Labeled Graphs and Dynamic Link Matching for Face Recognition and Scene Analysis

Dissertation zur Erlangung des Grades
eines Doktors der Naturwissenschaften
in der Fakultät für Physik und Astronomie
der Ruhr-Universität Bochum

vorgelegt von

Laurenz Wiskott

Juli 1995

Published as `Wiskott, L. (1995). Labeled Graphs and Dynamic Link Matching for Face Recognition and Scene Analysis. Verlag Harri Deutsch, Thun - Frankfurt am Main, Reihe Physik 53 (PhD thesis), ISBN 3-8171-1481-8.`
Page layout and numbers of this reprint are identical to the original ones.

Abstract

In many neural net applications visual data are represented as vectors, although it is known that this form of representation lacks syntactical structure. Labeled graphs have been proposed as a data format which provides the missing relational information. The present work argues that labeled graphs of perceptual patterns can be generated and processed based on simple principles. Complex and flexible object representations can be derived from single examples by graph matching.

Dynamic Link Matching has been developed as a biologically-motivated neural mechanism for graph matching. This work discusses the principles as well as the advantages and drawbacks of Dynamic Link Matching compared to other neural systems. A complete face recognition system based on Dynamic Link Matching is developed. In contrast to previous systems, the dynamics is autonomous, and matching between graphs of different size is made possible by an attention window. The performance is demonstrated for faces of different perspective or facial expressions against a gallery of 111 neutral frontal views.

For more technical applications, Elastic Graph Matching has been developed as an algorithmic counterpart to Dynamic Link Matching. In this work the system is developed further in several aspects: object-adapted graphs allow comparisons between very different views, efficiency has been increased significantly by separating graph generation from recognition, and phase information of the Gabor transform is used to increase matching accuracy. The key role is played by a newly introduced graph structure, called General Face Knowledge. It is based on a collection of individual sample faces, but it also represents faces that can be obtained by combining subparts of the samples. By this means, new faces can be processed without having a reference model of the individual person. Recognition results on galleries of 300 faces are presented.

The determination of facial attributes serves as a second demonstration. General Face Knowledge can be used to generate composite or phantom faces very similar to the original. If facial attributes such as gender or the presence of a beard are known for the sample faces of the General Face Knowledge, these attributes can be transfered to the phantom face. On that basis the facial attributes of the original face can be determined in a very simple and general way.

And finally, Elastic Graph Matching is applied to the recognition of occluded objects in cluttered scenes. Two different algorithms are presented. The first allows recognition of known objects between and behind unknown distractors. The second one requires that all objects in the scene are known to the system. It processes the scene from front to back and in addition determines the order of the objects in depth.

Preface

It is not the intention of this thesis to present one single monolithic model of object recognition, but rather several models, emphasizing different aspects of a larger conceptual framework. I have therefore tried to keep the different chapters independent of each other, making it possible to select single chapters without having read the preceding ones. The abstracts at the beginning of each chapter will help to provide an overview. Keywords referred to in the index are printed in italics.

The first chapters introduce the conceptual framework. In Chapter 2, I argue for labeled graphs as a data structure for object representation. In Chapter 3, the principles of Dynamic Link Matching as a neural system for processing labeled graphs are explained. The four subsequent chapters present four concrete models for different visual tasks on two different levels of abstraction. Chapter 4 deals with Dynamic Link Matching as applied to face recognition. The same task is solved in Chapter 5, but in a more technical system with Elastic Graph Matching, which is an algorithmic abstraction of Dynamic Link Matching. Closely related is Chapter 6, which is concerned with the determination of facial attributes such as gender, the presence of a beard or glasses. A very different application of Elastic Graph Matching, the analysis of cluttered scenes, is then presented in Chapter 7. (This chapter is in part a modified reprint of (Wiskott & von der Malsburg, 1993) ©World Scientific Publishing Co. Pte. Ltd., with the kind permission of the publisher.) All applications here use the Gabor wavelet transform as a visual preprocessing providing local features (see Appendix A).

I have been fortunate to be able to work with Professor von der Malsburg's group at the *Institut für Neuroinformatik* of the *Ruhr-Universität Bochum*, Germany and to visit several times his group in the *Department of Computer Science* and the *Section for Neurobiology* at the *University of Southern California*, Los Angeles, U.S.A. I enjoyed ideal working conditions and great freedom to develop my own interests and ideas. I am especially obliged to my advisor, Professor von der Malsburg, who has taught me in the course of many invaluable discussions what kind of questions are worth raising and what principles might lead to answers. I am also grateful to Professor Biederman at USC, who taught me many things about psychophysics. I thank Professor Wunner for providing the second report. Sincere thanks go also to my colleagues in Bochum and at the USC: Jean–Marc Fellous, Norbert Krüger, and Thomas Maurer, with whom I shared exciting cooperation within the face recognition project at USC; Martin Lades for his sofware contribution to the cluttered scenes project; Norbert Krüger, Thomas Maurer, Michael Pötzsch, and Andreas Schwarz for their critical remarks on the manuscript of this thesis; Jozsef Fiser, Bernd Fritzke, Wolfgang Konen, Jan Vorbrüggen, Rolf Würtz, and all those mentioned above, for many fruitful discussions and an enjoyable time in Bochum and at the USC. I thank Michael Neef for providing us with a well maintained

computer environment, Uta Schwalm for her help with all administrative questions, and Professor Edwin Hopkins and Jan Vorbrüggen for their corrections of the English. Finally, I especially thank my wife Julia for her support.

The system presented in Chapter 4 was simulated with NSL (Neural Simulation Language) developed by Alfredo Weitzenfeld. All other simulations were made using GNU sofware. The thesis was typeset in LaTeX. I would like to thank the authors of these convenient and free sofware tools.

This work was financially supported by the *German Federal Ministry for Science and Technology (BMFT)* (01 IN 101 B/9; *Neuronale Architekturprinzipien für selbstorganisierende mobile Systeme*). The work for Chapters 4 and 5 were supported by AFOSR (F49620-93-1-0109) and ARL/ARPA (01/93/K-0109) respectively.

Bochum, July 1995 Laurenz Wiskott

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Visual images are usually represented as pixel arrays, a square lattice of real numbers. Two indices, the x- and y-coordinates, denote the position, and the real number represents the local grey value. This is an appropriate representation for raw images, and it is complete within the limits of given spatial and brightness resolution. Conventional neural net applications tend to ignore the spatial relations implicit in the x- and y-coordinates. They simply use the indices as a unique address for the individual pixels or the respective input neurons. An image is then represented as a *vector* with distinct coefficients without further spatial relations. One could for example consistently permute the vector coefficients in all training and test patterns, and the network would perform as well as before.

Character recognition is a frequently used example to demonstrate the performance of a neural system. It was already used in the early years of modelling neural nets. Widrow & Hoff (1960), Block (1962), as well as Kohonen (1972) used character recognition for demonstration purposes. In all three cases the input units had no further relational structure, i.e. the input patterns had no *topology*. The consequence is demonstrated in Figure 1.1. On the left and on the right are shown two patterns that we assume have been learned by a neural net or stored by an associative memory. In the middle is shown a pattern that serves as an input and is supposed to be recognized or associated with one of the stored patterns. Which of the stored patterns is the correct one? The natural answer seems to be: the right one. But then one has used a metric which takes relational information into account. The neural nets mentioned above consider the patterns as vectors and use a different metric. The Hamming distance, for example, leads to the result that the left pattern is more similar to the one in the center, since the left pattern differs in fewer pixels from the central one. The same paradigm of treating the input pixels without any relational structure can be found in more recent applications (Kohonen, 1987; Kosko, 1987; de Edson et al., 1990).

Another example is shown in Figure 1.2. What is the common property of the top four patterns? The same question applies to the bottom four patterns: What is their common property? The answer to these questions depends very much on whether one takes into account the spatial structure of the patterns or not. With topology (see top row) one can already see from one example that it shows symmetry with respect to a diagonal, and this holds for the other patterns in the row as well. Ignoring the topology can be illustrated by permuting all pixels consistently, i.e. same permutation for all patterns (see bottom row). Then one has a hard time and would need many more

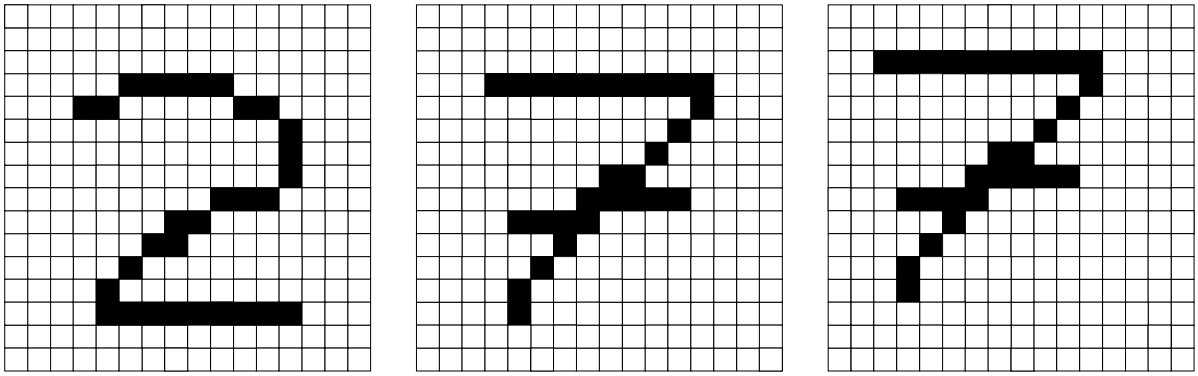

Figure 1.1: Visual patterns as used for training and testing conventional neural nets. Is the pattern in the center more similar to the left or to the right pattern?

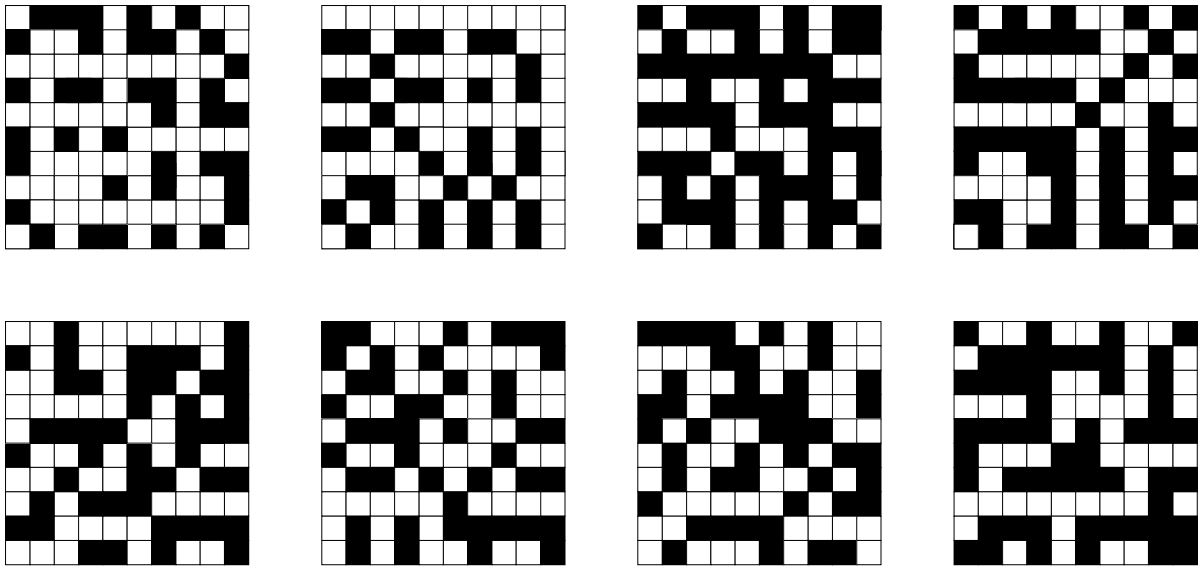

Figure 1.2: Patterns with second order correlations. What is the common property of the patterns in the top row? And what is the common property of the patterns in the bottom row? (The bottom patterns have been generated from symmetrical patterns by a constant permutation of the pixel positions.)

---

examples to notice that many pairs of cells are perfectly correlated. Sejnowski et al. (1986) have used this example. They applied a Bolzmann learning algorithm to the detection of different symmetries. For a 10×10 layer the algorithm needed about 40,000 presentations of *training* examples in order to reach a success level of 85%. The hidden units had to reveal through statistics that certain pairs of neurons were correlated, a property completely independent of the spatial arrangement of the correlated pairs. It is evident that the notion of symmetry can be inferred from few examples if the spatial structure of the patterns is taken into account. This was demonstrated by Konen & von der Malsburg (1992, 1993). In face recognition the most prominent example of representing visual patterns as vectors without any topological structure is the *Principal Component Analysis* directly applied to face images (see for example Kirby & Sirovich, 1990; Turk & Pentland, 1991; O'Toole et al., 1993).

Two different solutions are used to overcome this drawback of the conventional neural net paradigm. The first one is to compensate for the translation sensitivity of the vector representation by *preprocessing* the input images to get a normalized version of them, which is centered and possibly rescaled (KIDDER & SELIGSON, 1993; AVIITZHAK et al., 1995). This is frequently done in a rather technical way and is then not part of the neural system. A more neural system for translation correcting preprocessing is the translation-invariant network (WIDROW et al., 1988; MAO & KUO, 1992). But all these preprocessing systems require presegmented patterns and do usually not account for distortions. A second solution was demonstrated by FUKUSHIMA et al. (1983). Their *Neocognitron* is a multilayer feed-forward network with receptive fields that are restricted to a small region of the respective input layer. By this means some topological information is introduced, since neighboring neurons in the input layer usually belong to the same receptive fields. Within the receptive fields no further spatial relations are encoded. This connectivity repeats over several stages. When the Neocognitron is trained, the neurons develop into more and more complex and more and more translation-invariant feature detectors as one ascends the *hierarchy*. Translation invariance is achieved by low-pass filtering and subsampling the neural responses at each stage. Thus positional information that might be important for discrimination is lost and the performance of the system potentially degrades. In addition, the degree of possible low-pass filtering is tightly coupled to the complexity of the features, and translation invariance is therefore limited. A further disadvantage of this system is that it requires a very careful architectural design and selection of training patterns; *training effort* is very high. A more recent application of the Neocognitron can be found in (TING & CHUANG, 1993). Related to the Neocognitron is the weight-sharing back-propagation network (LECUN et al., 1989; MARTIN, 1993). These systems show only very little translation invariance. They are also very expensive in terms of training samples. However, they seem to be more robust in terms of architectural design, and they do not require training of each layer separately, as is necessary for the Neocognitron.

These examples illustrate the lack of structural information in the vector representation (cf. VON DER MALSBURG, 1981; VON DER MALSBURG, 1986; BIENENSTOCK & DOURSAT, 1991). This thesis is concerned with an alternative representation of visual patterns, the *labeled graph*. It combines feature information with the required structural information. The advantages of labeled graphs will be discussed in Chapter 2 and then demonstrated in different applications, for face recognition (Chapter 5), gender determination (Chapter 6), and scene analysis (Chapter 7). Although labeled gaphs are a very natural representation for visual patterns, they do not quite fit into the traditional concept of neural nets with a fixed connectivity, subject only to a slow learning process. VON DER MALSBURG (1981) therefore proposed the *Dynamic Link Architecture*, in which he enriched conventional neural nets with the concepts of *temporal binding* and *fast synaptic plasticity*. Based on these ideas BIENENSTOCK & VON DER MALSBURG (1987) developed *Dynamic Link Matching*, in which temporal structure of neural signals codes for relations between nodes by correlations. Fast synaptic plasticity admits dynamic matching between different layers depending on the represented patterns. The connectivity is no longer fixed, but subject to a complex self-organization process. Dynamic Link Matching will be discussed in Chapter 3 and applied to face recognition in Chapter 4.

Chapter 2

Labeled Graphs for Object Representation

Abstract: Sensory patterns can be appropriately represented by labeled graphs. Nodes are labeled with local features; edges are labeled with relational features. It is argued that these labeled graphs can be generated on the basis of simple grouping principles: proximity in feature type, in space, or in time. They can be matched onto each other if they are similar in features and structure. Fusion graphs can be generated for graphs with significant overlap. Labeled graphs provide a means to learn and generalize from single examples and might serve as a basis for more abstract processes such as finding analogies.

2.1 Introduction

In the introductory chapter it was argued that the typical data structure used in artificial neural nets, the vector, lacks relational information. In this chapter I describe labeled graphs as a uniform data format for sensory patterns providing the structure needed. A simple example of labeled graphs and graph matching is shown in Figure 2.1.

Labeled graphs for object representation have been widely used in the field of artificial intelligence (cf. Fu, 1982), but for neural nets they were probably first proposed by von der Malsburg (1981, 1983). He also developed Dynamic Link Matching as a neural mechanism to process labeled graphs in a neural architecture (see next chapter). Several concrete models based on these concepts will be presented in the subsequent chapters. A final conclusion will be given in Chapter 8.

2.2 Representation of Sensory Patterns

On a low level, *perception* begins with a structured encoding of the sensory input. Primitive segmentation and grouping mechanisms are necessary to provide higher levels with a useful representation of objects. I am now going to discuss briefly the general structure of *sensory patterns* and how elementary graph representations of objects can be generated.

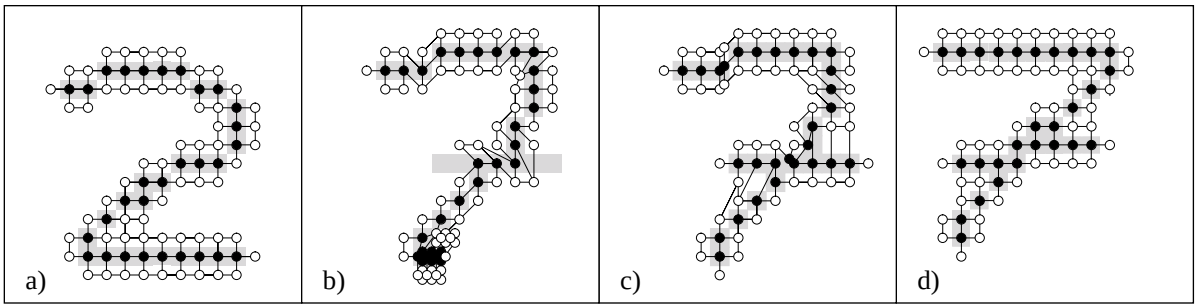

Figure 2.1: Simple graphs and graph matching. **a)** and **d)** The patterns '2' and '7' are the same as in Figure 1.1, but represented and stored as labeled graphs. The input pattern to be recognized is a different '7', distorted relative to the stored one. **b)** The '2'-graph matched to the input pattern. The curved line and diagonal of the '2' fits fairly well, but the horizontal foot of the '2' is completely compressed and the horizontal stroke of the '7' is not covered. **c)** The '7'-graph can be matched to the input pattern with little distortion (adapted from Bienenstock & Doursat, 1991).

2.2.1 Labeled Graphs of Sensory Patterns

A *labeled graph* consists of a set of nodes and a set of edges connecting pairs of nodes. The nodes are labeled with *local features*, and the edges are labeled with *relational features* (von der Malsburg, 1986; Bienenstock & von der Malsburg, 1987; Lades et al., 1993; see also Figure 2.2). A local feature represents absolute information that can be extracted from a small patch of an image such as color, local texture, or the orientation of an edge. For acoustic signals local features could be onset, offset, or energy in a particular frequency channel. I will refer to the complete set of local features for a given *modality* as the *feature space*. The relational features, on the other hand, can only be extracted from two such local patches. An example is the spatial distance between two locations. With one exception (see below) I will refer to the space from which the relational features are extracted as the *sensory space*. Sensory space depends on modality, too. For the auditory modality, for instance, the sensory space is the frequency axis. The exception to this is *time*. Time also provides relational features such as 'sooner' or 'later', but nevertheless time needs to be treated separately from the sensory spaces for several reasons. First of all, time is common to all modalities and thus constitutes the main cue for binding percepts of different modalities. Secondly, time cannot be represented in the same way as the other sensory spaces. There is no counterpart to the retina for the time dimension. Further peculiarities of time perception have been discussed by Pöppel (1978). Feature space, sensory space, and time can be considered as three subspaces in which sensory patterns are embedded, and we have seen how labeled graphs can serve as a discrete representation of such sensory patterns.

2.2.2 Graph Formation

How can a graph of a sensory pattern be generated? This process has two aspects: Firstly, nodes need to be located, and secondly, they need to be connected by edges. The sensory input, for example a pixel image, often has a much higher resolution than one would like to represent internally, locating a node at each pixel not being practical.

Thus one has to perform a selection. The appropriate density of nodes depends on the complexity and spatial extent of the local features. With complex features describing extended patches of an image, the nodes may be much sparser than for simpler features describing only a few pixels around each node. Two schedules for selecting nodes in a sensory pattern have been used. The first is to select nodes on a regular grid with constant spacing adapted to the complexity and extent of the local features (cf. Würtz, 1995). This schedule does not account at all for the specific character of patterns and is determined only by the characteristics of the local features and the spatial resolution that one wants to achieve. In the second schedule one attempts to select particular points in the sensory pattern, so-called *salient points*, that are especially important and carry maximal information. The problem is that these points have to be selected without object knowledge only on the basis of low-level information. Thus, one has to define an appropriate saliency measurement that allows one to find salient points with high reliability (cf. Manjunath et al., 1992). This approach has the advantage that fewer nodes are required and that the node positions are *object-adapted*, i.e. for the same object in different images it is likely that the same locations relative to the object are selected. It should be mentioned that several nodes may be located at the same image location if different feature types are used. Thus, for a red dot there might be a node representing 'dot' and another node representing 'red'. Using several nodes at the same location may also be appropriate for representing transparent objects.

When the node positions are selected, which nodes should be connected? This is the question of *grouping* and *segmentation*, and many mechanisms have been proposed for it (cf. the Gestalt principles in Boff et al., 1986, pp. 36-14–36-23). I concentrate only on the law of *proximity* in three different variations. It may be proximity in sensory space (*spatial proximity*), proximity in feature space (*feature similarity*), and proximity in time (*temporal proximity*). The probability of two nodes being connected increases with proximity and will be maximal if two nodes coincide in some aspects (see Figure 2.2). Neural models of grouping and segmentation based on spatial proximity and feature similarity were, for instance, presented by König & Schillen (1991), von der Malsburg & Buhmann (1992), and Vorbrüggen (1994). It is interesting that in these models the segmentation result was represented by temporal synchrony of coupled oscillators corresponding to the third mechanism, temporal proximity.

These are only three basic grouping mechanisms, and many more could be cited. However, it is unclear to what extent other, more complex laws of grouping are necessary under natural conditions and to what extent they could be learned from experience on the basis of the simple mechanisms cited here (Boff et al., 1986, pp. 36-11–36-14).

Once elementary node relations are induced by the laws of proximity, sets of nodes are connected to form a graph. The reasons for which the nodes were connected might in part be irrelevant to the object itself. For example, common motion is a strong segmentation cue and will bind together all nodes of one object, but the motion itself is usually not relevant for the object and should not be stored. Therefore the connections induced have to be transferred to all nodes. The result is a highly connected elementary graph, representing a sensory pattern. This is the basis for all further steps (see Figure 2.3).

