12	12	12	12	12	12
011	03	03	03	03	03	00	00	00	00	00
012	07	07	07	07	07	00	00	00	00	00
TEST 03										
013	13	11	11	11	13	13	01	01	11	13	13
014	02	02	02	02	02	02	00	00	00	00	00
TEST 04										
015	11	09	09	09	11	11	09	09	09	09	09
016	08	08	08	08	08	08	08	08	08	08	08
017	06	08	08	08	08	08	08	08	08	08	08
018	06	06	06	06	06	06	06	06	06	06	06
TEST 05										
019	07	00	00	00	00	00	00	00	00	00	00
020	09	07	07	07	07	07	09	09	09	09	09
021	06	06	06	06	06	06	06	06	06	06	06
TEST 06										
022	07	07	07	07	07	07	07	07	07	07	07
023	02	00	00	00	02	02	00	00	00	02	02
024	09	09	09	09	09	09	09	09	09	09	09
TEST 07										
025	07	06	07	07	07	07	00	00	00	00	00
026	07	07	07	07	07	07	07	07	07	07	07
TEST 08										
027	05	00	00	00	03	03	00	00	00	00	00
028	00	00	00	00	00	00	00	00	00	00	00
029	01	01	01	01	01	01	00	00	00	00	00
TEST 09										
030	00	00	00	00	02	00	02	00	00	00	02
031	00	00	00	00	00	00	00	00	00	00	00
032	08	08	08	08	08	08	08	01	01	01	08
TEST 10										
033	00	00	00	00	00	00	00	00	00	00	00
034	02	02	02	02	02	02	00	00	00	00	00
035	00	00	00	00	00	00	00	00	00	00	00
036	06	05	05	05	05	05	01	01	01	01	01

NETWORK NFM1

	REAL DEPTH MAP					IDEAL DEPTH MAP				
	20	25	30	35	40	20	25	30	35	40
TEST 11										
037	11	12	12	12	12	12	00	00	00	00	00
038	19	19	19	19	19	19	19	19	19	19	19
039	04	02	02	04	04	04	02	02	04	04	04
040	12	06	06	09	12	12	06	06	09	12	12
041	09	08	08	08	09	09	08	08	08	08	08
TEST 12										
042	00	00	00	00	00	00	00	00	00	00	00
043	00	00	00	00	00	00	00	00	00	00	00
044	01	01	01	01	01	01	01	01	01	01	01
045	02	02	02	02	02	02	02	02	02	02	02
TEST 13										
046	05	05	05	05	05	05	00	00	00	00	00
047	00	00	00	00	00	00	00	00	00	00	00
TEST 14										
048	05	04	04	04	05	05	00	00	00	00	00
049	12	13	13	13	13	13	02	02	00	00	00
050	05	05	05	05	05	05	05	05	05	05	05
TEST 15										
051	02	02	02	02	02	02	02	02	02	02	02
052	15	11	10	10	10	10	05	05	05	05	05
053	08	07	09	09	08	08	04	00	00	00	00
054	10	10	10	10	10	10	10	11	10	11	10
055	07	05	01	01	01	01	04	04	04	04	04
056	07	07	07	07	07	07	07	07	07	07	07
TEST 16										
057	10	02	10	10	08	10	00	00	00	00	00
058	10	00	10	13	13	13	00	00	00	00	00
059	13	13	13	13	12	13	01	12	12	12	12
TEST 17										
060	02	02	02	02	02	02	00	00	00	00	00
061	00	00	00	00	00	00	00	00	00	00	00
062	07	07	07	07	07	07	01	01	01	01	01
TEST 18										
063	01	01	01	01	01	01	01	01	01	01	01
064	01	01	01	01	01	01	00	00	00	00	00
065	02	02	02	02	02	02	02	02	02	02	02
066	09	00	03	06	06	06	00	03	06	06	06
TEST 19										
067	03	01	03	03	03	03	01	03	03	03	03
068	00	00	00	00	00	00	00	00	00	00	00
069	00	00	00	00	00	00	00	00	00	00	00
070	07	00	07	07	07	07	00	07	07	07	07
TEST 20										
071	11	11	11	11	11	11	11	11	11	11	11
072	03	00	00	02	02	02	00	00	02	02	02
073	06	06	06	06	06	06	06	06	06	06	06
TEST 21										
074	11	02	02	02	02	02	00	00	00	00	00
075	03	03	03	03	02	02	01	01	01	00	00
076	11	10	10	10	10	10	00	00	00	00	00
077	01	01	01	01	01	01	00	00	00	00	00