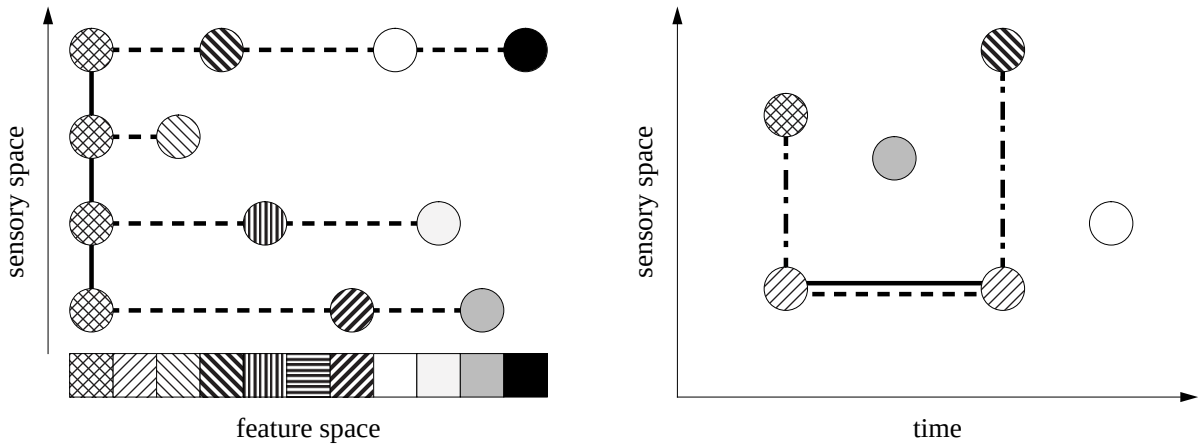

Figure 2.2: Sensory patterns are embedded in sensory space (e.g. retinal location), feature space (e.g. color), and in time. Labeled graphs are used to represent sensory patterns. Nodes are labeled with local features. Proximity in one of the three subspaces induces relations between nodes, which are represented by edges. Proximity in sensory space is indicated by dashed lines, proximity in feature space is indicated by solid lines, and proximity in time is indicated by dash-dotted lines. In the left example the leftmost feature type might, for example, represent common motion. In the right example, two nodes are connected due to proximity in sensory *and* feature space. Two other nodes are isolated, since they are proximate to none of the other nodes.

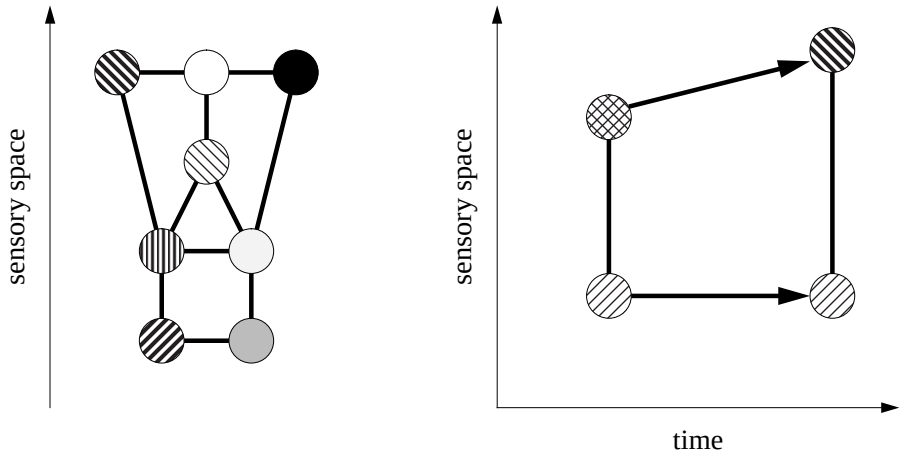

Figure 2.3: The edges of the graphs in Figure 2.2 induced by proximity are transferred to all nodes of the graph. Some nodes, such as those representing motion, are significant for segmentation but are irrelevant for representation of the object itself and can be removed. The result is a highly interconnected labeled graph representing the sensory pattern. Connections expressing temporal progression are indicated by arrows.

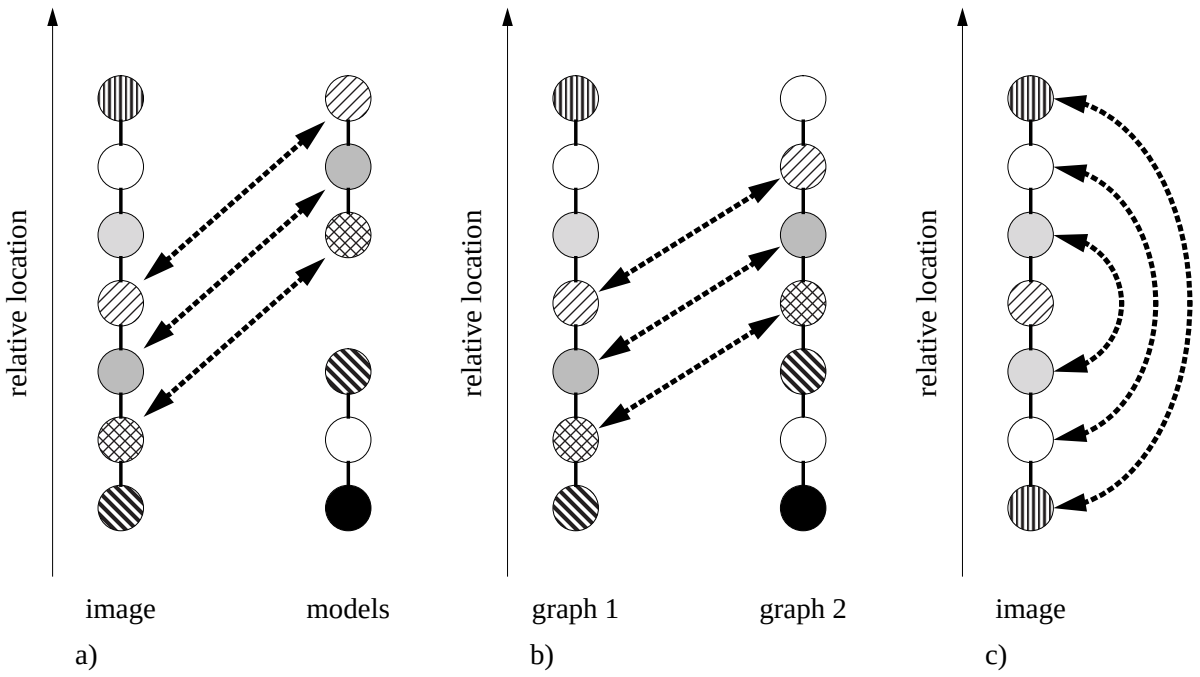

Figure 2.4: Graph matching: The task of graph matching is to connect graphs and subgraphs that are similar in features and structure. The match between the graphs is indicated by dotted arrows. The three examples illustrate **a)** object recognition, **b)** finding partial identity, and **c)** detecting symmetry.

2.3 Graph Matching

As a next step after generating graph representations for sensory patterns, one would like to compare them. The elementary process for that is *graph matching* and the determination of the *similarity* between graphs.

2.3.1 Pattern Recognition

Comparing graphs requires finding a *mapping* that connects nodes of one graph with nodes of another under the *constraints* firstly that the *topology* is preserved, i.e. neighbors are connected with neighbors, and secondly that the similarity between the features of connected nodes is high. This process is called *graph matching* (see Figure 2.4, cf. also Bienenstock & von der Malsburg, 1987; von der Malsburg, 1988; Kohonen et al., 1994, and Chapter 4 for neural models of graph matching). If only parts of two graphs are similar, these subgraphs should be matched onto each other while the remaining parts remain unconnected (see Reiser, 1991, and Chapter 7). This kind of graph and subgraph matching is in general *NP-complete*, but since the graphs have a topographical structure and are embedded in a low-dimensional sensory space, the complexity is reduced significantly, and simple matching schedules find good approximations of the optimal match in a reasonable amount of time (see Buhmann et al., 1989, 1992, and Chapter 5).

2.3.2 Finding Analogies

The graph matching described above can be used to compare graphs within the same modality and result in high similarity if the sensory patterns are similar. What can be said about two *graphs of different modality*? What remains comparable? This is first of all, the number of nodes, the presence or absence of connections between pairs of nodes, and relational information about temporal order. Secondly, in all sensory spaces two nodes may be close or distant to each other. For instance, a dot in a visual pattern may be closer to a second one than to a third, corresponding to three sound components of pitch C, D, and G in the auditory system. Hence the topology of patterns in different modalities may be comparable, although this has quite obvious limitations, e.g. a triangle cannot be represented on the frequency axis. Thirdly, intensity as a very general aspect of feature space may also be comparable between modalities, e.g. brightness, loudness, and pressure in the visual, auditory, and tactile system respectively. Graph matching based mainly on relational information can be interpreted as a mechanism for finding *analogies* (see Figure 2.5). This may be of questionable advantage on the low level of sensory patterns, but the idea of finding analogies between graphs on the basis of structural similarities might be interesting if one thinks of *abstract graphs* representing more abstract patterns such as trajectories, language, or ideas. A simple system for finding analogies on the basis of relational structure was presented by Chalmers et al. (1992). Their system could for example recover the analogy between the strings 'ppqrss' and 'aamnxx', mapping 'pp' to 'aa', 'qr' to 'mn', and 'ss' to 'xx'. It was based on features such as 'first', 'last', 'successor', and 'opposite'.

2.4 Fusion Graphs

If one has a method for comparing graphs, one can improve storage of a number of graphs by representing identical or similar graphs or subgraphs only once. The result is a *fusion graph* (Reiser, 1991; see Figure 2.6). This has the advantage of reducing the required storage capacity and it possibly helps to *generalize*. But one possible risk is that the fusion graph will overgeneralize, and that the stored examples cannot be recalled reliably. There are different mechanisms for avoiding this and obtaining recall of single examples.

2.4.1 Long Range Connections

Long range connections between different parts of a fusion graph can *disambiguate* the possible interpretations in order to recall the stored samples reliably (see Figure 2.6.c). A consistent realization of this idea, where all nodes are connected with all others, was presented by von der Malsburg (1985, 1988) and von der Malsburg & Bienenstock (1987). The attractiveness of this approach comes from its homogeneity and the fact that it requires no further nodes and probably little control structure. New graphs can be integrated into the fusion graph very naturally. The disadvantage is that generalization capabilities are lost. From a functional point of view, the system behaves like a collection of single graphs.

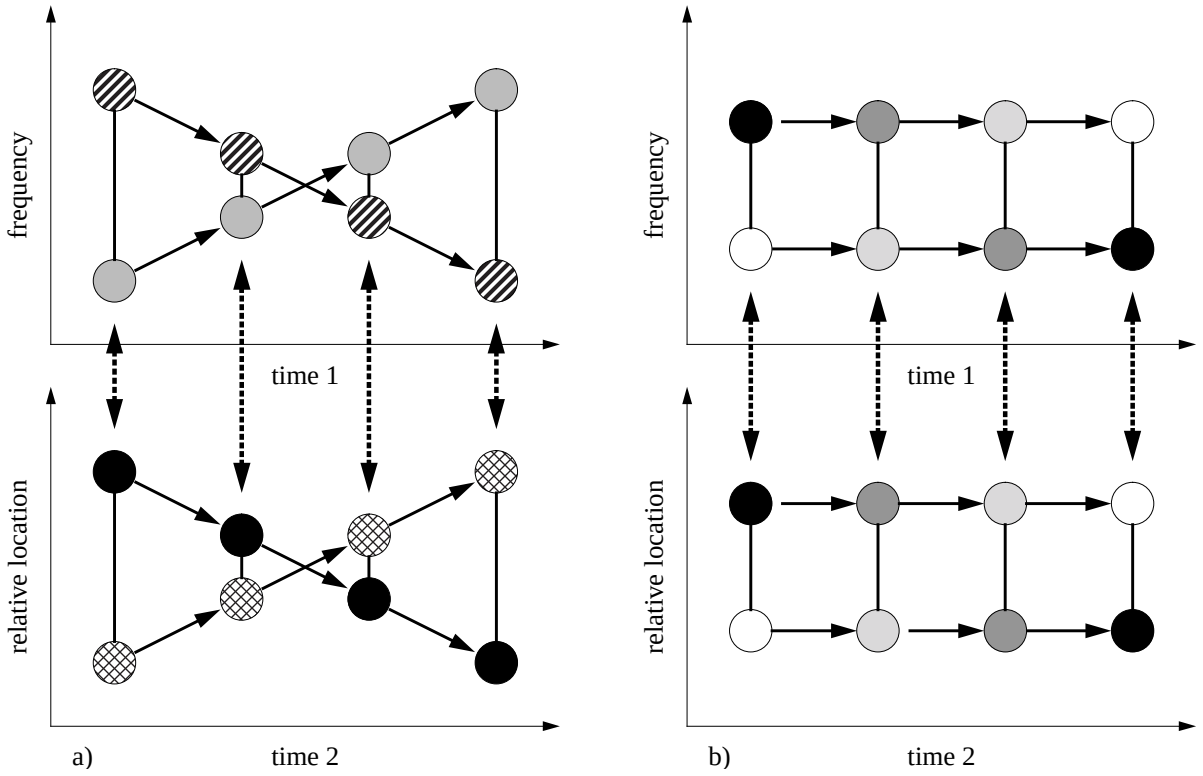

Figure 2.5: Two analogies between graphs of different modalities. The matching in this case is based on relational patterns rather than feature patterns. In **a)** the features of the nodes are completely ignored. It is only the structural information that leads to the illustrated matching. The upper pattern may represent two tones, one going down in pitch, the other up. The other pattern may represent the visual analogy, say two dots of different color, one moving downwards, the other upwards. One can also imagine that simple features such as intensity are compatible between modalities, as shown in **b)**. The upper graph may represent two tones, the lower one two lights. In both graphs one signal becomes more intense and one less.

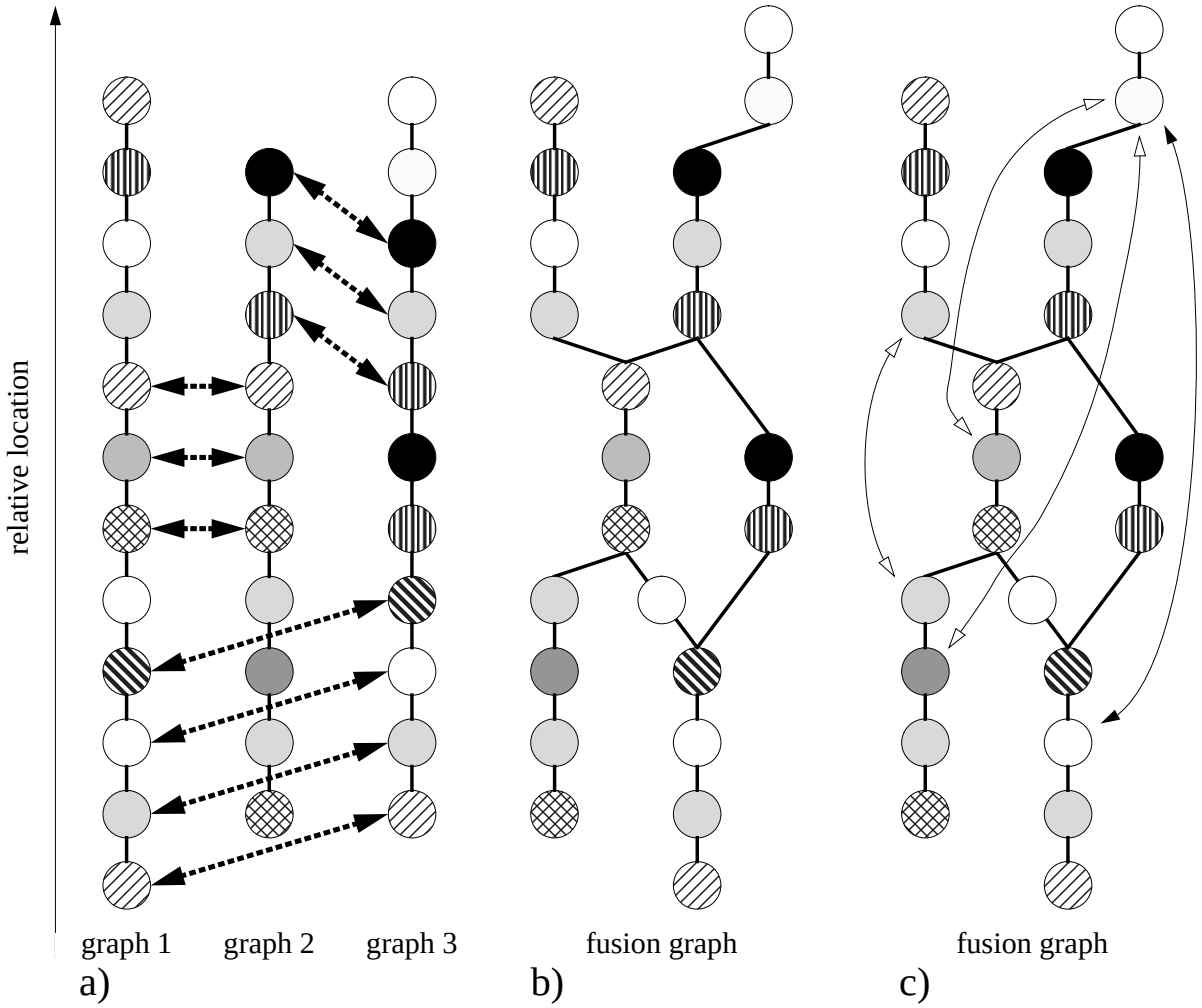

Figure 2.6: Fusion graphs: **a)** Three graphs with similar subpatterns. **b)** Fused graph representing common subpatterns only once. Since it is not encoded which subpatterns to combine, this representation is ambiguous, and combinations of subpatterns not presented in a) are valid as well. This can be considered as a generalization capability. **c)** Ambiguities can be resolved by long range connections. Filled arrows indicate that the parts belong together, while unfilled arrows indicate that the parts are unlikely to belong to the same object.

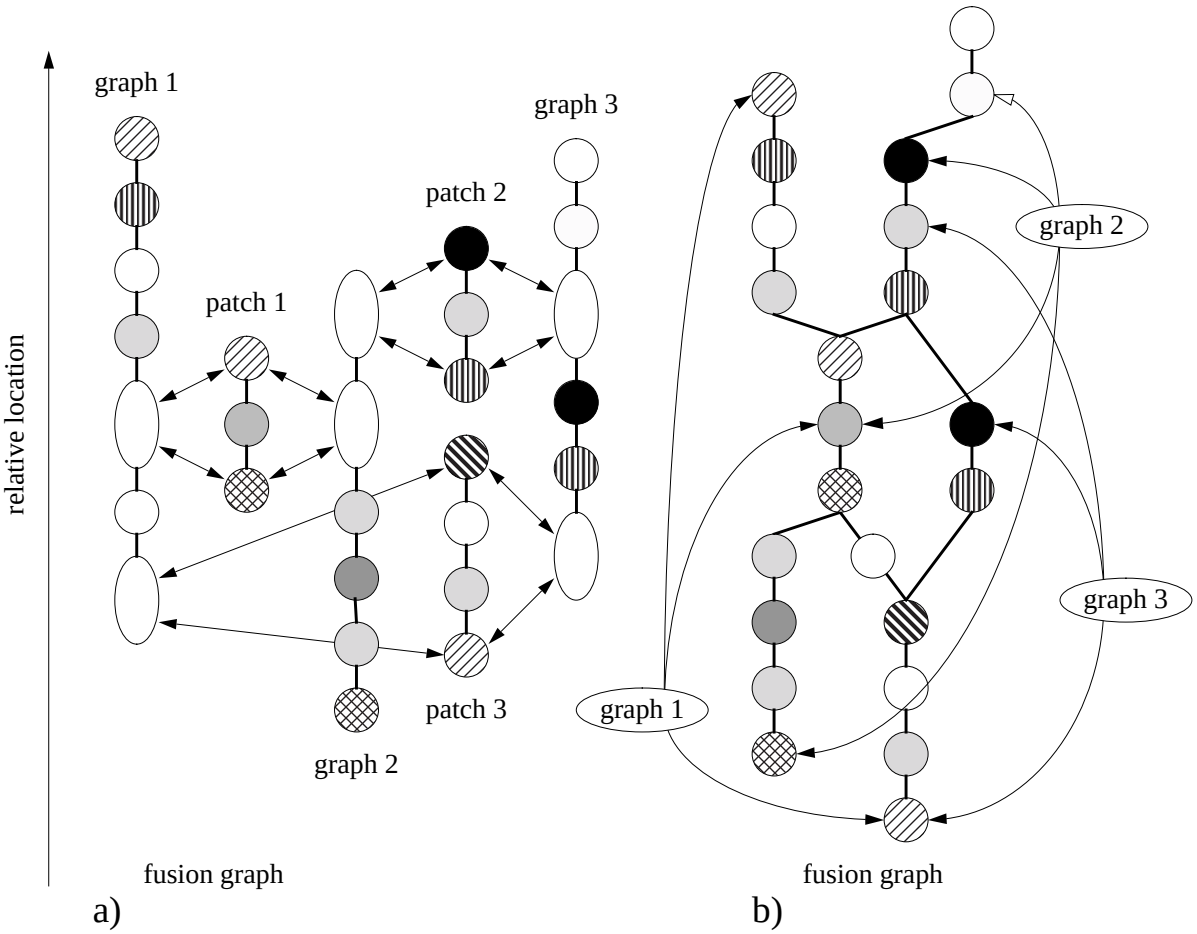

Figure 2.7: Two alternative ways to resolve the ambiguities of the fusion graph in Figure 2.6.b: **a)** Cardinal cells represent the common subpatterns or **b)** they represent entire graphs and point to parts of the fusion graph.

2.4.2 Cardinal Cells for Subpatterns

An alternative concept is to represent common subpatterns by *cardinal cells* (see Figure 2.7.a). Each graph then would consist of several elementary nodes and possibly some cardinal nodes that point to subpatterns. From a functional point of view this solution would again be equivalent to a collection of single graphs and would lack the potential for generalization. Compared to the previous solution, this method requires more nodes and a sophisticated method to determine which subpatterns are frequent enough to be worth representing by a cardinal node. A further problem would be to code exactly how a subpattern is to be integrated into the graph (cf. Ehrig, 1991).

2.4.3 Cardinal Cells for Whole Graphs

A more reasonable possibility of combining generalization abilities and a reliable recall of single examples is the use of cardinal cells for whole graphs (see Figure 2.7.b). The great advantage is that they provide control. If they are all inactive, the fusion graph shows full generalization properties. One can activate one of them and suppress all others recalling only the graph of this one sample. For a recognition task, one can apply

a winner-take-all mechanism between the cardinal cells, ensuring that the best fitting sample will win, without allowing spurious states.

2.5 Specific Graph Representations

Having discussed the general concept of labeled graphs for object recognition, I will now describe the concrete graph structures that I have used for the different applications presented in Chapters 4–7.

2.5.1 Face Recognition

For *face recognition* each face is represented by a single graph. A collection of these *model graphs* serves as a *gallery*. For a new image of a face a new *image graph* has to be generated, which can then be compared with the gallery. The most similar model is taken as the correct face. The image graph formation can be achieved by matching any stored model graph to the image. Assume the nodes in the model graph were taken from so-called *fiducial points*, e.g. eyes, tip of the nose, corners of the mouth, etc. If the model face is similar to the image face, the model nodes are likely to be correctly matched to the corresponding fiducial points in the image. Problems arise if the two faces are not similar, for example, due to a beard or glasses. Because of the very different appearance of identical fiducial points, the image graph formation might fail. Thus, instead of a single model graph one needs a more general representation of faces that can be matched to an image in order to find the fiducial points for different faces reliably.

For this purpose I have introduced a *general face knowledge* (GFK). Since all faces have the same structure, only one graph representing the facial geometry is required with one node at each fiducial point. But since the individual parts of faces may look different, each node of this graph is connected with a set of alternative local descriptions of the respective fiducial point, e.g. at the eye there are some descriptions for female eyes, others for male eyes, some for closed eyes, some for eyes with glasses, etc. Each node has its own set, and during the matching process, the description that fits the fiducial point in the image best is selected. Due to its combinatorial power (each node may select a local description independently of the others), the GFK potentially represents a wide range of different faces (see Chapter 5).

2.5.2 Determination of Facial Attributes

For the *determination of facial attributes* the general face knowledge has to be enriched by *context knowledge*. This can be done by cardinal nodes indicating the context of the face, for instance male or female, bearded or not, and whether or not the person wears glasses (see Figure 2.8.a). By comparing a new image graph with the GFK, the context knowledge can be used to determine the facial attributes of the image face (see Chapter 6).

2.5.3 Objects in a Scene

The situation is very different for *object recognition* in cluttered *scenes*. Objects are in general very different in structure; thus each object has to be represented by a different

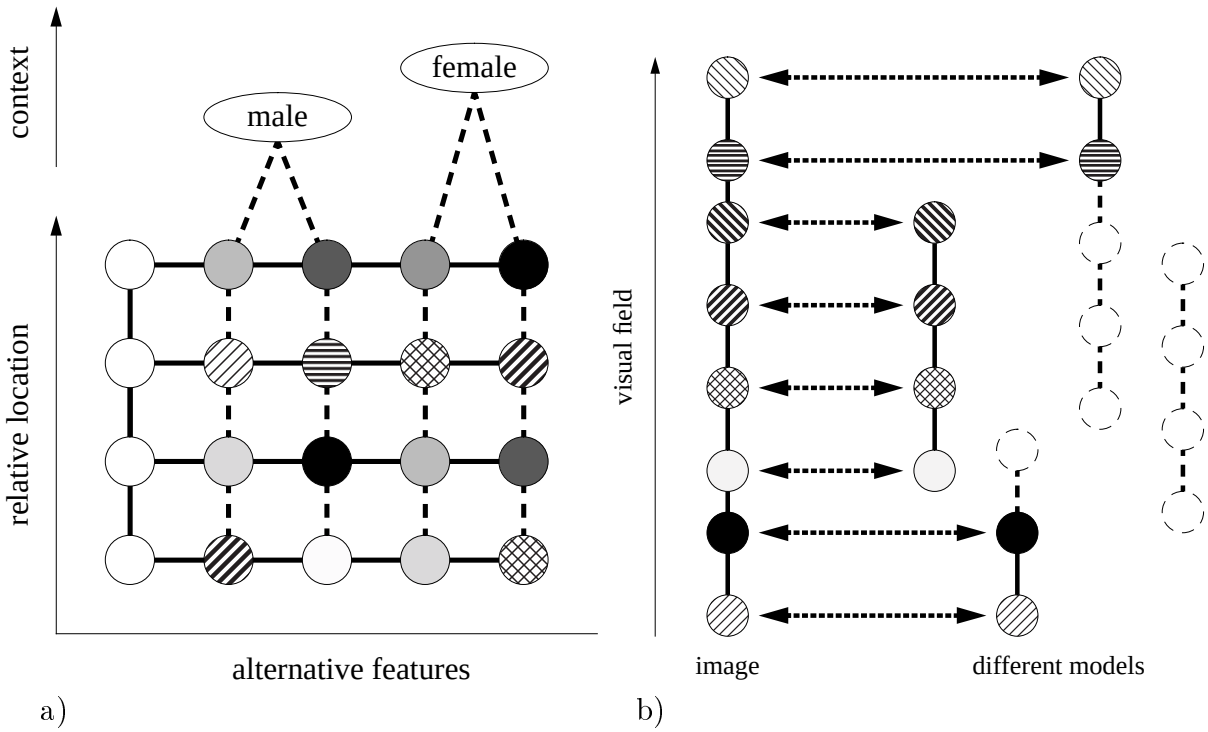

Figure 2.8: a) Faces all have the same general structure, but they differ in local features. For a general face representation it is therefore reasonable to store for each facial point a whole set of different local features that might be applicable alternatively, indicated as different nodes connected to one blank node on the left. In addition, context knowledge may group the local features into classes, for example male nodes versus female ones. **b)** In order to interpret the image of a cluttered scene, several models have to be matched to the image and in addition the order in depth has to be determined. Models compete with each other for image space and cooperate with the image. Models or parts of models are deactivated due to occlusion, as shown by blank features and dashed lines.

---

object graph, and each graph is matched to the image individually. Particular problems arise from the fact that objects may occlude each other. As part of the matching process, the system has to decide which regions of an object are occluded by others and which are visible. A further constraint comes from the fact that objects cannot intersect. This can be taken into account by enforcing a definite order in depth. If two model graphs overlap in a scene, one has to be completely in front of the other. All this leads to a complicated interaction between all model graphs matched to a scene. As shown in Figure 2.8.b, the models *compete* with each other for image space, while mutual intersection is prohibited (see also Chapter 7).

The concepts presented in this chapter will be extended and illustrated in the subsequent chapters. First, I will discuss Dynamic Link Matching as a neural mechanism to treat labeled graphs, and then present several illustrative applications. A conclusion will be given in Chapter 8.

Chapter 3

Dynamic Link Matching

Abstract: In a neural system, labeled graphs can be represented by layers of neurons. Dynamic Link Matching (DLM) is a mechanism to match such layers onto each other if their feature patterns are similar. Dynamic Link Matching has four basic principles: the single layer dynamics induces correlations encoding neighborhood; the dynamics of two layers synchronize with respect to the common feature pattern; this synchrony is robust against noise and accidental links; fast synaptic plasticity rules out all accidental links and establishes a regular connectivity between the two layers. Layer synchrony and connectivity improve in an iterative process. Along with its potential for graph matching and object recognition, invariant under translation, rotation, and mirror reflection, Dynamic Link Matching has two major drawbacks: it is slow and it is expensive in terms of connectivity.

3.1 Introduction

In Chapter 2 labeled graphs were presented as a uniform data structure for representing perceptual patterns. This Chapter 3 serves as a conceptual discussion of Dynamic Link Matching (DLM), which is a neural dynamics for matching labeled graphs. A concrete model of DLM is presented in Chapter 4.

Since conventional neural nets allow the synaptic weights to change only on the slower time scale of learning and not during a recognition task, they do not seem to be a natural architecture to deal with graphs, which require dynamic binding on a fast time scale in order to account for the relational information. They are much more appropriate for vectors. DLM is one of very few attempts to overcome this restriction and to match patterns represented as graphs. Such a system has to be at least translation invariant. In addition, it is rotation invariant and robust against distortion. Scale invariance can in principle be achieved in a multiscale representation, but this has not yet been demonstrated. Before I describe the task and principles of DLM I would like first to give a short historical overview of the development of DLM.

3.1.1 History

Related to the problem of graph matching is the question of how a *retinotopic projection* can self-organize during the weeks or months of ontogenesis. In both cases a regular *mapping* between two domains has to be established which preserves neighborhood relations,

i.e. neighbors are connected with neighbors. A model for the ontogenesis of retinotopic projections was proposed by WILLSHAW & VON DER MALSBURG (1976). This model is formulated in terms of neural activity. The very same principle, but formulated in terms of chemical markers, was demonstrated in (VON DER MALSBURG & WILLSHAW, 1977; WILLSHAW & VON DER MALSBURG, 1979).

From these models KOHONEN (1982) derived his algorithm for the self-organization of topographical feature maps. AMARI (1980, 1989) did a thorough analytical treatment of layer dynamics and formation of topographical maps. HÄUSSLER & VON DER MALSBURG (1983) derived autonomous equations for the *link dynamics* independent of the specific layer dynamics.

Regular projection patterns can also self-organize between other than two-dimensional domains. Different and more complicated structures are possible as well. The self-organization of hypercolumns of orientation selective cells develops a mapping between a circular one-dimensional pattern space and a cortical structure of two dimensions (VON DER MALSBURG, 1973). In case of ocular dominance stripes, two two-dimensional structures, the left and right eye, compete with each other for one two-dimensional structure in the cortex (VON DER MALSBURG, 1979). In both cases the conflict in topography leads to pattern formation: hypercolumns or ocular dominance stripes.

VON DER MALSBURG (1981, 1983, 1986) generalized the ideas of the retinotopy-related models and proposed to apply the same principles to visual recognition tasks. He introduced *labeled graphs* and formulated the idea of DLM as the neural realization of *graph matching*. The main differences between the retinotopy model and DLM is that the latter process is guided by features and their similarities, while the former has only the constraint of preserving neighborhood relations. A second difference is that DLM has to take place on a much faster *time scale* — a fraction of a second — while the other may take weeks.

VON DER MALSBURG and BIENENSTOCK presented first model simulations for the retrieval of stored graph structures (VON DER MALSBURG, 1985; BIENENSTOCK & VON DER MALSBURG, 1987) and for pattern recognition by DLM (BIENENSTOCK & VON DER MALSBURG, 1987; VON DER MALSBURG, 1988). In the pattern recognition applications three different patterns with artificial features were distinguished. KONEN & VON DER MALSBURG (1992, 1993) applied DLM to symmetry detection, and its ability to learn and generalize from single examples was demonstrated. Supplementary analytical considerations and a fast version of DLM can be found in (KONEN et al., 1994).

WANG et al. (1990) were probably the first who applied DLM to a recognition task on real world images, images of few faces as a gallery and a face with different expression as input. They also used running blobs instead of the stationary ones used in all models before. More recently a model for face recognition was developed by KONEN & VORBRÜGGEN (1993) .

3.2 Abstract Model

3.2.1 Task

DLM was proposed to serve as an elementary process to match and compare *labeled graphs* in a *neural system*. The nodes of the graphs are represented by model neurons, and each neuron has attached feature information. The *graph topography* is expressed by lateral connections between neurons of one graph. They are usually excitatory between neighbors and inhibitory for distant nodes. We assume two layers representing similar patterns. Figure 3.1, for example, shows two graphs representing two different images of the same face.

The task of DLM is to establish a connectivity between the two graphs that connects only *corresponding neurons*. Two neurons correspond to each other if they represent the same part of the object. This definition is obviously useless in a system that is supposed to find these correspondences only on the basis of image information. A more operational definition is the following: two neurons correspond to each other if they have a similar feature and if they have common neighbors (i.e. neighbors should be connected with neighbors). The second *constraint* of neighborhood preservation leads to continuous and regular mappings, but the mapping may still contain mirror reflection, translation, rotation, scaling, and distortion to a certain degree. From all these possible mappings the first constraint of feature similarity distinguishes one as the best, and that one has to be found.

In order to make all regular mappings possible, each neuron in one layer has to be potentially connected to each neuron in the other layer (*all-to-all connectivity*). And since preferably neurons with similar features should be connected, it is reasonable to *initialize* the *synaptic weights* of the links with the similarity between the features. This is indicated in Figure 3.1 by arrows of different line width. DLM has to rule out most of them, ending up with an approximate *one-to-one* mapping.

3.2.2 Principles

DLM is a dynamic process that can be modeled in many different ways. A system using a running blob dynamics will be presented in the following chapter. In this section I will try to illustrate four basic principles of DLM that are important for a model based on neural activities. In the formulation of chemical markers several terms would have to be replaced, but the principal ideas would be the same. In the formulation of *autonomous link dynamics* the following four principles would have to be replaced by two more abstract ones.

Correlation Encodes Neighborhood

Since topography plays a crucial role, the first principle of DLM is that the dynamics on one layer encodes neighborhood relations through *correlation* in the neural activities (see Figure 3.2). The correlation between neurons is high if they are adjacent and decreases with distance. Conversely, knowing the temporal signals of two neurons one can tell from the types of correlation whether they are neighbors or not. This can be achieved by many different dynamics generating clustered activities. In most models so far, stationary blobs

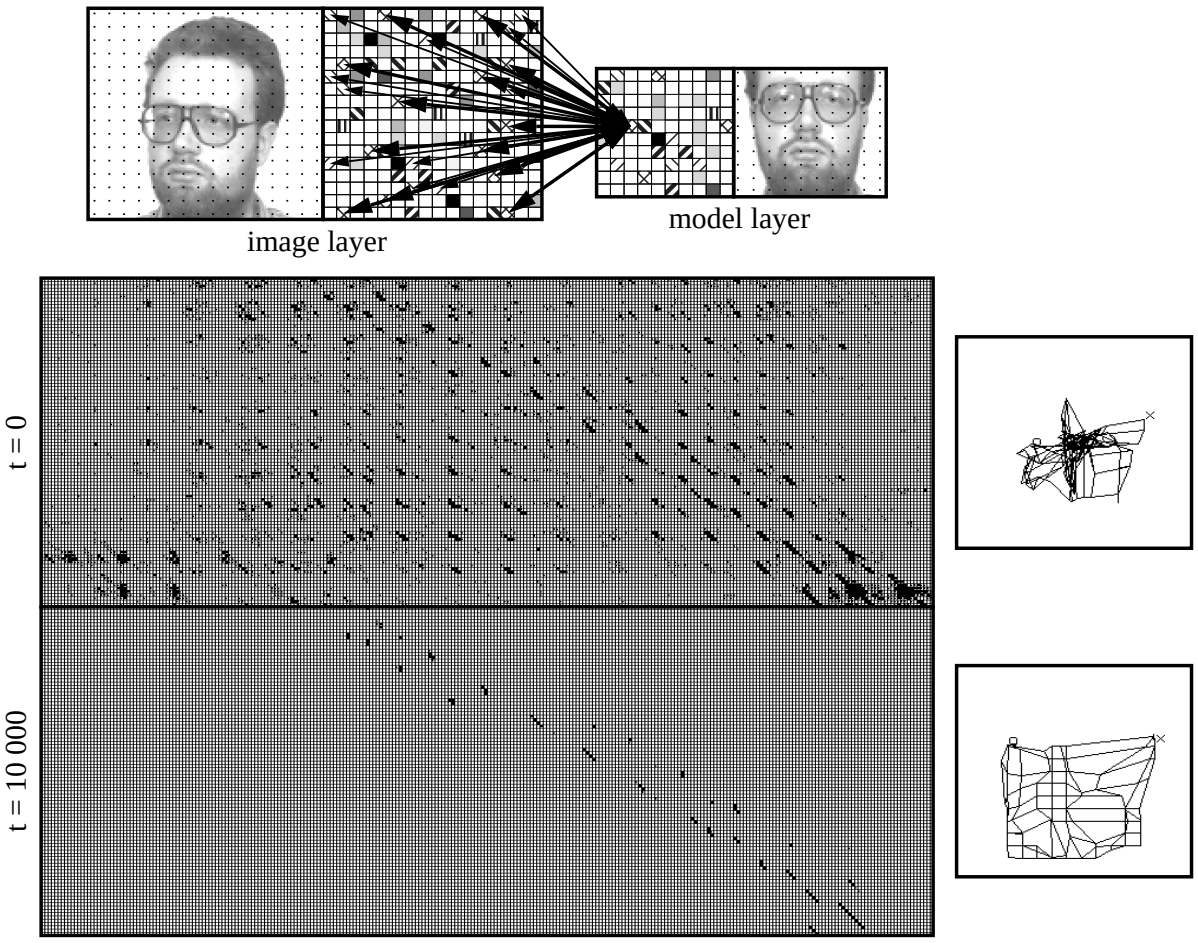

Figure 3.1: Initial and final connectivity for DLM. Image and model are represented by layers of 16×17 and 10×10 nodes, respectively, indicated by black dots. Each node is labeled with a local feature indicated by small texture patterns. Initially, the image layer and the model layer are connected all-to-all with synaptic weights depending on the feature similarities of the connected nodes, indicated by arrows of different line widths. The task of DLM is to select the correct links and establish a regular one-to-one mapping. We see here the initial connectivity at $t = 0$ and the final one at $t = 10\,000$. Since the *connectivity* between a model and the image is a four-dimensional matrix, it is difficult to visualize it in an intuitive way. If the rows of each layer are concatenated to a vector, top row first, the connectivity matrix becomes two-dimensional as shown at the left. The model index increases from left to right, the image index from top to bottom. High similarity values are indicated by black squares. A second way to illustrate the connectivity is the net display shown at the right. The image layer serves as a canvas on which the model layer is drawn as a net. Each node corresponds to a model neuron, neighboring neurons are connected by an edge. The location of the nodes indicates the center of gravity of the projective field of the model neurons, considering synaptic weights as physical mass. In order to favor strong links, the masses are taken to the power of three. (See Figure 3.6 for connectivity development in time.)

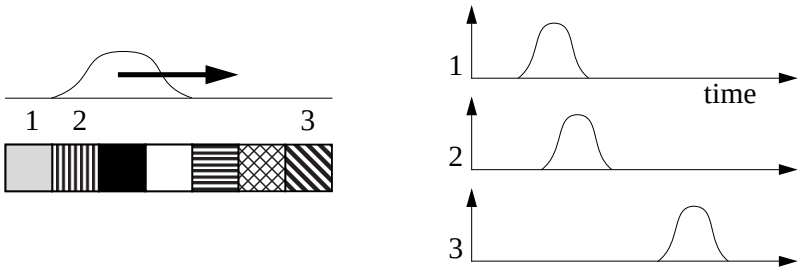

Figure 3.2: First principle of DLM. Neighborhood relations in each layer are encoded by signal correlation. As an example the dynamics is shown as a running blob of activity. From the temporal signals shown on the right one can tell that neurons 1 and 2 are neighbors while neurons 2 and 3 are not.

---

of activity have been used. In Section 4.2.3 a dynamics with running blobs is presented. Waves of different orientation are being investigated in our institute as well (SCHWARZ, 1995), and one could also think of oscillatory modes of different frequency such as in a membrane or a layer of coupled chaotic oscillators.

Layer Dynamics Synchronize

The second principle of DLM is that of *layer synchronization* (see Figure 3.3). Assuming two layers of the same size are connected by a perfect one-to-one mapping, then the activity dynamics of both layers have to synchronize such that after a short while *corresponding neurons* of the two layers are well-*correlated*, and vica versa: you can infer from the temporal signals of neurons of two different layers whether they correspond to each other or not. This is typically achieved by cooperation through the mutual connections. As a side effect of the cooperation the blobs become larger and the layer activities stronger.

Synchrony is Robust Against Noise

The third principle of DLM is *robustness* against noise (see Figure 3.4). Usually the *initial connectivity* between two layers is not perfect but given only by the feature similarities of the neurons. Hence many *accidental links* are present and correct links may be missing if the patterns are not identical. Distortion may cause the correct mapping not to be one-to-one. If one pattern is partially occluded only the remaining part can be matched.

Robustness against noise is achieved by *cooperation* between *neighboring links*. Two links are neighboring if both the source nodes and the destination nodes are neighbors. Since each link tends to synchronize the connected nodes, and since neighboring nodes are synchronized through the layer dynamics, groups of neighboring links can cooperate and are more successful than accidental links, even if the latter are stronger. Groups of neighboring links emerge if whole patches of local features are similar, i.e. if at least subgraphs can be matched. The cooperation between neighboring links is illustrated in Figure 3.5.

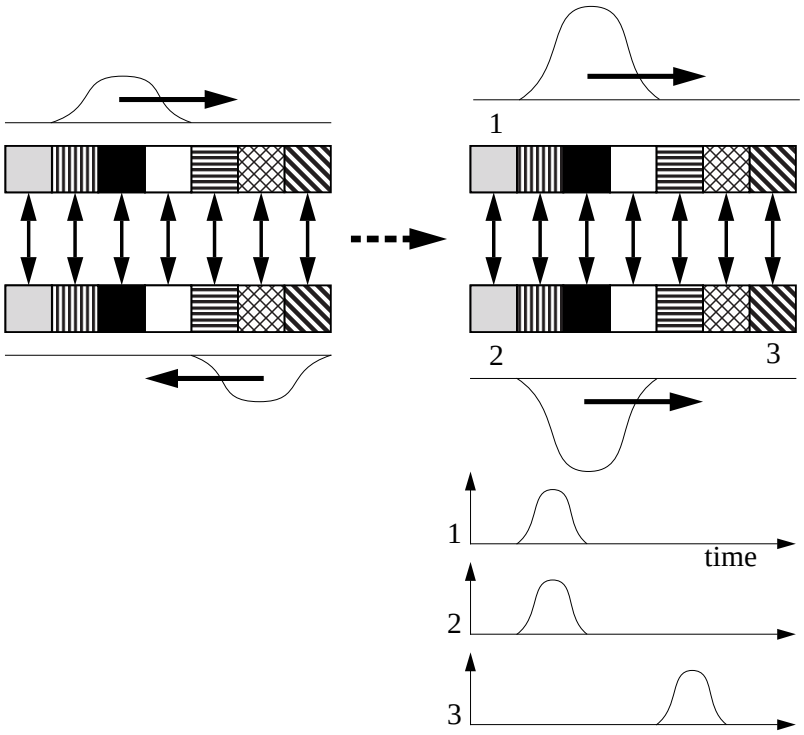

Figure 3.3: Second principle of DLM. Layer activity dynamics synchronize, and correlation of neurons of different layers encode correspondence. Initially the blobs move in different directions. Since the input into one layer is a copy of the activity of the other layer, the two blobs tend to synchronize and run aligned with each other from then on. The correspondence between neurons of different layers can then be read off their time signals as shown in the bottom graphs. As a side effect of the cooperation the blobs become larger.

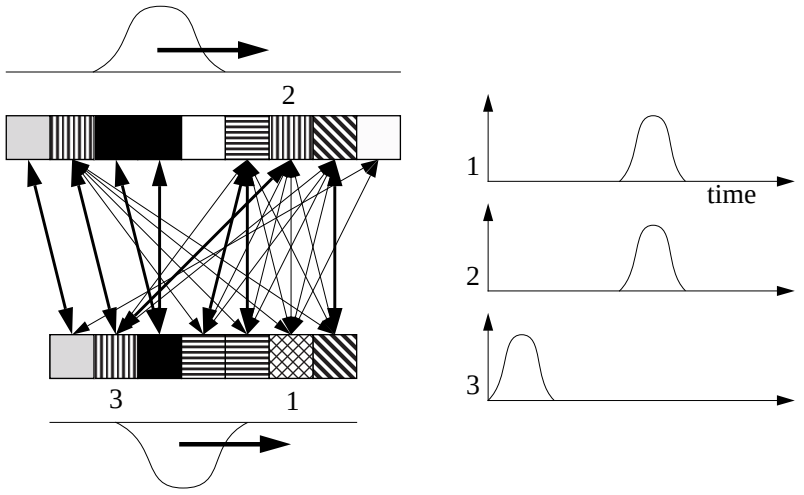

Figure 3.4: Third principle of DLM. Layer activity dynamics synchronize despite presence of noise, i.e. accidental links, missing links, distortion, and occlusion. Since the blobs cover a neighborhood of cells, links can cooperate if they have same neighbors in both layers. This reduces the influence of accidental links, which are usually isolated (see also Figure 3.5).

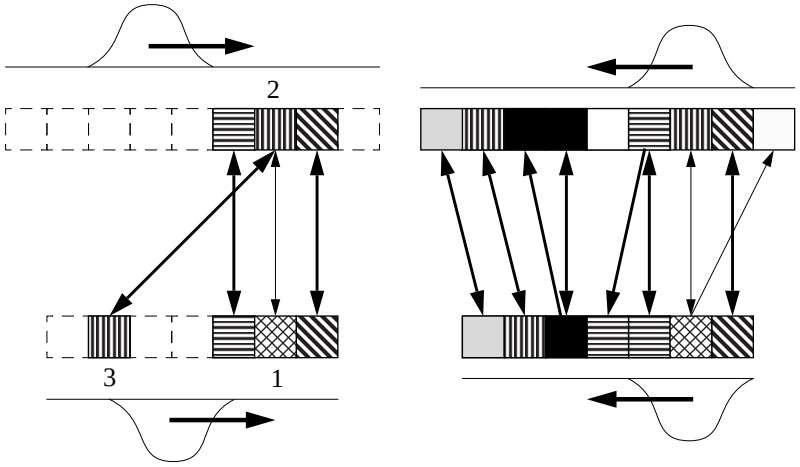

Figure 3.5: Fourth principle of DLM. The initial connectivity is refined on the basis of correlations between corresponding neurons. Since the layer dynamics synchronize despite noise in the initial connectivity, accidental links can be suppressed by other links connecting better correlated neurons. The final state is shown on the right. On the left the cooperation between links is illustrated. We see here two links converging onto neuron 2. One is weaker than the other but it can cooperate with the two directly neighboring links. That favors the weak one, which will eventually survive the link dynamics.