NETWORK NFM1

	REAL DEPTH MAP					IDEAL DEPTH MAP				
	20	25	30	35	40	20	25	30	35	40
TEST 22										
078	05	00	05	05	05	05	00	00	00	00	00
079	01	01	01	01	01	01	01	01	01	01	01
TEST 23										
080	06	06	06	06	06	06	06	06	06	06	06
081	05	04	05	05	05	05	04	05	05	05	05
082	09	09	09	09	09	09	09	09	09	09	09
TEST 24										
083	00	00	00	00	00	00	00	00	00	00	00
084	09	05	05	05	05	05	02	02	02	02	02
085	03	02	02	02	02	02	00	00	00	00	00
TEST 25										
086	02	02	02	02	02	02	02	02	02	02	02
087	12	12	10	12	12	12	12	12	12	10	12
088	08	08	08	08	08	08	08	08	08	08	08
089	05	05	05	05	05	05	05	05	05	05	05
090	09	09	09	09	09	09	09	09	09	09	09
TEST 26										
091	08	08	08	08	08	08	08	08	08	07	07
092	06	06	06	05	06	05	00	00	00	00	00
093	05	05	05	05	05	05	05	05	05	05	05
094	09	07	09	09	09	09	05	05	05	01	01
TEST 27										
095	10	10	10	10	10	10	00	00	00	00	00
096	07	05	05	05	05	06	05	05	05	05	07
097	00	00	00	00	00	00	00	00	00	00	00
098	12	07	07	07	07	07	12	12	12	12	12
099	10	10	10	10	10	10	06	06	06	06	04
TEST 28										
100	00	00	00	00	00	00	00	00	00	00	00
101	06	06	06	06	06	06	00	00	00	00	00
102	01	01	01	01	01	01	01	01	01	01	01
103	08	08	08	08	08	08	04	04	04	04	04
104	00	00	00	00	00	00	00	00	00	00	00
TEST 29										
105	08	08	08	08	08	08	08	08	08	08	08
106	00	00	00	00	00	00	00	00	00	00	00
107	04	00	00	00	00	00	00	00	00	00	00
108	11	11	11	11	11	11	11	11	11	11	11
TEST 30										
109	02	02	02	02	02	02	02	02	02	02	02
110	14	11	14	14	14	14	11	14	14	14	12
111	08	07	07	08	07	07	07	07	08	08	07
TEST 31										
112	01	00	00	00	00	01	00	00	00	00	01
113	00	00	00	00	00	00	00	00	00	00	00
TEST 32										
114	09	09	09	08	08	08	09	09	09	09	09
115	08	04	04	04	07	07	04	04	04	07	07
116	08	08	08	08	08	08	04	04	04	04	04