Synchrony Structures Connectivity

The first three principles induce correlations between neighboring neurons or those corresponding to each other if they belong to different layers. This synchrony is robust against noise and usually cleaner than the initial connectivity. As a fourth principle one can therefore apply *fast synaptic plasticity* to modify the links on the basis of induced synchrony between neural activities (see Figure 3.5). It typically consists of a Hebbian-like *growth rule*, but on a fast time scale, and a *normalization rule*. The growth rule favors links between synchronized neurons, while the normalization rule supresses links between less synchronized neurons. This principle works iteratively, i.e. synchrony improves with the development of a regular connectivity and the connectivity is refined by the establishing synchrony. Figure 3.6 shows connectivity and correlations developing in time. One can see that at a given time the correlations are cleaner than the connectivity and that both improve together.

3.3 Discussion

3.3.1 Critique

In Chapter 2 it was argued that labeled graphs are a promising basis for a uniform theory of perception. The problem with graphs is that they do not seem to fit into neural architecture. So far DLM is the only serious approach for processing graphs in neural nets, and we have seen its principles. Despite its potential, DLM is not able to account for the performance of the mature visual system for the two following reasons:

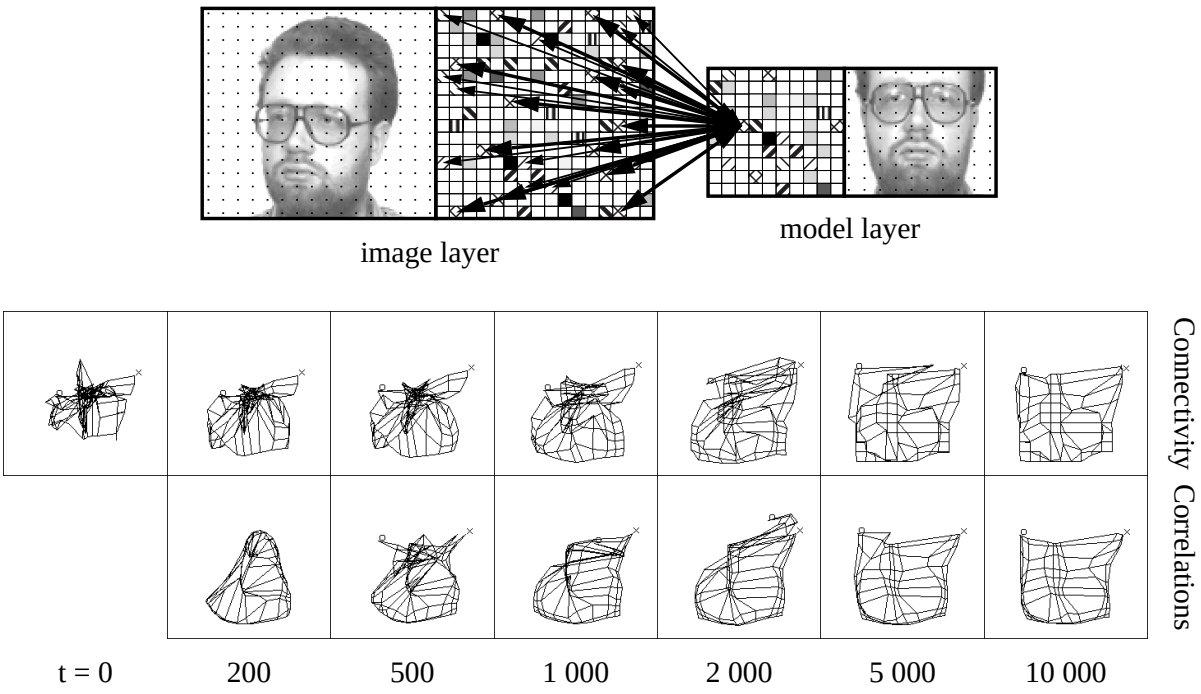

Figure 3.6: Connectivity and correlations developing in time. It can be seen how the correlations develop faster and are cleaner than the connectivity. Both are iteratively refined on the basis of the other. (Based on simulations as described in Chapter 4.)

DLM is Too Slow

So far DLM has been applied only to visual tasks. Psychophysical experiments by Subramaniam et al. (1995) show that our visual system is extremely flexible and quick. Subjects are able to recognize line drawings of objects from a sequence in which each object is shown only for 72 ms. Neurobiological experiments show that highly object-specific cells in the anterior superior temporal polysensory area (STPa) respond with a delay of only 70 ms to the stimulus (Oram & Perrett, 1994). If one assumes 10 ms for the interaction between two neurons, one would have only 7 iterations left for one recognition. That is definitely too few for DLM in the current formulation.

There are several ways to speed up DLM under certain conditions. For example if the outline of the object is given by reliable segmentation cues, planar waves could be induced running twice over the object in different directions. By this, the connectivity can in principle be induced very quickly. DLM with running waves has recently been investigated by Schwarz (1995) .

von der Malsburg (1994) suggested that DLM in its pure form is used only in the infant, and that later shortcuts are developed with experience to achieve the high performance of the adult. On difficult visual tasks the shortcuts fail, and then even the adult system has to revert to the slow process of DLM.

It is interesting to compare the speed of DLM with other neural models of perception. It is obvious that feed-forward nets such as back-propagation perform much faster on recognition tasks once they are trained. But in terms of *training effort* it has been shown that DLM is much faster (Bienenstock & Doursat, 1991; Konen & von der Malsburg, 1992, 1993). Due to the matching process, DLM can take full advantage of

single examples. While typical neural nets have to learn invariance or robustness against transformations such as translation, rotation, and distortion, DLM is already endowed with these abilities.

A performance comparison of DLM with KOHONEN'S algorithm has been done by BEHRMANN (1993). The Kohonen-algorithm, which has been derived from DLM, is widely used in technical applications, and one should assume that it is much more efficient than the original model. But it has turned out that both models are comparable in terms of speed. DLM has the additional advantage of being less sensitive to parameter variations.

DLM is Too Expensive in Terms of Connectivity

As seen above, DLM initially requires links from all neurons in one layer to all neurons in the other in order to obtain full invariance against translation, rotation, etc. It it clear that our visual system cannot afford *all-to-all connectivity* from V1 to all stored models in our memory. Somehow the huge number of required connections has to be reduced.

The only solution to this problem is to introduce *hierarchy*. This can be done in two ways. Firstly, a cascade of restricted mappings approximates the general one with much less connectivity. This principle was very well demonstrated in (ANDERSON & VAN ESSEN, 1993; VAN ESSEN et al., 1994; OLSHAUSEN, 1994).

Secondly, the mapping can start with coarse resolution and then be refined to a higher resolution level. RINNE (1995) has applied DLM in this way to grey value images. Instead of a full initial connectivity he used a sparse connectivity of superlinks, each superlink representing a subarray of normal links. After ruling out most of the superlinks by DLM, he replaced the remaining ones by the respective bunches of normal links and continued refining the mapping by DLM. He thus was able to match 128×128 layers of neurons with each other. A similar technique has been used by WÜRTZ (1995) for a hierachical DLM on a wavelet multiscale representation. These two hierarchical DLM models are still biologically implausible, since in them links of one type get directly replaced or initialized by links of another type.

3.3.2 Comparison with Other Models

Object recognition invariant against translation is a very difficult task for conventional neural systems with fixed connectivity. For that reason only few attempts have been made to build such systems.

The first class of systems is the *Neocognitron* of FUKUSHIMA et al. (1983) and related models such as the *weight sharing back-propagation networks* (LECUN et al., 1989), in one dimension also known as time delayed neural networks (TDNN). The Neocognitron is a feed-forward network which achieves translation invariance by spatial low-pass filtering and subsampling of neural responses, by which also discriminative information gets lost. Translation invariance is limited since the possible low-pass filtering and subsampling depends on the complexity of the features. The weight sharing back-propagation applies only subsampling and achieves very little translation invariance. The *training effort* is very high for these architectures and the Neocognitron in addition requires a very careful design of the network layout highly adapted to the training patterns. Compared to that, DLM achieves full translation invariance without loss of discriminative capabilities. It

has a simple and robust architecture and is able to learn and generalize from single examples. The main advantages of the feed-forward architectures is that they are much faster in recognition and that they show a feature hierarchy, which is not yet present in the DLM models.

A very different approach are the *neural routing circuits* (Anderson & van Essen, 1993; van Essen et al., 1994; Olshausen, 1994). The authors have implemented a cascade of restricted mappings between neural layers; those can be controlled and realize a large range of mappings. This system is restricted to translation and scaling. Rotation as well as distortion could easily be implemented, but if the system gets too many degrees of freedom the control might become too expensive. The neural routing circuits perform a normalization of an input pattern and recognition has to be achieved by an additional module such as an associative memory. Their advantage is that they allow the system to control the mappings and that the degrees of freedom are reasonably reduced. Thus the neural routing circuits can establish the appropriate mapping between image and model much faster than DLM. Problems may arise with general distortions, which cannot easily be accounted for by neural routing circuits, since it would require too much control structure.

3.3.3 Future Perspectives

DLM need not be restricted to visual pattern recognition. The potential of labeled graphs goes much farther, and the principles of DLM are more generally applicable. One might therefore think about representing more *abstract patterns* such as trajectories, language, or even ideas by means of labeled graphs and about matching them with DLM. The next step would then be to apply DLM to graphs with a more general topography than just the two-dimensional one of the visual field. Graphs of high dimension and unusual topography should be treatable by DLM as well.

Another direction of research would be of a more theoretical nature. So far DLM-models have been built mainly heuristically. Many design decisions are motivated by practical experience rather by a solid theoretical basis. I think it would be very helpful to consider DLM on the more abstract level of *autonomous link dynamics*. The work of Häussler & von der Malsburg (1983) and Wagner & von der Malsburg (1995) goes in this direction, but is restricted to the retinotopy problem.

Finally it would be very interesting to integrate DLM with the other models discussed above. Neocognitron and related models, neural routing circuits, and DLM all have their advantages and drawbacks, and it might be possible to combine their capabilities.

Chapter 4

Face Recognition by Dynamic Link Matching

Abstract: A complete system for face recognition based on Dynamic Link Matching (DLM) is presented. Faces are represented by layers of neurons with jets attached as local features. A gallery of model layers is connected to one image layer, which is of larger size. The connectivity between models and image is initialized according to the jet similarities. The layer dynamics generates blobs of activity continuously moving over the layers. The blobs interact via the connectivity matrices and align. Based on that, fast synaptic plasticity develops a one-to-one mapping between the layers. The model layers have a total activity dependent on their similarity to the image. A winner-take-all mechanism sequentially rules out the less similar models, letting the most similar model survive. Since the image layer is larger than the model layers, an attention window is introduced in the form of a large blob restricting the space available for the small running blob. Due to interactions with the running blob, the attention blob automatically aligns with the correct face position. Recognition results on galleries of up to 111 faces are presented.

4.1 Introduction

In the previous chapter I described *Dynamic Link Matching* as a neural process for matching labeled graphs. DLM systems have recently been developed by Konen & von der Malsburg (1992, 1993) and Konen & Vorbrüggen (1993). They all use the following type of dynamics: Noise, local excitation and global inhibition on the image layer induce the development of one stationary blob of activity. Links transfer this activity from the image to the model layer, and the same dynamics generates a second stationary blob there. If the patterns represented on the image and the model layer are similar, the blob on the model layer is likely to appear in a corresponding location. After both blobs have developed, the weights of the links between the layers are modified based on the co-activity of image and model neurons. Then all neurons are reset to zero activity, and the process starts from the beginning. The blob dynamics can be replaced by algorithmically setting a blob at a random position on the image layer and at the position in the model layer with maximal input from the image layer (Konen et al., 1994).

This stationary blob dynamics for DLM has two conceptual drawbacks: Firstly, the

whole process has to be controlled in a fairly artificial way to realize the sequence of blobs and weight adaptation steps. One would rather have an autonomous dynamics that needs no accurate schedule for layer dynamics, weight adaptation, and resetting. Secondly, the information about correspondence that was obtained with the pair of blobs is almost completely lost for the next pair of blobs. It is only stored in the weight matrix as a small modification. *Topography* is only conveyed by the overlap of the blobs, which seems to be inherently a slow process. The first intention of the work presented here was therefore to replace the stationary blob dynamics by a continuous and autonomous dynamics to overcome the two above-mentioned drawbacks. The solution that I chose was to introduce delayed self-inhibition that makes the activity blob run. It moves continuously over the whole layer. The blob positions at one particular moment are used for determining the blob positions at the next moment, i.e. the information about correspondences is preserved in the layer dynamics. In addition, topography is conveyed by the continuous motion of the blobs, which is potentially faster than the old method. Weight adaptation can take place on-line, and no sophisticated control schedule is required. Running blobs for DLM have previously been used by WANG et al. (1990), but they generated them with asymmetrical convolution kernels, which has conceptual drawbacks: For example, since the speed and direction of the blobs is fixed, layers must have wrap-around conditions and thus have to be of equal size.

It has always been claimed that DLM is a model for object recognition. Although it serves as the neural conceptual basis of an already successful technical face recognition system (LADES et al., 1993; WISKOTT et al., 1995; Chapter 5), the DLM recognition systems are very limited so far. In most applications only few models, about three, were discriminated. The data were often artificial, and the detection of the correct model was done in a biologically implausible way, e.g. by considering the sum over the weights, a measure that is not accessible in a real system. It was therefore my second intention to build a DLM system that is actually a complete recognition system, solving the same task as the technical system for face recognition, although much slower. A gallery of 111 faces is stored and new faces on images larger than the models have to be recognized. This requires finding the face in the image, matching it to the model gallery, and recognizing the correct one among the 111 competing alternatives.

4.2 The System

4.2.1 Architecture and Dynamics — Overview

Figure 4.1 shows the general architecture of the system. *Faces* are represented as rectangular *graphs* by layers of neurons. Each neuron represents a node and has a *jet* attached. A jet is a local description the grey-value distribution (see Appendix A). Topographical relationships between nodes are encoded by excitatory and inhibitory lateral connections. The *model graphs* are scaled horizontally and vertically and aligned manually, such that certain nodes of the graphs are placed on the eyes and the mouth (cf. Section 4.3.1). Model graphs (10×10 nodes) are smaller than the *image graph* (16×17 nodes). Since the face in the image may be arbitrarily translated, the *connectivity* between *model* and *image domain* has to be *all-to-all* initially. The connectivity matrices are initialized using the *similarities* between the *jets* of the connected nodes. DLM serves as a process

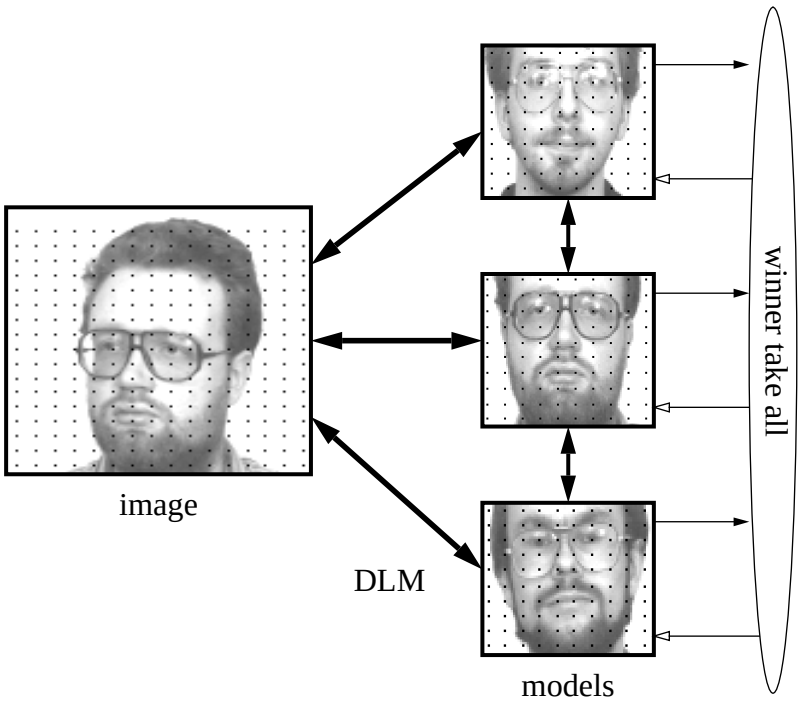

Figure 4.1: Architecture of the DLM face recognition system. Several models are stored as neural layers of local features on a 10×10 grid, as indicated by the black dots. A new image is represented by a 16×17 layer of nodes. Initially, the image is connected all-to-all with the models. The task of DLM is to find the correct mapping between the image and the models, thus providing translation invariance and robustness against distortion. Once the correct mapping is found, a simple winner-take-all mechanism can detect the model that is most active and most similar to the image.

---

to restructure the connectivity matrices and to find the correct *(one-to-one) mapping* between the models and the image (see Figure 3.1). The models cooperate with the image depending on their similarity. A simple winner-take-all mechanism sequentially rules out the least active and least similar models, and the best-fitting one eventually survives.

The dynamics on each layer (graph) of neurons (nodes) is such that it produces a *running blob* of activity which moves continuously over the whole layer. An activity blob can easily be generated from noise by *local excitation* and *global inhibition*. It is caused to move by *delayed self-inhibition*, which also serves as a memory for the locations where the blob has recently been. Since the models are aligned with each other, it is reasonable to enforce *alignment* between their running blobs by excitatory connections between neurons representing the same facial location. The blobs on the *image* and the *model layers* cooperate through the connection matrices; they tend to align and induce correlations between corresponding neurons. Then fast synaptic plasticity and a normalization rule coherently modify the synaptic weights, and the correct connectivities between models and image layer can develop. Since the models get different input from the image, they differ in their total activity. The model with strongest connections from the image is the most active one. The models compete on the basis of their total activity. After a while the winner-take-all mechanism suppresses the least competitive models,

Layer dynamics:

$$
\begin{aligned}
h_i^p(t_0) &= 0 \\
\dot{h}_i^p(t) &= -h_i^p + \sum_{i'} \max_{p'} \left(g_{i-i'} \sigma(h_{i'}^{p'}) \right) - \beta_h \sum_{i'} \sigma(h_{i'}^p) - \kappa_{hs} s_i^p \qquad (4.1) \\
&\quad + \kappa_{hh} \max_{qj} \left(W_{ij}^{pq} \sigma(h_j^q) \right) + \kappa_{ha} \left(\sigma(a_i^p) - \beta_{ac} \right) - \beta_\theta \Theta(r_\theta - r^p) \\
s_i^p(t_0) &= 0 \\
\dot{s}_i^p(t) &= \lambda_\pm (h_i^p - s_i^p) \qquad (4.2) \\
g_{i-i'} &= \exp\left(-\frac{(i-i')^2}{2\sigma_g^2} \right) \qquad (4.3) \\
\sigma(h) &= \begin{cases} 0 & : \quad h \le 0 \\ \sqrt{h/\rho} & : \quad 0 < h < \rho \\ 1 & : \quad h \ge \rho \end{cases} \qquad (4.4)
\end{aligned}
$$

Attention dynamics:

$$
\begin{aligned}
a_i^p(t_0) &= \alpha_{\mathcal{N}} \mathcal{N}(\mathcal{J}_i^p) \\
\dot{a}_i^p(t) &= \lambda_a \left(-a_i^p + \sum_{i'} g_{i-i'} \sigma(a_{i'}^p) - \beta_a \sum_{i'} \sigma(a_{i'}^p) + \kappa_{ah} \sigma(h_i^p) \right) \qquad (4.5)
\end{aligned}
$$

Link dynamics:

$$
\begin{aligned}
W_{ij}^{pq}(t_0) &= \mathcal{S}_{ij}^{pq} = \max\left(\mathcal{S}_\phi(\mathcal{J}_i^p, \mathcal{J}_j^q), \alpha_{\mathcal{S}} \right) \\
\dot{W}_{ij}^{pq}(t) &= \lambda_W \left(\sigma(h_i^p)\sigma(h_j^q) - \Theta\left(\max_{j'} (W_{ij'}^{pq} / \mathcal{S}_{ij'}^{pq}) - 1 \right) \right) W_{ij}^{pq} \qquad (4.6)
\end{aligned}
$$

Recognition dynamics:

$$
\begin{aligned}
r^p(t_0) &= 1 \\
\dot{r}^p(t) &= \lambda_r r^p \left(F^p - \max_{p'} (r^{p'} F^{p'}) \right) \qquad (4.7) \\
F^p(t) &= \sum_i \sigma(h_i^p)
\end{aligned}
$$

Table 4.1: Formulas of the DLM face recognition system

and eventually only the best one survives. Since the image layer may be significantly larger than the model layers, I introduce an attention window in form of a large blob. It interacts with the running blob, restricts its region of motion, and can be shifted by it to the actual face position.

The equations of the system are given in Table 4.1; the respective symbols are listed in Table 4.2. In the following sections I will explain the system step by step: blob formation, blob mobilization, interaction between two layers, link dynamics, attention dynamics, and recognition dynamics. (In order to make the description clearer, parts of the equations in Table 4.1 corresponding to these functions will be repeated.)

Variables:

h	internal state of the layer neurons
s	delayed self-inhibition
a	attention
W	synaptic weights between neurons of two layers
r	recognition variable
F	fitness, i.e. total activity of each layer

Indices:

$(p;p';q;q')$	layer indices, 0 indicates image layer, $1,...,M$ indicate model layers
$=(0;0;1,...,M;1,...,M)$	if formulas describe image layer dynamics
$=(1,...,M;1,...,M;0;0)$	if formulas describe model layers dynamics
$(i;i';j;j')$	two-dimensional indices for the individual neurons in layers $(p;p';q;q')$ respectively

Functions:

$g_{i-i'}$	Gaussian interaction kernel
$\sigma(h)$	nonlinear squashing function
$\Theta(\cdot)$	Heavyside function
$\mathcal{N}(\mathcal{J})$	saliency of feature jet $\mathcal{J}$ (see Equation A.5)
$\mathcal{S}_\phi(\mathcal{J},\mathcal{J}')$	similarity between feature jets $\mathcal{J}$ and $\mathcal{J}'$ (see Equation A.7)

Parameters:

β_h	$=$	0.2	strength of global inhibition
β_a	$=$	0.02	strength of global inhibition for attention blob
β_{ac}	$=$	1	strength of global inhibition compensating for the attention blob
β_θ	$=$	∞	global inhibition for model suppression
κ_{hs}	$=$	1	strength of self-inhibition
κ_{hh}	$=$	1.2	strength of interaction between image and model layers
κ_{ha}	$=$	0.7	effect of the attention blob on the running blob
κ_{ah}	$=$	3	effect of the running blob on the attention blob
$\lambda_\pm$			decay constant for delayed self-inhibition
$=\lambda_+$	$=$	0.2	if $h-s>0$
$=\lambda_-$	$=$	0.004	if $h-s\leq 0$
λ_a	$=$	0.3	time constant for the attention dynamics
λ_W	$=$	0.05	time constant for the link dynamics
λ_r	$=$	0.02	time constant for the recognition dynamics
$\alpha_\mathcal{N}$	$=$	0.001	parameter for attention blob initialization
$\alpha_\mathcal{S}$	$=$	0.1	minimal weight
ρ	$=$	2	slope radius of squashing function
σ_g	$=$	1	Gauss width of excitatory interaction kernel
r_θ	$=$	0.5	threshold for model suppression

Table 4.2: Variables and parameters of the DLM face recognition system

4.2.2 Blob Formation

Blob formation on a layer of neurons can easily be achieved by *local excitation* and *global inhibition*. Local excitation generates clusters of activity, and global inhibition lets the clusters compete against each other. The strongest one will finally suppress all others and grow to an equilibrium size determined by the strengths of excitation and inhibition. The corresponding equations are (cf. Equations 4.1, 4.3, and 4.4):

$$\dot{h}_i(t) = -h_i + \sum_{i'} (g_{i-i'}\sigma(h_{i'})) - \beta_h \sum_{i'} \sigma(h_{i'}), \tag{4.8}$$

$$g_{i-i'} = \exp\left(-\frac{(i-i')^2}{2\sigma_g^2}\right), \tag{4.9}$$

$$\sigma(h) = \begin{cases} 0 & : \quad h \leq 0 \\ \sqrt{h/\rho} & : \quad 0 < h < \rho \\ 1 & : \quad h \geq \rho \end{cases} . \tag{4.10}$$

The internal state of the neurons is denoted by h_i, where i is a two-dimensional Cartesian coordinate for the location of the neuron. The neurons are arranged on a regular square lattice with spacing 1, i.e. $i = (0,0), (0,1), (0,2), ..., (1,0), (1,1), ...$ The neural activity (which can be interpreted as a mean firing rate) is determined by the squashing function $\sigma(h)$ of the neuron's internal state h. The neurons are connected excitatorily through the Gaussian interaction kernel g. The strength of global inhibition is controlled by β_h. It is obvious that a blob can only arise if $\beta_h < g_0 = 1$ (imagine only one neuron is active), and that the blob is larger for smaller β_h. Infinite growth of h is prevented by the decay term $-h$, because it is linear, while the blob formation terms saturate due to the squashing function $\sigma(h)$. The special shape of $\sigma(h)$ is motivated by three factors. Firstly, σ vanishes for negative values to suppress oscillations in the simulations by preventing undershooting. Secondly, the high slope for small arguments stabilizes small blobs and makes blob formation from low noise easier, because for small values of h the interaction terms dominate over the decay term. Thirdly, the finite slope region between low and high argument values allows the system to distinguish between the inner and outer parts of the blobs by making neurons in the center of a blob more active than at its periphery. Additional multiplicative parameters of the decay or excitation terms would only change time and activity scale, respectively, and do not generate qualitatively new behavior. In this sense the parameter set is complete and minimal. A detailed discussion of this dynamics has been given by Amari (1977), also in the context of self-organizing topographic mappings (Amari, 1980, 1989).

4.2.3 Blob Mobilization

Generating a running blob can be achieved by delayed self-inhibition, which drives the blob away from its current to a neighboring location, where the blob generates new self-inhibition. This mechanism produces a continuously moving blob (see Figure 4.2). The driving force and the recollection time as to where the blob has been can be independently controlled by their respective time constants. The corresponding equations are (cf. Equations 4.1 and 4.2):

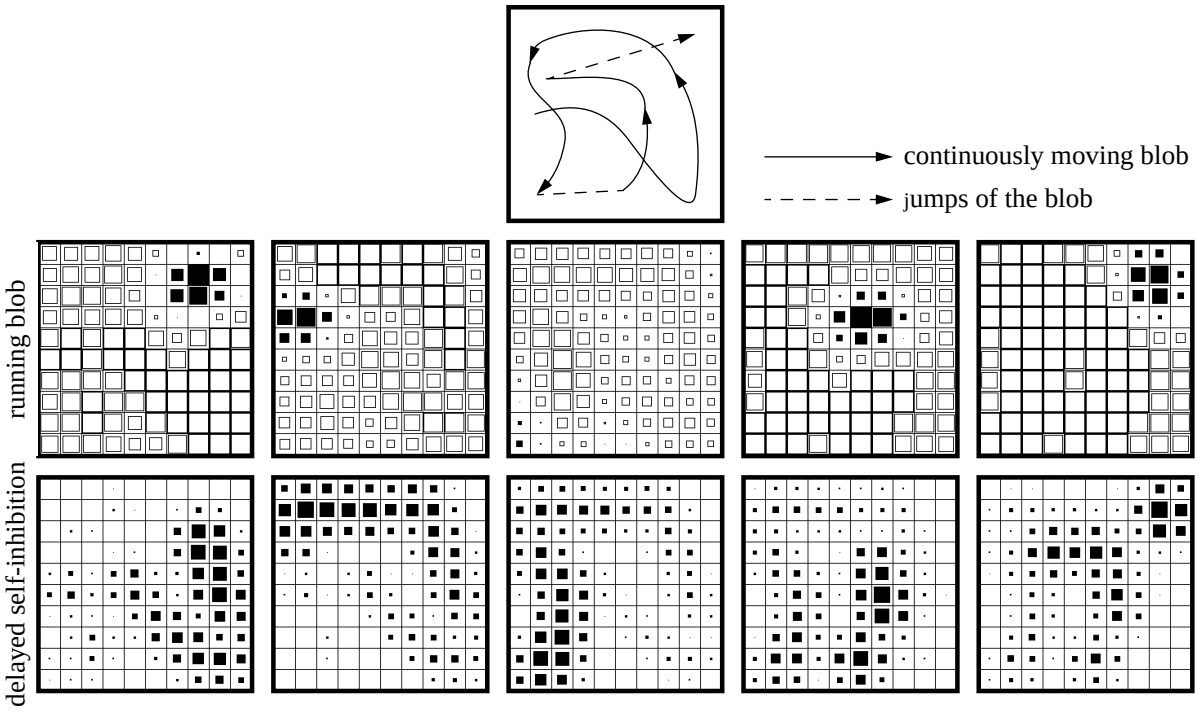

Figure 4.2: A sequence of layer states as simulated with Equations 4.11 and 4.12. The activity blob h shown in the middle row has a size of approximately six active nodes and moves continuously over the whole layer. Its course is shown in the upper diagram. The delayed self-inhibition s, shown in the bottom row, follows the running blob and drives it forward. One can see the self-inhibitory tail that repels the blob from regions just visited. Sometimes the blob runs into a trap (cf. column three) and has no way to escape from the self-inhibition. It then disappears and reappears again somewhere else on the layer. (The temporal increment between two successive frames is 20 time units.)

$$\dot{h}_i(t) = -h_i + \sum_{i'} (g_{i-i'}\sigma(h_{i'})) - \beta_h \sum_{i'} \sigma(h_{i'}) - \kappa_{hs} s_i, \tag{4.11}$$

$$\dot{s}_i(t) = \lambda_{\pm}(h_i - s_i). \tag{4.12}$$

The self-inhibition s is realized by a leaky integrator with decay constant $\lambda_{\pm}$. The decay constant has two different values depending on whether $h-s$ is positive or negative. This accounts for the two different functions of the self-inhibition. The first function is to drive the blob forward. In this case $h > s$ and a high decay constant λ_+ is appropriate. The second function is to indicate where the blob has recently been, i.e. to serve as a memory and to repel the blob from regions recently visited. In this case $h < s$ and a low decay constant λ_- is appropriate. For small layers, λ_- should be larger than for large ones, because the blob visits each location more frequently. The speed of the blob is controlled by λ_+ and the coupling parameter κ_{hs}. They may also change the shape of the blob. Small values such as those used in the simulations presented here allow the blob to keep its equilibrium shape and drive it slowly; large values produce a fast-moving blob distorted to a kidney-shape.

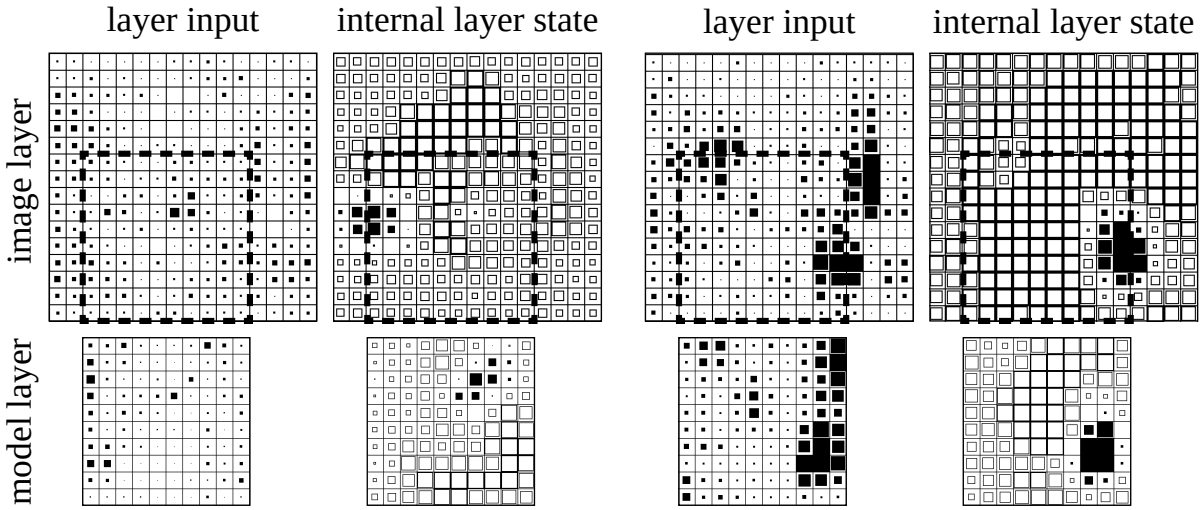

Figure 4.3: Synchronization between two running blobs as simulated with Equations 4.13 and 4.14. Layer input as well as the internal layer state h is shown at an early stage, in which the blobs of two layers are not yet aligned, left, and at a later state, right, when they are aligned. The two layers are of different size, and the region in layer 1 that correctly maps to layer 2 is indicated by a square defined by the dashed line. In the early non-aligned case one can see that the blobs are smaller and not at the location of maximal input. The locations of maximal input indicate where the actual corresponding neurons of the blob of the other layer are. In the aligned case the blobs are larger and at the locations of high layer input.

4.2.4 Layer Interaction and Synchronization

In the same way that the running blob is repelled by its self-inhibitory tail, it can also be attracted by excitatory input from another layer as conveyed by a connection matrix. Imagine two layers of the same size mutually connected by the identity matrix, i.e. each neuron in one layer is connected only with the one corresponding neuron in the other layer having the same index value. The input then is a copy of the blob of the other layer. This favors alignment between the blobs, because then they can cooperate and stabilize each other. This synchronization principle holds also in the presence of the noisy connection matrices generated by real image data (see Figure 4.3). The corresponding equation is (cf. Equation 4.1):

$$\begin{aligned} \dot{h}_i^p(t) &= -h_i^p + \sum_{i'} \left(g_{i-i'}\sigma(h_{i'}^p)\right) - \beta_h \sum_{i'} \sigma(h_{i'}^p) - \kappa_{hs} s_i^p \\ &\quad + \kappa_{hh} \max_j \left(W_{ij}^{pq}\sigma(h_j^q)\right), \qquad (4.13) \\ \dot{s}_i^p(t) &= \lambda_{\pm}(h_i^p - s_i^p). \qquad (4.14) \end{aligned}$$

The two layers are indicated by the indices p and q. The synaptic weights of the connections are W, and the strength of mutual interaction is controlled by the parameter κ_{hh}. (The reason why I use the maximum function instead of the usual sum will be discussed in Section 4.2.10.)

4.2.5 Link Dynamics

In Section 3.2.2 it was demonstrated that the links between two layers can be cleaned up and structured by *fast synaptic plasticity* on the basis of correlations between pairs of neurons (see Figure 3.6). The correlations result from the layer synchronization described in the previous section. The link dynamics typically consists of a *growth rule* and a *normalization term*. The former lets the weights grow according to the correlation between the connected neurons. The latter prevents the links from growing indefinitely and induces competition such that only one link per neuron survives which suppresses all others. The corresponding equations are (cf. Equations 4.6):

$$\begin{aligned} W_{ij}^{pq}(t_0) &= \mathcal{S}_{ij}^{pq} = \max\left(\mathcal{S}_\phi(\mathcal{J}_i^p, \mathcal{J}_j^q), \alpha_\mathcal{S}\right), \\ \dot{W}_{ij}^{pq}(t) &= \lambda_W \left(\sigma(h_i^p)\sigma(h_j^q) - \Theta\left(\max_{j'}(W_{ij'}^{pq}/\mathcal{S}_{ij'}^{pq}) - 1\right)\right) W_{ij}^{pq}. \end{aligned} \tag{4.15}$$

Links are initialized by the similarity $\mathcal{S}_\phi$ between the jets $\mathcal{J}$ of connected nodes (see Equation A.7). The parameter $\alpha_\mathcal{S}$ guarantees a minimal positive synaptic weight, permitting each link to suppress others, even if the similarity between the connected neurons is small. This can be useful to obtain a continuous mapping if a link has a neighborhood of strong links inducing high correlations between the pre- and postsynaptic neurons of the weak link. The synaptic weights grow exponentially, controlled by the correlation between connected neurons defined as the product of their activities $\sigma(h_i^p)\sigma(h_j^q)$. The learning rate is additionally controlled by λ_W. Due to the Heavyside-function Θ, normalization takes place only if links grow beyond their initial value. Then the link dynamics is dominated by the normalization term, with a common negative contribution for all links converging to the same neuron. Notice that the growth term, based on the correlation, is different for different links. Thus the link with the highest average correlation will eventually suppress all others converging to the same neuron. Since the similarities $\mathcal{S}_\phi$ cannot be larger than 1, the synaptic weights W are restricted to the interval $[0, ..., 1]$.

4.2.6 Attention Dynamics

The alignment between the running blobs depends very much on the constraints, i.e. on the size and format of the layer on which they are running. This causes a problem, since the image and the models have different sizes. I have therefore introduced an *attention blob* which restricts the movement of the running blob on the image layer to a region of about the same size as that of the model layers. Each of the model layers likewise has an attention blob to keep the conditions for their running blobs similar to that in the image layer; this is important for alignment. The attention blob restricts the region for the running blob, but it can be shifted by the latter into a region where input is especially large and favors activity. The attention blob therefore automatically aligns with the actual face position (see Figures 4.4 and 4.5). The attention blob layer is initialized by a primitive segmentation cue, in this case the saliency of the respective jets (see Equation A.5), since the norm indicates the presence of textures of high contrast. The corresponding equations are (cf. Equations 4.1 and 4.5):

$$
\begin{aligned}
\dot{h}_i^p(t) &= -h_i^p + \sum_{i'} \left(g_{i-i'}\sigma(h_{i'}^p)\right) - \beta_h \sum_{i'} \sigma(h_{i'}^p) - \kappa_{hs} s_i^p \\
&\quad + \kappa_{hh} \max_j \left(W_{ij}^{pq}\sigma(h_j^q)\right) + \kappa_{ha}\left(\sigma(a_i^p) - \beta_{ac}\right), \qquad (4.16)\\
\dot{s}_i^p(t) &= \lambda_{\pm}(h_i^p - s_i^p), \qquad (4.17)
\end{aligned}
$$

$$
\begin{aligned}
a_i^p(t_0) &= \alpha_{\mathcal{N}} \mathcal{N}(\mathcal{J}_i^p), \\
\dot{a}_i^p(t) &= \lambda_a \left(-a_i^p + \sum_{i'} g_{i-i'}\sigma(a_{i'}^p) - \beta_a \sum_{i'} \sigma(a_{i'}^p) + \kappa_{ah}\sigma(h_i^p) \right). \qquad (4.18)
\end{aligned}
$$

The equations show that the attention blob a is generated by the same dynamics as was discussed in Section 4.2.2 for the formation of the running blob, without delayed self-inhibition, though since the attention blob is to be larger than the running blob, β_a has to be smaller than β_h. The attention blob restricts the region for the running blob via the term $\kappa_{ha}\left(\sigma(a_i^p) - \beta_{ac}\right)$, which is an excitatory blob $\sigma(a_i^p)$ compensating the constant inhibition β_{ac}. The attention blob on the other hand gets excitatory input $\kappa_{ah}\sigma(h_i^p)$ from the running blob. By this means the running blob can slowly shift the attention blob into its favored region. The dynamics of the attention blob has to be slower than that of the running blob; this is controlled by a value $\lambda_a < 1$. $\mathcal{N}$ is the saliency of the jets, and $\alpha_{\mathcal{N}}$ determines the initialization strength.

4.2.7 Recognition Dynamics

Each model cooperates with the image depending on its similarity. The most similar model cooperates most successfully and is the most active one. Hence the total activity of the model layers indicates which is the correct one. I have derived a winner-take-all mechanism from Eigen's (1978) evolution equation and applied it to detect the best model and suppress all others. The corresponding equations are (cf. Equations 4.1 and 4.7):

$$
\begin{aligned}
\dot{h}_i^p(t) &= -h_i^p + \sum_{i'} \left(g_{i-i'}\sigma(h_{i'}^p)\right) - \beta_h \sum_{i'} \sigma(h_{i'}^p) - \kappa_{hs} s_i^p \qquad (4.19)\\
&\quad + \kappa_{hh} \max_j \left(W_{ij}^{pq}\sigma(h_j^q)\right) + \kappa_{ha}\left(\sigma(a_i^p) - \beta_{ac}\right) - \beta_\theta \Theta(r_\theta - r^p), \\
\dot{s}_i^p(t) &= \lambda_{\pm}(h_i^p - s_i^p), \qquad (4.20)
\end{aligned}
$$

$$
\begin{aligned}
r^p(t_0) &= 1, \\
\dot{r}^p(t) &= \lambda_r r^p \left(F^p - \max_{p'}(r^{p'} F^{p'}) \right), \qquad (4.21)\\
F^p(t) &= \sum_i \sigma(h_i^p).
\end{aligned}
$$

The total layer activity is considered as a *fitness* F^p, different for each model p. The modified evolution equation can be easily analyzed if the F^p are assumed to be constant in time and the recognition variables r^p are initialized to 1. For the model

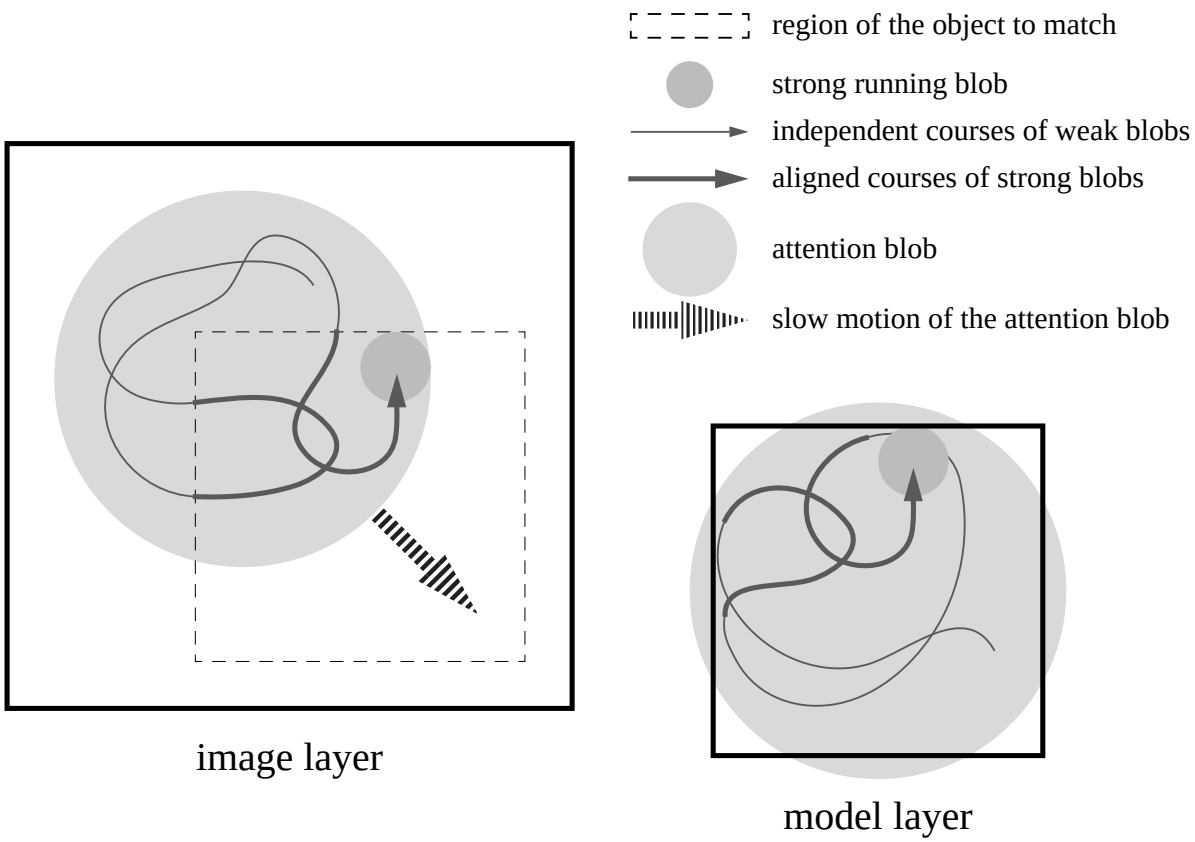

Figure 4.4: Schematic of the attention blob's function. The attention blob restricts the region in which the running blob can move. The attention blob, on the other hand, receives input from the running blob. That input will be strong in regions where the blobs in both layers cooperate and weak where they do not (see Figure 4.3). Due to this interaction the attention blob slowly moves to the correct region indicated by the square of dashed lines. The attention blob in the model layer is required to keep the conditions for the running blobs symmetrical.