NETWORK NFM1

	REAL DEPTH MAP					IDEAL DEPTH MAP				
	20	25	30	35	40	20	25	30	35	40
TEST 33										
117	09	07	09	09	09	06	09	09	09	09	09
118	05	05	05	05	05	05	05	05	05	05	05
119	01	01	01	01	01	01	01	01	01	01	01
120	14	14	14	14	14	14	14	14	14	14	14
TEST 34										
121	03	03	03	03	03	03	00	00	00	00	00
122	08	03	07	07	07	08	00	00	00	00	00
123	06	06	06	06	06	06	06	06	06	06	06
124	18	08	08	08	08	08	08	08	08	08	08
TEST 35										
125	03	03	03	03	03	03	03	03	03	03	03
126	12	09	09	08	09	08	06	06	01	01	01
TEST 36										
127	05	05	05	05	05	05	05	05	05	05	05
128	00	00	00	00	00	00	00	00	00	00	00
129	00	00	00	00	00	00	00	00	00	00	00
130	01	00	00	00	00	00	00	00	00	00	00
TEST 37										
131	03	03	03	03	03	03	03	03	03	03	03
132	20	04	09	11	11	15	04	09	11	11	15
TEST 38										
133	07	07	07	07	07	07	02	02	02	02	02
134	08	08	08	08	08	08	08	08	08	08	08
135	04	04	04	04	04	04	04	04	04	04	04
136	10	11	10	11	10	11	10	10	10	08	07
TEST 39										
137	06	06	06	06	06	06	06	06	06	06	06
138	01	01	01	01	01	01	00	00	00	00	00
139	04	04	04	04	04	04	04	04	04	04	04
140	05	05	05	05	05	05	02	02	00	00	00
141	00	00	00	00	00	00	00	00	00	00	00
TEST 40										
142	06	05	05	05	05	05	01	01	01	01	01
143	01	01	01	01	01	01	00	00	00	00	00
TEST 41										
144	04	04	04	04	04	04	02	02	02	02	02
145	00	00	00	00	00	00	00	00	00	00	00
146	04	00	00	00	00	00	00	00	00	00	00
TEST 42										
147	09	02	02	02	02	02	02	02	02	00	00
148	04	04	04	04	04	04	04	04	04	04	04
149	12	12	12	12	12	12	01	01	01	01	01
150	05	05	05	05	05	05	00	00	00	00	00
TEST 43										
151	05	05	05	05	05	05	03	03	03	03	03
152	03	02	02	02	02	02	02	02	02	02	02
153	03	03	03	03	03	03	03	03	03	03	03
154	00	00	00	04	04	04	00	00	00	00	00
155	02	02	02	02	02	02	02	02	02	02	02
156	10	08	08	08	08	08	00	00	00	00	00

NETWORK NFM1

	REAL DEPTH MAP					IDEAL DEPTH MAP				
	20	25	30	35	40	20	25	30	35	40
TEST 44										
157	10 00	00	01	06	06	00	00	01	06	06
158	00 00	00	00	00	00	00	00	00	00	00
159	03 03	03	03	03	03	03	03	03	03	03
160	10 05	05	05	05	05	03	03	03	01	01
161	05 05	05	05	05	05	00	00	00	00	00
TEST 46										
162	05 05	05	05	05	05	05	05	05	05	05
163	07 07	07	07	07	07	02	02	02	02	02
164	01 01	01	01	01	01	01	01	01	01	01
165	03 03	03	03	03	03	03	03	03	03	03
TEST 47										
166	06 06	06	06	06	06	06	06	06	06	06
167	06 06	06	06	06	06	06	06	06	06	06
168	07 07	07	07	07	07	07	07	07	07	07
169	09 09	09	09	09	09	09	09	09	09	09
170	02 00	00	00	02	02	00	00	00	00	00
TEST 48										
171	04 04	04	04	04	04	04	04	04	04	04
172	01 01	01	01	01	01	01	01	01	01	01
173	10 10	10	10	10	10	00	00	00	00	00
TEST 49										
174	05 05	05	05	05	05	00	00	00	00	00
175	09 12	12	12	12	12	03	03	03	02	02
176	03 00	00	00	00	00	01	01	01	00	00
177	01 01	01	01	01	01	01	01	01	01	01
178	02 00	00	00	00	00	00	00	00	00	00
179	04 02	03	03	03	03	03	03	03	03	03
TEST 50										
180	03 03	03	03	03	03	03	03	03	03	03
181	10 05	05	05	00	00	00	00	00	00	00