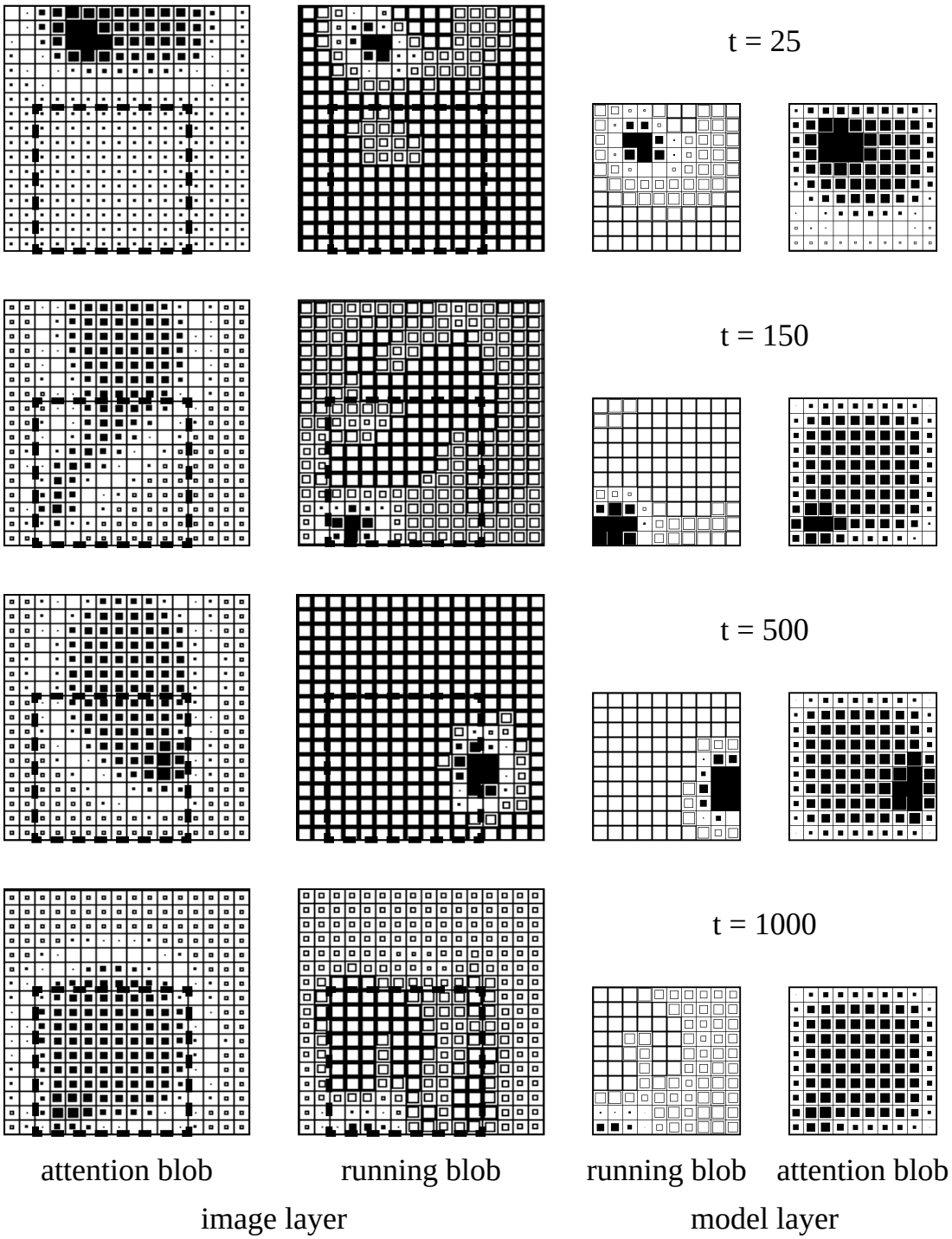

Figure 4.5: Function of the attention blob, using an extreme example of an initial attention blob manually misplaced for demonstration. At $t = 150$ the two running blobs ran synchronously for a while, and the attention blob has a long tail. The blobs then lost alignment again. From $t = 500$ on, the running blobs remained synchronous, and eventually the attention blob aligned with the correct face position, indicated by a square of dashed lines. The attention blob moves slowly compared to the small running blob, as it is not driven by self-inhibition. Without an attention blob the two running blobs may synchronize sooner, but their alignment will never become stable.

layer p_b with the highest fitness, the equation simplifies to $\dot{r}^{p_b}(t) = \lambda_r r^{p_b}(1 - r^{p_b})F^{p_b}$ with a stable fixed point at $r^{p_b} = 1$. For all other models the equation then simplifies to $\dot{r}^p(t) = \lambda_r r^p(F^p - F^{p_b})$, which results in an exponential decay of the r^p for all $p \neq p_b$. When a recognition variable r^p drops below the suppression threshold r_θ, the activity on layer p is suppressed by the term $-\beta_\theta \Theta(r_\theta - r^p)$. The time scale of the recognition dynamics can be controlled by λ_r.

4.2.8 Bidirectional Connections

The connectivity between two layers is bidirectional and not unidirectional as in the previous system (Konen & Vorbrüggen 1993). This is necessary for two reasons: Firstly, by this means the running blobs of the two connected layers can more easily align. With unidirectional connections one blob would systematically run behind the other. Secondly, connections in both directions are necessary for a recognition system. The connections from model to image layer are necessary to allow the models to move the attention blob in the image into a region that fits the models well. The connections from the image to the model layers are necessary to provide a discrimination cue as to which model best fits the image. Otherwise each model would exhibit the same level of activity.

4.2.9 Blob Alignment in the Model Domain

Since faces have a common general structure, it is advantageous to align the blobs in the model domain to insure that they are always at the same position in the faces, either all at the left eye or all at the chin etc. This is achieved by connections between the layers and leads to the term $+\sum_{i'} \max_{p'} \left(g_{i-i'} \sigma(h_{i'}^{p'}) \right)$ instead of $+\sum_{i'} \left(g_{i-i'} \sigma(h_{i'}^{p}) \right)$ in Equation 4.1. If the model blobs were to run independently, the image layer would receive input from all face parts at the same time, and the blob there would have a hard time aligning with a model blob, and it would be very uncertain whether it would be the correct one. The cooperation between the models and the image would depend more on accidental alignment than on the similarity between the models and the image, and it would then be very likely that the wrong model is picked up as the recognition result. One alternative is to let the models inhibit each other such that only one model can have a blob at a time. The models then would share time to match onto the image, and the best-fitting one would get most of the time. This would probably be the appropriate setup if the models were very different and without a common structure, as it is for general objects. The disadvantage is that the system needs much more time to decide which model to accept, because the relative layer activities in the beginning depend much more on chance than in the other setup.

4.2.10 Maximum Versus Sum Neurons

The model neurons used here use the maximum over all input signals instead of the sum. The reason is that the sum would mix up many different signals, while only one can be the correct one, i.e. the total input would be the result of one correct signal and many misleading ones. Hence the signal-to-noise ratio would be very low. I have observed an example where even a model identical to the image was not picked up as the correct

one, because the sum over all the accidental input signals favored a completely different-looking person. For that reason I introduced the maximum input function, which is reasonable since the correct signal is likely to be the strongest one. The maximum rule has the additional advantage that the dynamic range of the input into a single cell does not vary much when the connectivity develops, whereas the signal sum would decrease or increase significantly during synaptic re-organization depending on the normalization rule. Thus the blobs would either loose their alignment or would be driven into saturation.

4.3 Experiments

4.3.1 Database

As a face database I used galleries of 111 different persons. Of most persons there is one neutral frontal view, one frontal view of different facial expression, and two views rotated in depth by 15 and 30 degrees respectively. The neutral frontal views serve as a model gallery, and the other three are used as test images for recognition. The models, i.e. the neutral frontal views, are represented by layers of size 10×10 (see Figure 4.1). Though the grids are rectangular and regular, i.e. the spacing between the nodes is constant for each dimension, the graphs are scaled horizontally in the x- and vertically in the y-direction and are aligned manually: The left eye is always represented by the node in the fourth column from the left and the third row from the top, the mouth lies on the fourth row from the bottom, etc. The x-spacing ranges from 6.6 to 9.3 pixels with a mean value of 8.2 and a standard deviation of 0.5. The y-spacing ranges from 5.5 to 8.8 pixels with a mean value of 7.3 and a standard deviation of 0.6. An input image of a face to be recognized is represented by a 16×17 layer with an x-spacing of 8 pixels and a y-spacing of 7 pixels. The image graphs are not aligned, since that would already require recognition. The *size* variations of up to a factor of 1.5 in the x- and y-spacings must be compensated for by the DLM process.

4.3.2 Technical Aspects

DLM in the form presented here is computationally expensive. I have performed single recognition tasks with the complete system, but for the experiments referred to in Table 4.3 I modified the system in several respects to achieve a reasonable speed. I split up the simulation into two phases. The only purpose of the first phase is to let the attention blob become aligned with the face in the input image. No modification of the connectivity was applied in this phase, and only one *average model* was simulated. Its connectivity W^a was derived by taking the maximum synaptic weight over all models for each link:

$$\begin{aligned} W^a_{ij}(t_0) &= \max_{pq} W^{pq}_{ij}(t_0), \\ \dot{W}^a_{ij}(t) &= 0. \end{aligned} \tag{4.22}$$

This attention period takes 1000 time steps. Then the complete system, including the attention blob, is simulated, and the individual connection matrices are subjected to

DLM. Neurons in the model layers are not connected to all neurons in the image layer, but only to an 8×8 patch. These patches are evenly distributed over the image layer with the same spatial arrangement as the model neurons themselves. This still preserves full translation invariance. Full rotation invariance is lost, but the jets used are not rotation invariant in any case. The link dynamics is not simulated at each time step, but only after 200 simulation steps or 100 time units. During this time a running blob moves about once over all of its layer, and the correlation is integrated continuously. The simulation of the link dynamics is then based on these integrated correlations, and since the blobs have moved over all of the layers, all synaptic weights are modified. For further increase in speed, models that are ruled out by the winner-take-all mechanism are no longer simulated; they are just set to zero and ignored from then on ($\beta_\theta = \infty$). The CPU time needed for the recognition of one face against a gallery of 111 models is approximately 10–15 minutes on a Sun SPARCstation 10-512 with a 50 MHz processor.

In order to avoid border effects, the image layer has a frame with a width of 2 neurons without any features or connections to the model layers. The additional frame of neurons helps the attention blob to move to the border of the image layer. Otherwise it would have a strong tendency to stay in the center.

4.3.3 Results

Figure 4.6 shows two recognition examples, one using a test face rotated in depth and the other using a face with a very different expression. In both cases the gallery contains five models. Due to the tight connections between the models, the layer activities show the same variations and differ only very little in intensity. This small difference is averaged over time and amplified by the recognition dynamics that rules out one model after the other until the correct one survives. The examples were monitored for 2000 units of simulation time. An attention phase of 1000 time units had been applied before, but is not shown here. The second recognition task was obviously harder than the first. The sum over the links of the connectivity matrices was even higher for the fourth model than for the correct one. This is a case where the DLM is actually required to stabilize the running blob alignment and recognize the correct model. In many other cases the correct face can be recognized without modifying the connectivity matrix.

Recognition rates for *galleries* of 20, 50, and 111 models are given in Table 4.3. As is already known from previous work (Lades et al., 1993), recognition of depth-rotated faces is in general less reliable than, for instance, recognition of faces with an altered expression (the examples in Figure 4.6 are not typical in this respect). It is interesting to consider recognition times. Although they vary significantly, a general tendency is noticeable: Firstly, more difficult tasks take more time, i.e. recognition time is correlated with error rate. This is also known from psychophysical experiments (see for example Bruce et al., 1987; Kalocsai et al., 1994). Secondly, incorrect recognition takes much more time than correct recognition. Recognition time does not depend very much on the size of the gallery.

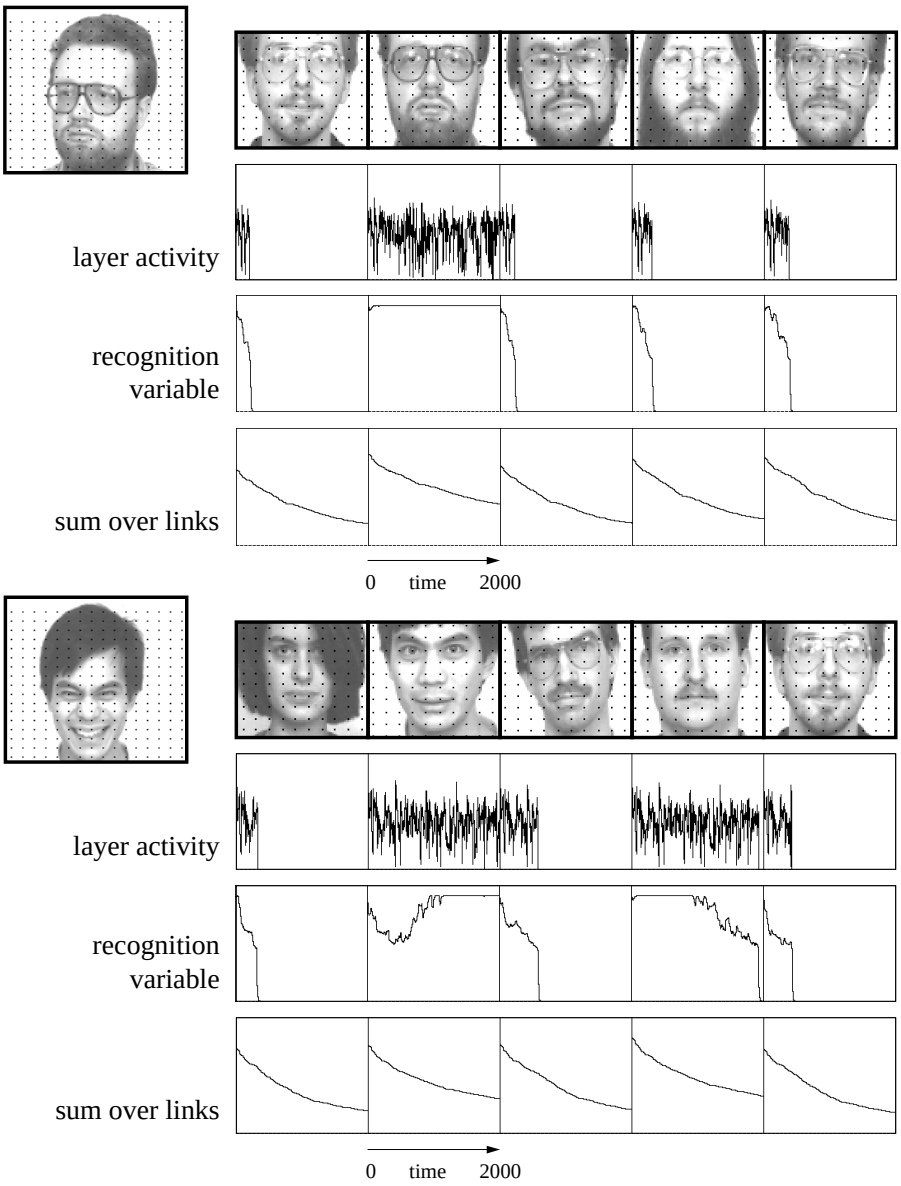

Figure 4.6: Simulation examples of DLM recognition. The test images are shown on the left with 16×17 neurons indicated by black dots. The models have 10×10 neurons and are aligned with each other. The respective total layer activities, i.e. the sum over all neurons of one model, are shown in the upper graphs. The most similar model is usually slightly more active than the others. On that basis the models compete against each other, and eventually the correct one survives, as indicated by the recognition variable. The sum over all links of each connection matrix is shown in the lower graphs. It gives an impression of the extent to which the matrices self-organize before the recognition decision is made.

Gallery Size	Test Images	Correct Recognition #	Rate %	Recognition Time for Correct Recognition	Incorrect Recognition
20	111 rotated faces (15 degrees)	106	95.5	310 ± 400	5120 ±3570
	110 rotated faces (30 degrees)	91	82.7	950 ±1970	4070 ±4810
	109 frontal views (grimace)	102	93.6	310 ± 420	4870 ±6010
50	111 rotated faces (15 degrees)	104	93.7	370 ± 450	8530 ±5800
	110 rotated faces (30 degrees)	83	75.5	820 ± 740	5410 ±7270
	109 frontal views (grimace)	95	87.2	440 ±1000	2670 ±1660
111	111 rotated faces (15 degrees)	102	91.9	450 ± 590	2540 ±2000
	110 rotated faces (30 degrees)	73	66.4	1180 ±1430	4400 ±4820
	109 frontal views (grimace)	93	85.3	480 ± 720	3440 ±2830

Table 4.3: Recognition results against a gallery of 20, 50, and 111 neutral frontal views. Recognition time (with two iterations of the differential equations per time unit) is the time required until all but one models are ruled out by the winner-take-all mechanism.

4.4 Discussion

The two main features of the system presented here compared to the preceding stationary blob system are the continuous and autonomous dynamics and the fact that the system actually performes face *recognition* on a large gallery. This latter is definitely a success. The former seems to be a conceptual step forward as well, but it is worthwhile to discuss the advantages and drawbacks of the two different dynamics more thoroughly. The first advantage of the running blob dynamics is obvious: It requires no external control schedule (in the sense of a certain sequence of phases such as required for the stationary blob dynamics, for which the layer dynamics, the link dynamics, and a complete reset of the layer dynamics iterate). Its second advantage is that running blobs potentially convey topography faster and more reliably. Although the blobs may jump, their generally continuous motion enforces continuity in the mapping much more than a sequence of independent stationary blobs.

Nevertheless the running blobs have some disadvantages: Firstly, if the blobs in the image and the model layer have started at non-corresponding positions, they run independently of each other for quite a while and may even cross each other's path before they lock onto each other and run in alignment from then on. In the stationary blob dynamics, each new blob in the image layer has the chance of producing a corresponding blob in the model layer independently of the previous one. Therefore the stationary blobs may align faster. Secondly, the running blobs have the strong tendency to move straight over the whole layer. That causes problems if the layers are of different size or format and requires additional control dynamics in form of the attention window. (Though the old system had the problem that if the image was larger than the model, many blobs in the image layer were placed at locations without any counterpart in the model layer. A mechanism like the attention blob would probably have been useful in that system as well.) Thirdly, the paths of the running blobs are not random but are partially determined by the input from the other layers, which remains the same

for a given location of a blob. Thus certain paths dominate and topology is encoded inhomogeneously: strongly along typical paths and weakly elsewhere.

For these reasons, further research will have to investigate alternatives to both blob dynamics in order to find an optimal dynamics. Some experiments have recently been made with layers of coupled Bonhoeffer-van der Pol oscillators generating plane running waves (SCHWARZ, 1995). Plane waves are supposed to encode topology much faster than the running blobs, because in theory only two successive waves running perpendicularly to each other suffice to determine all locations uniquely. The problem of plane waves is that they have such strongly autonomous dynamics that they need a long time to align and then they have usually passed the layer already. Therefore the running wave model is still slower than the running blob model.

Beside these layer dynamics issues, there are many directions in which the system could be further developed to make it more complete and realistic: It has not yet been investigated how new models can be added to the gallery in a neural fashion, it will be necessary to introduce hierarchy into the recognition process, and more control structure for context knowledge is required, to mention only a few aspects.

Chapter 5

Face Recognition by Elastic Graph Matching

Abstract: The face recognition system presented below is based on Elastic Graph Matching (EGM) as an algorithmic version of Dynamic Link Matching. Individual faces are represented as labeled graphs. Nodes are labeled with jets; edges are labeled with distance vectors. The graphs are object-adapted, i.e. nodes are located at fiducial points, such as eyes, tip of the nose, corners of the mouth, etc. In order to be able to represent a wide range of different faces, a collection of individual face graphs is fused to a General Face Knowledge (GFK), a graph structure in which a set of alternative jets instead of only one is attached to a node. With the GFK, probe faces can be represented as a nodewise composition of the known sample faces, which makes the system more reliable on unknown faces. A similarity function is defined to compare two graphs, taking into account the similarities of the individual jets and the relative distortion of the graphs. New image graphs are generated by maximizing this similarity between the GFK and a sequence of image graphs selected from an image. This process is known as Elastic Graph Matching. Different views are represented by graphs or GFKs which differ in structure. For matching and recognition, only jets referring to corresponding fiducial points are compared. Recognition results are given for galleries of 300 faces. Performance is good on frontal views against frontal views but relatively poor on different views, e.g. half-profile against frontal view.

5.1 Introduction

In Chapter 4 faces were represented by layers of neurons, and the whole process of matching and recognition was achieved by neural dynamics. *Topography* was induced by lateral connections and a particular layer dynamics, matching was performed by synchronization and link dynamics, and the recognition dynamics finally detected the correct face. In this chapter, I present an algorithmic version of the very same basic ideas. But topography is here explicitely expressed by edge labels, the matching is performed by maximizing a similarity function, and recognition is based on the resulting similarity values, taking the most similar model as the correct face. This algorithmic formulation is more appropriate for technical applications, since the matching is much faster and more flexible than in the neural formulation.

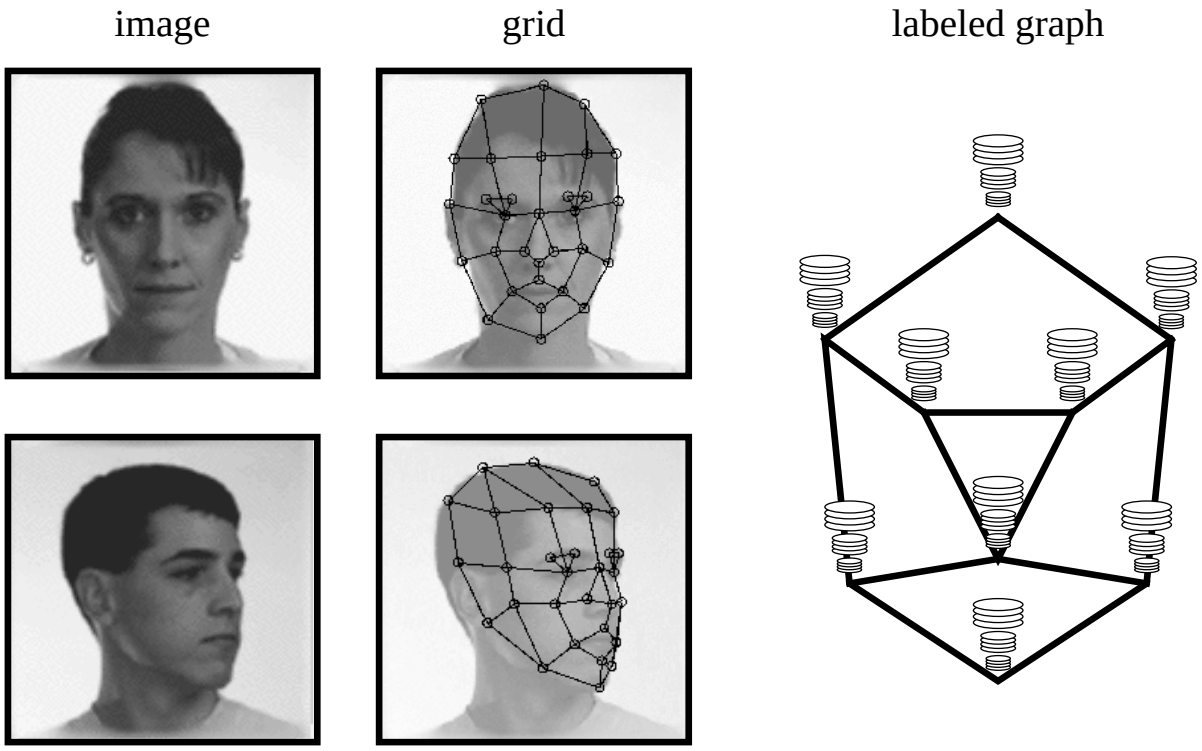

Figure 5.1: Labeled graphs representing faces. Shown here are two faces of different pose (left) and the manually defined grids (middle). Nodes are placed at fiducial points, which are assumed to be important and easy to find. On the right a sketch of a graph labeled with jets is shown schematically.

5.2 The System

5.2.1 Face Representation

Individual Faces

For faces, a set of *fiducial points* is defined, e.g. the pupils, the corners of the mouth, the tip of the nose, the top and bottom of the ears, etc. A *labeled graph* $\mathcal{G}$ representing a face consists of N nodes on these fiducial points at positions $\vec{x}_n, n = 1, ..., N$ and E edges between them. The nodes are labeled with *jets* $\mathcal{J}_n$. The edges are labeled with distances $\Delta\vec{x}_e = \vec{x}_n - \vec{x}_{n'}, e = 1, ..., E$, where edge e connects node n' with n. Hence the edge labels are two-dimensional vectors and represent the *topography* of the graph. This *face* or *model graph* is *object-adapted*, since the nodes are selected from face-specific points (fiducial points, see Figure 5.1).

Graphs of *different views* differ in geometry and local features. Although the fiducial points refer to corresponding object points, some may be occluded, and jets as well as distances vary due to rotation in depth. In order to be able to compare graphs of different views, pointers have to be defined that associate nodes of different graphs, referring to corresponding fiducial points. This was done manually.

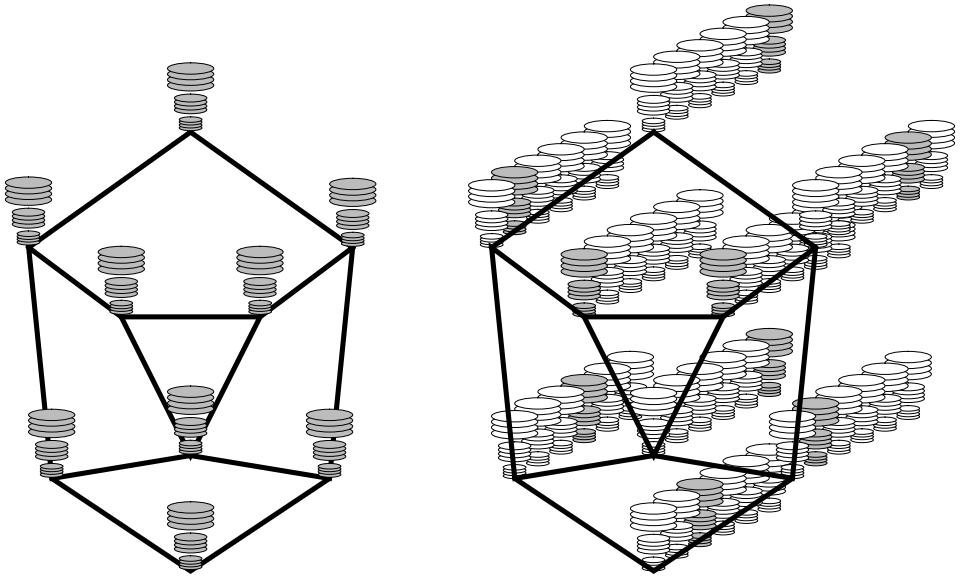

Figure 5.2: General Face Knowledge (GFK) serves as a representation of faces in general. It is designed to cover all possible variations in appearance of faces. In order to do so, it has an average grid and a whole set of jets at each node. An image graph to be compared with the GFK has only one jet per node. In the comparison, the best fitting jet in the GFK is selected for each node independently, indicated as grey jets in rows of white ones.

General Face Knowledge

In order to deal with new faces, one needs a representation for faces in general rather than models of individual faces. This representation should cover a wide range of possible variations in the appearance of faces, such as differently shaped eyes, mouths, or noses, different types of beards, variations due to gender and age, etc. I call this representation *General Face Knowledge* or GFK and denote it with $\mathcal{K}$. Notice that no explicit face model is employed. Instead, for a given view, M model graphs $\mathcal{G}^{\mathcal{K}m}$ $(m = 1, ..., M)$ of identical structure taken from different sample faces are combined. The nodes of the GFK are labeled with corresponding sets of jets $\mathcal{J}_n^{\mathcal{K}m}$; the edges are labeled with the averaged distances $\Delta\vec{x}_e^{\mathcal{K}} = \sum \Delta\vec{x}_e^m / M$. The GFK represents not only the sample faces, but also all faces that can be obtained by combining the local features of different sample faces: the mouth from one face, the nose from a second, parts of the hair from a third, etc. Each fiducial point may be represented by a different sample face (see Figure 5.2).

5.2.2 Generating a Face Representation by Elastic Graph Matching

So far I have only described how individual faces and the GFK are represented by labeled graphs. I am now going to explain how these *graphs* are *generated*.

The simplest method is to do so *manually*. For a given image a set of fiducial points has to be marked and edges between them have to be drawn. The edge labels can be computed as the differences between the pixel positions. This defines a *grid*, i.e. the

structural and metric information about a graph. Finally the Gabor wavelet transform provides the jets for the nodes. This is actually the method for generating initial graphs for the system. For each view one graph has to be defined by hand, including the pointers indicating which nodes in different views correspond to each other.

If the system has a GFK (possibly consisting of one model only), graphs for new images can *automatically* be generated by *Elastic Graph Matching*. In the beginning, when the GFK contains only very few faces, one has to review and correct the result of the graph matching, but once the face knowledge is rich enough (approximately 70 graphs) one can rely on the matching and generate large *galleries* of faces automatically.

Similarity Function for Matching

The key role in Elastic Graph Matching (EGM) is played by a function evaluating the *graph similarity* between an *image graph* and the GFK of identical view. It depends on the *jet similarities* and the *distortion* of the image grid relative to the GFK grid. For a graph $\mathcal{G}^{\mathcal{I}}$ with nodes $n = 1, ..., N$ and edges $e = 1, ..., E$ and a GFK $\mathcal{K}$ with model graphs $m = 1, ..., M$ the similarity is defined as

$$\mathcal{S}_{\mathcal{K}}(\mathcal{G}^{\mathcal{I}}, \mathcal{K}) = \frac{1}{N} \sum_n \max_m \left(\mathcal{S}_\phi(\mathcal{J}_n^{\mathcal{I}}, \mathcal{J}_n^{\mathcal{K}m}) \right) - \frac{\lambda}{E} \sum_e (\Delta \vec{x}_e^{\mathcal{I}} - \Delta \vec{x}_e^{\mathcal{K}})^2, \tag{5.1}$$

where λ determines the relative importance of jets and metric. $\mathcal{J}_n$ are the jets at node n and $\Delta \vec{x}_e$ are the distance vectors used as labels at edges e. Since the GFK provides several jets for each fiducial point, the best one is selected and used for comparison. This best fitting jet serves as the *local expert* for the image face.

Matching Schedule

The goal of EGM on a probe image is to find the fiducial points and thus to select from all possible graphs in the image the one that maximizes the similarity with the GFK. In practice one has to apply a heuristic algorithm to find a good approximation to the optimum in a reasonable amount of time. I use a coarse to fine approach. The matching schedule has the following stages:

Stage 1 Find the face in the image: Average over the amplitudes of the jets in the GFK and generate an *average graph*, or alternatively select one arbitrary graph as a representative. Use this as a rigid model ($\lambda = \infty$) and evaluate its similarity at each location of a square lattice with a spacing of 4 pixels. At this stage the similarity function $\mathcal{S}_a$ without phase is used instead of $\mathcal{S}_\phi$. Repeat the scanning around the best fitting position with a spacing of 1 pixel. The best fitting position finally serves as starting point for the next stage.

Stage 2 Find the right position and size of the face: Now the GFK is used without averaging. The GFK grid is varied in position and size. Check the four different positions $(\pm 3, \pm 3)$ pixels displaced from the position found in Stage 1, and at each position check two different sizes which have the same center position, a factor of 1.18 smaller or larger than the GFK average size. This is without effect on the metric similarity, since the vectors $\vec{x}_e^{\mathcal{K}}$ are transformed accordingly. I still keep $\lambda = \infty$. For each of these eight variations the best fitting jet for each node

is selected and its displacement according to Equation A.11 is computed. This is done with a *focus* of 1, i.e. the displacements may be of a magnitude up to half the wavelength of the lowest frequency kernel. The grids are then rescaled and repositioned in order to minimize the square sum over the displacements.

Stage 3 Find the right size and format of the face: A similar relaxation process as described for Stage 2 is applied, relaxing the x- and y-dimension independently now. In addition the focus increases successively from 1 to 5.

Stage 4 Local distortion: In a pseudo-random sequence the position of each individual image node is varied in order to increase further the similarity to the GFK. Now the metric similarity is taken into account by setting $\lambda = 2$ and using the vectors $\vec{x}_e^{\mathcal{K}}$ as obtained in Stage 3. In this stage only positions are used where the estimated displacement vector is small ($d < 1$, see Equation A.11). For this *local distortion* the focus again increases from 1 to 5.

The resulting graph is called the *image graph* and is stored as a representation for the individual face of the image (see Figure 5.3).

Normalizing Face Size

The original images have a format of 256×384 pixels, and the faces vary in size by about a factor of 3. In order to compensate for *size variation* and transform the images into the 128×128 pixel format that is used in the system, I use a *preprocessing* stage developed by Krüger (1994). The preprocessing uses the very same EGM as described above to estimate size and position of a face, but a GFK with fewer nodes is used, and it is split into three different size categories. Once the size and position of the face in the original image is known, an appropriate frame can be selected and resized to the required 128x128 format.

5.2.3 Recognition

EGM with the GFK allows us to generate graphs for probe faces automatically. By this means one can build up large galleries of model graphs without further need for matching or distortion if one compares faces with each other. A *gallery* is distinct from the *GFK*, since the former represents a set of individual faces to be recognized, while the latter represents what the system knows about faces in general and is used to generate graphs. In addition I distinguish between *image graphs/galleries* and *model graphs/galleries*. The latter represent the stored faces known to the system, while the former represent the probe faces to be recognized by comparison with the models.

For comparing graphs, I use a very simple similarity function simply averaging over the similarities between the corresponding jets, ignoring the distortions created by rotation in depth. If image graph and model graph are of different view, one has to take care that only jets belonging to the corresponding fiducial point are compared with each other. Assume the image graph $\mathcal{G}^{\mathcal{I}}$ has N nodes of which N' have a corresponding node in model graph $\mathcal{G}^{\mathcal{M}}$. n' runs over nodes with a counterpart, e.g. $n' = 1, .., 7, 9, 11, .., N-1$ if nodes 8, 10, and N have no counterpart. Node $n_{n'}$ in the model graph corresponds to

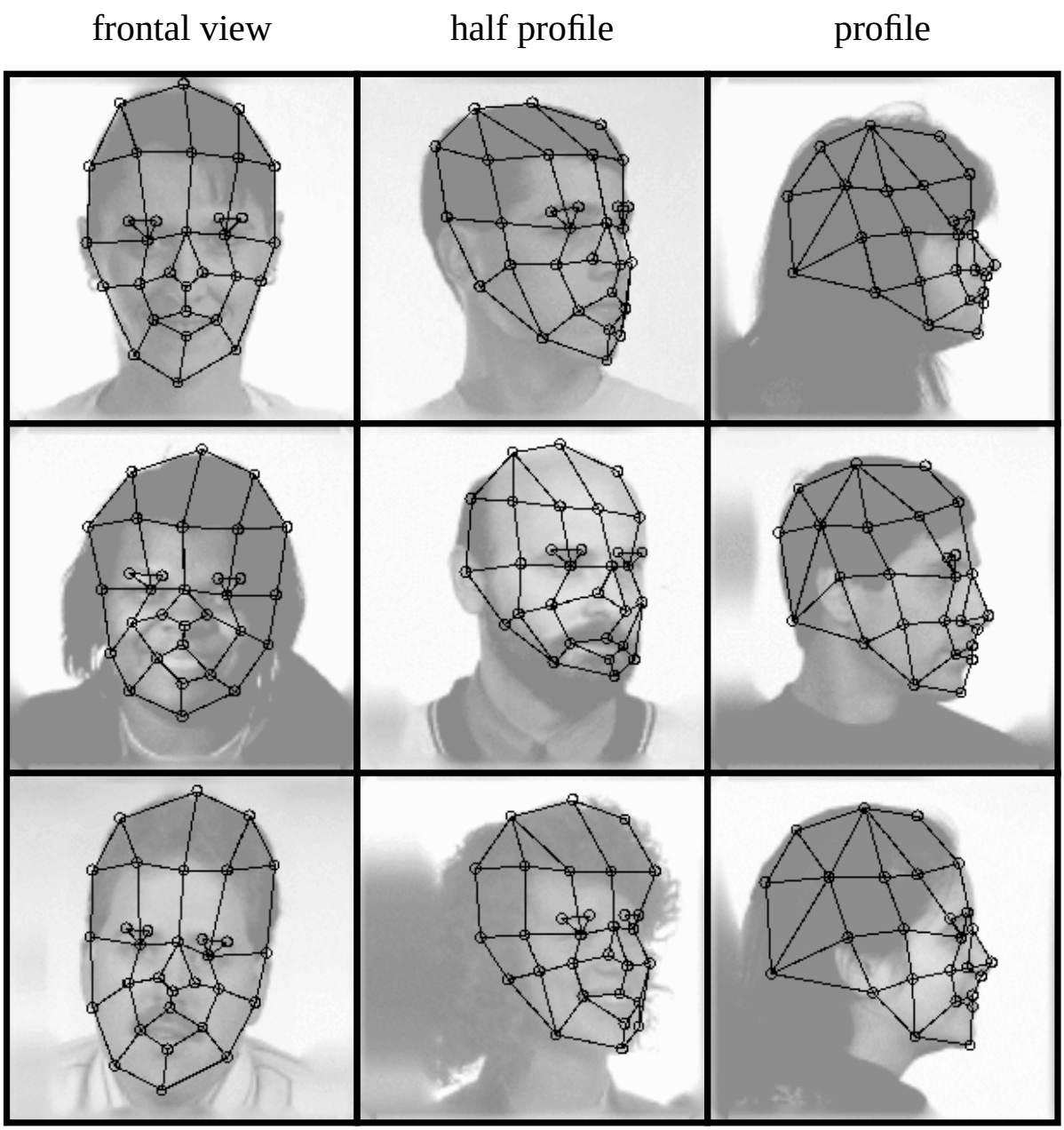

Figure 5.3: Sample grids as generated automatically by EGM against the GFK. One can see that in general the matching finds the fiducial points quite accurately. But mismatches occurred for example for the face in the center. The chin was not found accurately, because of the beard. The leftmost node and the node below it should be at the top and the bottom of the ear respectivly. See the model above for a correct match. The graphs used in Section 5.3.2 had about 14 additional nodes which are not shown here for simplicity.

node n' in the image graph. I then define *graph similarity* as:

$$\mathcal{S}_{\mathcal{G}}(\mathcal{G}^{\mathcal{I}}, \mathcal{G}^{\mathcal{M}}) = \frac{1}{N'} \sum_{n'} \mathcal{S}_a(\mathcal{J}^{\mathcal{I}}_{n'}, \mathcal{J}^{\mathcal{M}}_{n_{n'}}). \tag{5.2}$$

Here the *jet similarity* function without phase turned out to be more discriminative.

Recognition with Confidence

Given one image graph $\mathcal{G}^{\mathcal{I}}$ and a gallery of model graphs $\{\mathcal{G}^{\mathcal{M}}_m | m = 1, ..., M\}$ consider the *distribution* of the similarities

$$\mathcal{S}_m = \mathcal{S}_{\mathcal{G}}(\mathcal{G}^{\mathcal{I}}, \mathcal{G}^{\mathcal{M}}_m). \tag{5.3}$$

In this the correct model usually stands out with a significantly higher value than all others. To quantify this, I have adopted the *confidence criterion* of (LADES et al., 1993). Assume that the models are ordered such that $\mathcal{S}_m > \mathcal{S}_{m+1}$. The confidence $\mathcal{C}$ is defined as

$$\mathcal{C}(\mathcal{G}^{\mathcal{I}}, \{\mathcal{G}^{\mathcal{M}}_m\}) = \frac{\mathcal{S}_1 - \mathcal{S}_2}{s}, \tag{5.4}$$

where s is the standard deviation of set $\{\mathcal{S}_m | m = 2, ..., M\}$. A face is considered to be *recognized with confidence* if $\mathcal{C}(\mathcal{G}^{\mathcal{I}}, \{\mathcal{G}^{\mathcal{M}}_m\})$ is larger or equal to a certain threshold $\mathcal{C}_\theta$. This criterion is relative in the sense that a global shift and a global scaling of the similarity distribution do not matter. These global variations might be due to a different pose of the face in the image or variations in the set of jets or coefficients used for the comparison. Nevertheless this criterion is rather heuristic, and a more theoretically motivated confidence criterion would be valuable. The main disadvantage of the criterion is that it depends on the gallery size and that it is not directly applicable to *mixed galleries*, i.e. galleries containing models of different views.

With a confidence criterion, the recognition samples fall into four classes:

	First rank model is accepted $\mathcal{C}(\mathcal{G}^{\mathcal{I}}, \{\mathcal{G}^{\mathcal{M}}_m\}) \geq \mathcal{C}_\theta$	First rank model is rejected $\mathcal{C}(\mathcal{G}^{\mathcal{I}}, \{\mathcal{G}^{\mathcal{M}}_m\}) < \mathcal{C}_\theta$
First rank model is correct	true positives	false negatives
First rank model is not correct	false positives	true negatives

The goal of the face recognition system without confidence criterion is to maximize the *raw recognition rate*, i.e. to minimize the false cases. Given the raw recognition rate, the purpose of the confidence criterion is to discriminate the false cases from the true cases, i.e. to minimize *false negatives* and *false positives* while maximizing *true positives* and *true negatives*. The threshold $\mathcal{C}_\theta$ determines the distribution over the classes. A high threshold will provide high reliability on rejecting false positives, a low threshold will provide high reliability on accepting correct models.

Figure 5.4: Sample faces from the ARPA/ARL FERET database: frontal views A and B, quarter views, half-profile, and profile. The images shown here are already rescaled to a normal size by the preprocessing stage. Notice the variation in the rotation angle for the quarter views and half-profiles.

5.3 Experiments

5.3.1 Database

The galleries of images are taken from the *ARPA/ARL FERET database* provided by the US Army Research Laboratory. For the test I used four *different views*: frontal view, quarter view (about 20 degrees rotated), half-profile (about 40-70 degrees rotated), and profile (see Figure 5.4). Some are rotated to the left and others to the right. The views are known to the system. As the simplest invariance transformation I flip all right views to left views, assuming that since faces are sufficiently symmetrical, this is a useful manipulation to recognize half-left profile against half-right profile. For most faces there are two frontal views with *different facial expression*. Apart from a few exceptions the persons have no disguise or variations in hairstyle or clothing. The background is always homogeneous, exept for smoothly varying shadows, and sometimes light and sometimes grey. The *size* of the faces varies by about a factor of three, but is constant for each individual. I therefore rescaled all faces (see 'Normalizing face size' in Section 5.2.2). The format of the original images is 256x384 pixels.

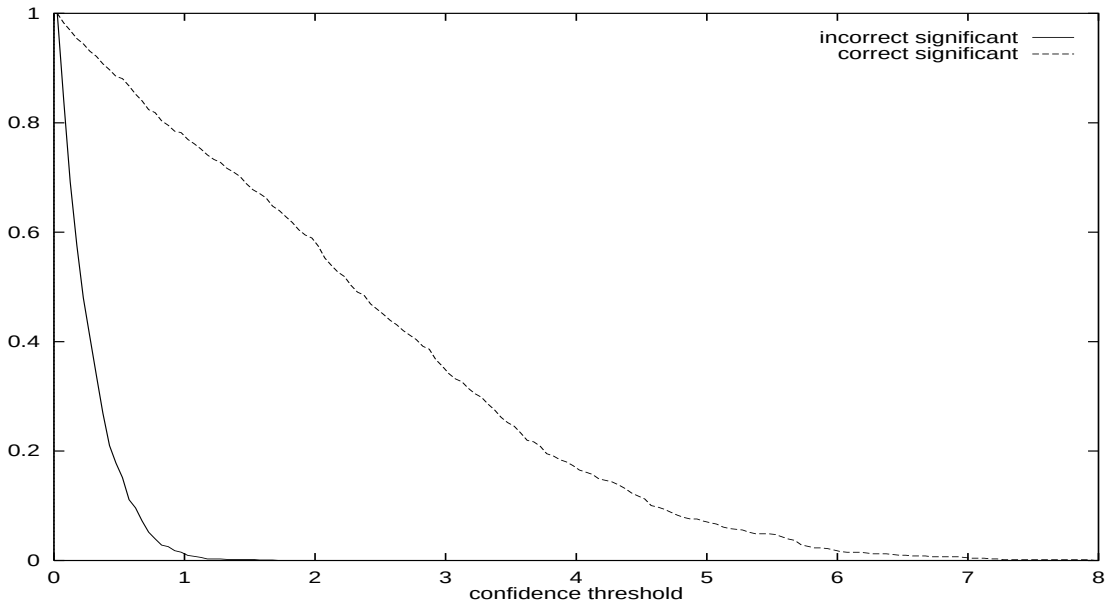

Figure 5.5: Significant recognition of correct and incorrect first rank models for all gallery pairs listed in Table 5.1 combined. The solid line shows the proportion of false positives and the dashed line the proportion of true positives, depending on the confidence threshold. The former should be as low as possible, while the latter should be as high as possible. For the results in Table 5.1 I chose a confidence threshold of $\mathcal{C}_\theta = 1$.

5.3.2 Results

For the experiments I used model galleries of 300 faces with only one image per person. One complete recognition, i.e. normalizing face size, generating the face graph, and comparison with 300 models, takes approximately 20 seconds on a Sun SPARCstation 20-502 with a 50 MHz processor.

Recognition results are shown in Table 5.1. For frontal views against frontal views the results are very good. Recognizing faces of different pose turns out to be a much harder task; the recognition rates are relatively poor. The results are asymmetrical for different poses. Performance is better if frontal views or profiles serve as galleries than if half-profiles are used. This is due to the fact that frontal views as well as profiles are much more standardized in pose than half-profiles, where the angle varies between 40 and 70 degrees. Since the *graph similarities* degrade with rotation angle independently of the individuals, the 40 degrees half-profile models are favored if compared with a frontal-view image instead of with profile. Analogously, the 70 degrees half-profile models are favored if compared with a profile image instead of with frontal view. This effect degrades recognition performance. The results are significantly better for quarter views right than for quarter views left. One reason might be that the left views are flipped while the right views are not. But the more likely reason is that on average the right views are less rotated in depth than the left.

Figure 5.5 shows the proportion of true positives relative to all true cases and false positives relative to all false cases. The *confidence criterion* would work perfectly if there were no false positives and no true negatives. But this is not the case and therefore one has to compromise between too many false positives and too few true positives. From

model gallery	probe images	first 15 ranks lower ranks #	 %	first rank lower ranks #	 %	true pos. false pos. #	 %	false neg. true neg. #	 %
300	300	297	99.0	292	97.3	276	92.0	16	5.3
frontal views A	frontal views B	3	1.0	8	2.7	0	0.0	8	2.7
300	300	298	99.3	294	98.0	266	88.7	28	9.3
frontal views B	frontal views A	2	0.7	6	2.0	0	0.0	6	2.0
300	23	23	100.0	15	65.2	10	43.5	5	21.7
frontal views A	quarter views right	0	0.0	8	34.8	0	0.0	8	34.8
300	23	15	65.2	7	30.4	2	8.7	5	21.7
frontal views A	quarter views left	8	34.8	16	69.6	0	0.0	16	69.6
300	300	132	44.0	40	13.3	8	2.7	32	10.7
frontal views A	half-profiles	168	56.0	260	86.7	1	0.3	259	86.3
300	300	103	34.3	38	12.7	4	1.3	34	11.3
half-profiles	frontal views A	197	65.7	262	87.3	0	0.0	262	87.3
300	300	104	34.7	33	11.0	6	2.0	27	9.0
half-profiles	profiles	196	65.3	267	89.0	3	1.0	264	88.0
300	300	120	40.0	41	13.7	8	2.7	33	11.0
profiles	half-profiles	180	60.0	259	86.3	6	2.0	253	84.3

Table 5.1: Recognition results for cross-runs between different galleries. The number of gallery models and probe images and their pose is displayed in the first and second column respectively. In all other entries of the table, four figures are given. On the left are the absolute numbers, on the right the respective percentages. The upper figures refer to good rankings or correct recognition cases, the lower ones refer to poor rankings or incorrect recognition. The third column says how often the correct face is among the 15 best models. The number of correctly recognized faces (i.e. if the correct model has the highest similarity with the image) is given in the next column. In the last two columns the confidence criterion is applied with a threshold of 1, significant recognition in the left column and rejection in the right column. Notice that the numbers in the last two columns add up to the numbers in the fourth column.

Figure 5.5, I have choosen a *confidence threshold* of $\mathcal{C}_\theta = 1$ in order to avoid false positives. The results in Table 5.1 were obtained with this threshold.

5.4 Discussion

I have presented a general and flexible system applied to face recognition. It is designed for an *in-class recognition* task, i.e. for recognizing members of a known class of objects, but the system is in no way tailored to faces. In principle it should be directly applicable to other in-class recognition tasks such as recognizing individuals of a given animal species, given the same level of standardization of the images. In contrast to many neural network systems, no extensive *training* for new faces or new object classes is required. The individuals are simply shown to the system once.

The performance is high on faces of the same pose. Recognizing unfamiliar faces in very different poses is a much more difficult task and the performance of the system is significantly degraded in that case. It is known from psychophysical experiments that human subjects perform poorly on recognizing faces taken from *different views*, as well. Bruce et al. (1987) showed that reliability on judging whether two unfamiliar faces are the same degrades significantly with rotation angle in depth. A similar result was obtained by Kalocsai et al. (1994) if no *easy features* such as hairstyle, type of beard, wearing glasses or not, are available.

5.4.1 Comparison with the Preceding System

Compared to the preceding system of Lades et al. (1993) I have made three major modifications. The first two are of general advantage; only the last one focusses specifically on face recognition or rather on in-class recognition tasks. *Phase information* was used for better positioning of the nodes on the fiducial points, *object-adapted graphs* were introduced to deal with different views, and a set of sample graphs was combined to a *General Face Knowledge* in order to represent a wide range of different and previously unknown faces.

The modified system has several advantages. Firstly, the previous system (Lades et al., 1993) matched each model of the gallery separately to a face image. By introducing the GFK and by using phase information, image graphs can be generated with good reliability, even if no image of that particular person has been shown to the system before. This makes it possible to separate the graph generation phase from the recognition phase, which makes the system much faster by generating an image graph only once and not for each model again. Secondly, the flexible graphs provide a way to deal with very different poses. Nodes can refer to the same fiducial points regardless of view. That is essential for many operations that one wants to apply to the graphs (cf. next section). Thirdly, using phase information provides relatively precise node locations that can potentially be used as an additional recognition cue (though topography is not used for recognition in the current system). Previously the localization of the nodes was very rough and of little use for the recognition.

5.4.2 Comparison with Other Systems

There is a considerable literature on face recognition, and many different techniques have been applied to this task (see SAMAL & IYENGAR, 1992; VALENTIN et al., 1994 for reviews). Since recognition results depend very much on database design, a comparison of performance would not be meaningful, but it is worthwhile to do a comparison under conceptual aspects.

Several systems are designed specifically for faces on the basis of manually defined features. YUILLE (1991), for example, represents eyes by a circle within an almond-shape and defines an energy function to optimize a total of 9 parameters of this model for matching it to an image. BRUNELLI & POGGIO (1993a, 1993b) similarly employ specific models for eyebrows, nose, mouth, etc. and derive 35 geometrical features such as eyebrow thickness, nose width, mouth width, and eleven radii describing the chin shape. The drawback of these systems is that the features as well as the prcedures to extract them must be defined and programmed by the user for each object class again, and the system has no means to adapt to samples for which the features fail. For example, the eye models mentioned above may fail for faces with sun glasses or have problems if the eyes are closed. The chin radii cannot be extracted if the face is bearded. In these cases the user has to design new features and new algorithms to extract them. With this paradigm, the system can never become autonomous, it will always depend on the user and programmer. The system presented here consequently avoids such user defined features (except the user defined locations of the fiducial points in the beginning, which has to be replaced by autonomous procedures, see following section). Within the EGM approach, such exceptions as faces with sun glasses or a beard can very naturally and automatically be included into the GFK, and it was mentioned above that the system should be directly applicable to other classes of objects.

Another approach to face recognition not using manually defined features is based on *Principal Component Analysis* (PCA) (SIROVICH & KIRBY, 1987; KIRBY & SIROVICH, 1990; TURK & PENTLAND, 1991; O'TOOLE et al., 1993). In this approach, faces are first aligned with each other and then treated as high-dimensional vectors (this alignment is frequently done manually or by means of manually defined features, but it can also be done automatically within the PCA framework, see TURK & PENTLAND, 1991). The PCA computes eigenvectors, so-called *eigenfaces*, and the respective eigenvalues. Each probe face is decomposed with respect to these eigenvectors and represented by the corresponding coefficients in a very efficient way (approximately 30 suffice to obtain a good reconstruction). PCA is optimal with respect to compression, but its appropriateness for recognition purposes can not be shown theoretically. It is known that the first eigenvectors capture mainly general information about faces and are therefore not as discriminative as eigenvectors with lower eigenvalue (O'TOOLE et al., 1993). Thus the discriminative features are not optimally represented by eigenvectors.

In contrast to our EGM system, PCA is a completely holistic approach. Thus one obvious disadvantage is that it has conceptually no means to deal with occlusions as is demonstrated for the more localized EGM in Chapter 7. A second disadvantage is that geometry is tightly coupled with local features. As was already discussed in Chapter 1, geometrical variations, such as a different nose–mouth distance can thus not be coded by a displacement, but has to be treated as a completely new face, with a different mouth and/or nose. As a solution to this problem one can first apply a procedure which

compensates for geometrical variations and generates a so-called shape-free face model (LANITIS et al., 1995). Then all facial features are aligned with each other and can be optimally encoded by PCA. The way PCA and EGM compose a probe face of known components is very different. That has consequences for the ability to *generalize*. As can be seen in the following chapter, the GFK can, for instance, compose a probe face with glasses and a beard out of two known faces, one beardless face with glasses and one bearded face without glasses. This cannot be done by PCA, since the eigenvectors always represent the whole face. PCA on the other hand is able to combine holistic features in a way the GFK is not able to. While the advantage of the localized composition is obvious, I do not have a clear view of the potential of the holistic composition for generalization.

5.4.3 Future Perspectives

The newly introduced features of the system open many possibilities to improve the system further. The *object-adapted graphs* make it possible to treat the different nodes individually. KRÜGER (1995), for example, has recently introduced trainable node weights to take into account that some fiducial points are more reliable or more robust against rotation in depth than others. This individual treatment is especially important for faces of different pose. MAURER & VON DER MALSBURG (1995) are currently working on linear transformations on the jets in order to compensate for the effect of rotation in depth. However, linear transformation is obviously not sufficient, and one may have to train and apply more general transformations. An alternative approach might be to use the GFKs and do the transformation based on sample faces (see Section 6.4.2).

By using phase information and the GFKs, matching accuracy has improved significantly. However, many partial mismatches still occur. This is probably due to the primitive way topography is encoded in the graphs, distortions being controlled by elastic forces to keep the spatial vectors between two nodes approximately constant. But faces do not distort arbitrarily as in a fun-house mirror. There are rather typical *distortion patterns*, e.g. due to rotation in depth, variations in facial expression, different hairstyles, or different but symmetrical shapes of the faces. It would be of great help if these typical distortion patterns could be analyzed, and if the local distortion could be replaced by a global distortion with much fewer degrees of freedom just covering these typical distortion patterns. One might then possibly get information about the reason for the distortion as well, whether it comes from laughing or rotation in depth of a certain degree. Information about rotation in depth could be especially useful, since a precise pose estimation would make recognition easier. Some research in this direction has, for example, been done by LANITIS et al. (1995). When the matching is reliable enough, it will be interesting to investigate to what extent the grid topography can be used for recognition (cf. BRUNELLI & POGGIO, 1993b).

A further shortcoming of the system is that all graph structures have to be defined manually. That has to be replaced by a self-organizing process able to generate appropriate representations for object classes in an autonomous fashion. This can most easily be done on image sequences, since they provide many cues for grouping, segmentation, and detecting correspondences. For example, nodes could be taken from salient points and grouped on the basis of common motion (cf. MANJUNATH et al., 1992). Monitoring a rotating object by continuously applying EGM can then reveal which nodes refer to corresponding fiducial points in different views (cf. REISER, 1991). A General Object

Knowledge could be established by matching object graphs and combining those which are similar, assuming that they belong to the same class of objects.

Chapter 6

Phantom Faces and Face Analysis

Abstract: In this system the General Face Knowledge (GFK) introduced in the previous chapter is enriched with facial attribute labels such as gender or the presence of a beard or glasses. Elastic Graph Matching provides information about which jet in the GFK best fits the image at which node. The best fitting jet for a node is called the local expert. A composite or phantom face similar in appearance to the original can artificially be composed based on these local experts. The facial attribute labels can be transferred to the phantom face and provide a good cue for determining the facial attributes of the original, for instance, if most local experts belong to female models the original is likely to be female. A statistical analysis based on Bayes' formula is given, and the relative significance of each node for the determination of gender and the presence of a beard or glasses is computed. Results concerning attribute determination are given for a gallery of up to 111 faces.

6.1 Introduction

We have seen in the previous chapter how a graph representation of a probe face can be generated automatically by Elastic Graph Matching against a General Face Knowledge (GFK). Each node of the image graph was allowed to select its best fitting jet from a different model. In this chapter I am going to investigate further possibilities for analyzing a face on the basis of jet similarities between the nodes of a graph and the nodes in a GFK. The matching result is visualized by generating composite or phantom faces. Attributes of the probe face, such as gender, beard, and glasses, are determined on the basis of the corresponding attributes of the models in the general face knowledge. This is a step in the direction of face *analysis* rather than just face *recognition*. If this system works well for several facial attributes it might help to improve face recognition by reducing the search space; this would be especially valuable for recognizing faces in different poses.

6.2 The System

6.2.1 Phantom Faces

First I am going to illustrate how well the *General Face Knowledge* can represent a probe face. Figure 6.1 shows an image, its graph, the GFK, and arrows pointing to the

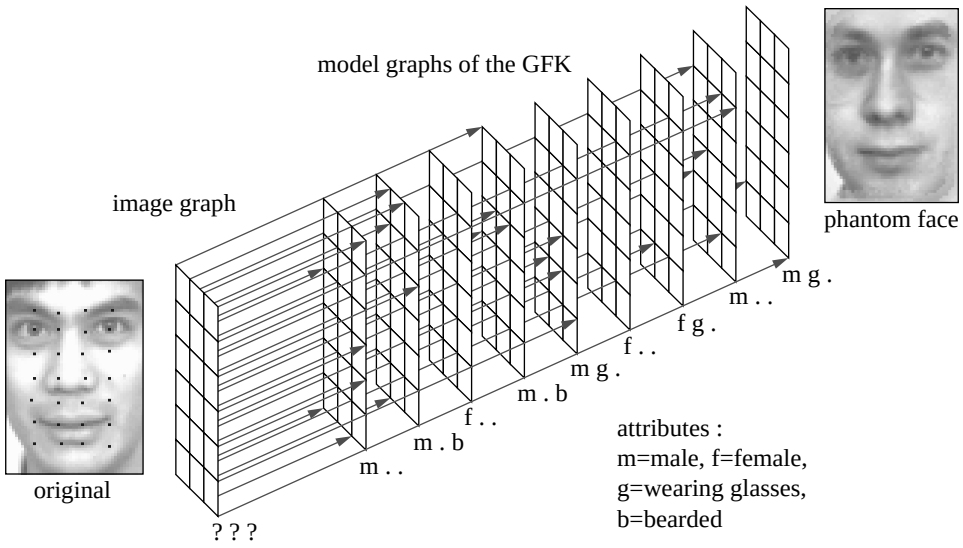

Figure 6.1: The stack structure of the General Face Knowledge. We see here how the individual nodes of an image graph best fit different model graphs. Each model graph is labeled with known attributes, based on which the attributes of the probe face can be determined. On the right a phantom face is shown.

---

best fitting jets in the GFK. These best fitting jets are called *local experts*. In order to generate a *phantom face*, respective patches of grey values from the models at the *fiducial points* at which they fit best are joined together with smooth transitions. The precise positions of the patches are given by the pixel positions of the image graph as yielded by the matching process. The result is shown on the right. More examples are shown in Figure 6.2.

Notice that since only faces of the same pose are compared there is no need for object-adapted graphs in the sense of the preceding chapter. However, the regular graphs used here are aligned with each other as in Chapter 4, with certain nodes lying on the eyes and certain others on the line where the two lips meet. The other nodes are positioned by the regular structure of the grids. However, I will refer to the node positions as fiducial points.

Notice that no grey-value information of the original image is used to generate a phantom face. Only the matching information (the local experts and their locations) and the model images are used, and a phantom face is typically composed of ten to twenty different models. The first thing that strikes one is that they look so natural in spite of actually being a patch-work. They can also look very similar to the original. However, only what is represented in the GFK can be reconstructed. For example there was no Asian face in the GFK when generating the phantom face in Figure 6.1. Hence one gets a Caucasian phantom face, wich is otherwise very similar. It is obvious that the quality of phantom faces improves with the size of the GFK.

6.2.2 Determining Facial Attributes

In the previous section it was demonstrated that a lot of information about a probe face can be represented by the local experts in the GFK. I will now demonstrate that not only the face image can be reconstructed, but also *facial attributes* such as gender, the

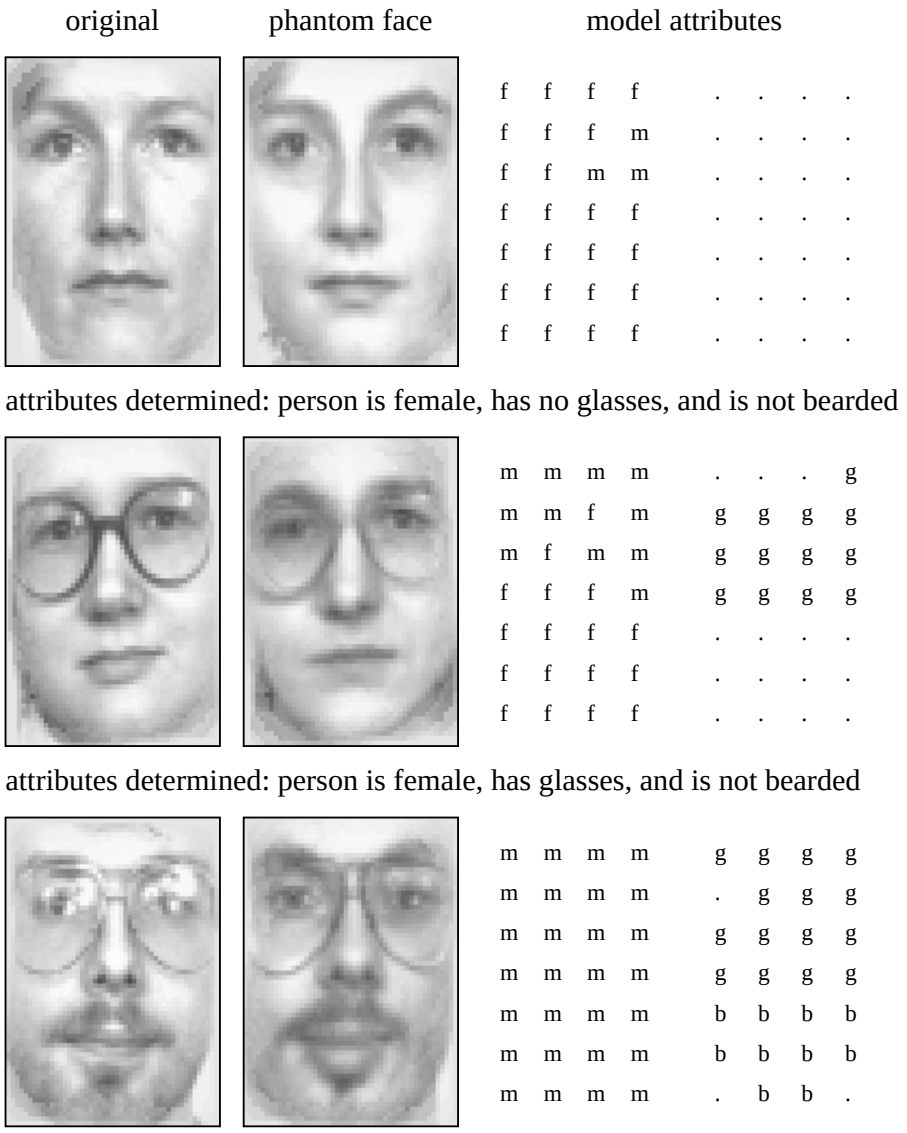

Figure 6.2: Shown here is the original and the phantom face for three different persons. Notice that the phantom image was generated only on the basis of information provided by the match with the General Face Knowledge; no image information from the original was used. That is why certain details, such as the reflections in the glasses or the precise shape of the lips of the top image are not reproduced accurately. The fields of labels on the right side indicate the attributes of the models which provide the local experts for the individual nodes; m: male, f: female, b: bearded, g: glasses.

presence of a beard, and wearing glasses can be inferred from the match result in a very simple way.

Let us assume that gender and the presence of a beard or glasses is known for the models in the GFK. Since the phantom face looks so similar to the original, it is reasonable to assume that the labels of the models providing the local experts correspond to the attributes of the probe face with some reliability as well. The nodes of a female will most often fit female models, and a bearded man will pick up bearded models in the lower half of nodes. This principle is demonstrated in Figure 6.1. Figure 6.2 shows examples of actual experiments. In order to decide whether a probe face is male or female one simply has to count whether more local experts belong to male or female models. Similarly for beard and glasses, considering, however, only the lower or upper half of the nodes, respectively.

This is an illustration of the principle idea. In practice one would like to have a more thorough analysis of the node label distributions, especially concerning the question of which of the nodes are reliable and which are not. I am therefore now going to apply a more systematic statistical analysis.

6.2.3 Statistical Analysis

In order to perform a statistical analysis, I consider the process of a node in the image graph pointing to a model with a particular attribute as a probabilistic event. For each node n I introduce the stochastic variable X_n, which can assume the values 1 and 0 depending on whether the respective local expert has a particular attribute or not. X is the corresponding random variable for the probe face. A sample of these stochastic variables is denoted by x_n and x, respectively. Given an image with a certain value x of X, one can ask for the conditional probability $P(x_1, ..., x_N|x)$ of a particular combination of node labels. I make the strong assumption that the conditional probabilities for the individual nodes are independent of each other: $P(x_1, ..., x_N|x) = \prod_n P(x_n|x)$. The *Bayes a posteriori probability* for a probe face having the attribute x given the node labels x_n then is

$$\begin{aligned} P(x|x_1, ..., x_N) &= \frac{P(x_1, ..., x_N|x)P(x)}{P(x_1, ..., x_N|1)P(1) + P(x_1, ..., x_N|0)P(0)} \\ &= \frac{P(x)\prod_n P(x_n|x)}{P(1)\prod_n P(x_n|1) + P(0)\prod_n P(x_n|0)}\,. \end{aligned} \tag{6.1}$$

The decision whether the attribute is present ($x = 1$) or not ($x = 0$), is based on whether $P(1|x_1, ..., x_N) > P(0|x_1, ..., x_N)$ or not.

The probabilities $P(x_n|x)$ are not known and have to be estimated on the basis of relative frequencies $F(x_n|x)$ evaluated on a training set of images for which the attributes are known. Assume that there are N images in the training set, of which $N(x)$ images have value x for X, with $x \in \{1, 0\}$. $N(x_n|x)$ of them are labeled with value $x_n \in \{1_n, 0_n\}$ at node n. For example for a training set of 21 faces one could get

$$\begin{aligned} N &= N(1) + N(0) = 16 + 5 = 21, \\ N(1) &= N(1_n|1) + N(0_n|1) = 12 + 4 = 16, \\ N(0) &= N(0_n|0) + N(1_n|0) = 5 + 0 = 5. \end{aligned}$$

The derived relative frequencies are

$$\begin{aligned}
F(1) &= N(1) /N &= 0.76,\\
F(1_n|1) &= N(1_n|1)/N(1) &= 0.75,\\
F(0_n|1) &= N(0_n|1)/N(1) &= 0.25,\\
F(0) &= N(0) /N &= 0.24,\\
F(0_n|0) &= N(0_n|0)/N(0) &= 1.00,\\
F(1_n|0) &= N(1_n|0)/N(0) &= 0.00.
\end{aligned}$$

If one used these relative frequencies as probabilities for evaluating the *a posteriori* probability, a probe face with label 1 at node n would have a vanishing probability of having attribute 0. That is a too strong statement on the basis of such a small sample set. I therefore enforce the probabilities being greater than zero by incrementing the zero class and decrementing the corresponding one class by one. Hence $N(0_n|0)$ and $N(1_n|0)$ would be corrected to 4 and 1 respectively. The relative frequencies $F(0_n|0)$ and $F(1_n|0)$ would become 0.8 and 0.2 respectively. Now the relative frequencies can be taken as probabilities for Bayes' formula.

Another issue is how to estimate $P(1)$ and $P(0)$. One can take the relative frequencies of the training set. But that might be too strong a prejudice, producing good results for the majority class but relatively poor results for the minority class. In addition, this would implicitly take into account knowledge about the composition of the test set, since training set as well as test set are drawn from the same complete set, therefore having approximately the same fraction of females, etc. For these reasons I consistently chose $P(1) = P(0) = 0.5$ to avoid prejudicing the test set composition.

6.2.4 Equivalence between Bayes' and Weights Formulation

An alternative to the Bayes approach would be to train *weights* for each *node* in order to optimize the correct determination rates. The decision would then be made on the basis of a weighted sum over all nodes with a certain attribute. Since weights provide an easy interpretation and a good visualization, I now transform Bayes' formula into an equivalent weight formulation.

As seen above, the Bayes method determines a attribute based on whether $P(1|x_1, ..., x_N) > P(0|x_1, ..., x_N)$ or not. By means of Equation 6.1 and taking into account that x_n may only assume the values 0 and 1, this can be transformed in the following way:

$$\begin{aligned}
& & P(1|x_1, ..., x_N) &> P(0|x_1, ..., x_N)\\
&\iff & P(1)\prod_n P(x_n|1) &> P(0)\prod_n P(x_n|0)\\
&\iff & \sum_n \ln\left(\frac{P(x_n|1)}{P(x_n|0)}\right) &> \ln\left(\frac{P(0)}{P(1)}\right)\\
\iff & \sum_n x_n\left(\ln\left(\frac{P(1_n|1)}{P(1_n|0)}\right) - \ln\left(\frac{P(0_n|1)}{P(0_n|0)}\right)\right) &> \ln\left(\frac{P(0)}{P(1)}\right) - \sum_n \ln\left(\frac{P(0_n|1)}{P(0_n|0)}\right)\\
&\iff & \sum_n x_n \ln\left(\frac{P(1_n|1)P(0_n|0)}{P(1_n|0)P(0_n|1)}\right) &> \ln\left(\frac{P(0)}{P(1)}\right) - \sum_n \ln\left(\frac{P(0_n|1)}{P(0_n|0)}\right)\\
& & \iff \sum_n x_n\beta_n &> \theta, \qquad (6.2)
\end{aligned}$$

male female total	bearded		beardless		total	
	#	%	#	%	#	%
glasses	9	8.1	18	16.2	27	24.3
	0	0.0	4	3.6	4	3.6
	9	8.1	22	19.8	31	27.9
no glasses	12	10.8	33	29.7	45	40.5
	0	0.0	35	31.5	35	31.5
	12	10.8	68	61.3	80	72.1
total	21	18.9	51	45.9	72	64.9
	0	0.0	39	35.1	39	35.1
	21	18.9	90	81.1	111	100.0

Table 6.1: Composition of the General Face Knowledge.

with

$$\beta_n = \ln\left(\frac{P(1_n|1)P(0_n|0)}{P(1_n|0)P(0_n|1)}\right), \tag{6.3}$$

$$\theta = \ln\left(\frac{P(0)}{P(1)}\right) - \sum_n \ln\left(\frac{P(0_n|1)}{P(0_n|0)}\right). \tag{6.4}$$

The weights β_n are shown in Figure 6.3 as black circles with a diameter proportional to the weights. It is obvious that the bottom rows are *significant* for beard detection and that the top rows are significant for glasses detection. For gender, the weights show no strong emphasis on a particular region. The weights are not perfectly symmetrical with respect to the vertical axis, and there are some negative weights. This is probably due to the fact that the galleries were not large enough, especially for the pure sets.

6.3 Experiments

6.3.1 Database

The *gallery of faces* used here was set up at the *Institut für Neuroinformatik*, Bochum, and contains 111 neutral frontal views. The images had 128×128 pixels subsampled from 512×512 pixels with 256 grey levels. The *size* of the faces varied up to a factor of 1.5, with a tendency for male faces to be larger than female faces. I therefore rescaled all images such that the x- and y-spacing is 10 pixels on average; the ratio between x- and y-spacing was kept as in the original image. In order to avoid a bias of the gender determination due to gender specific hairstyles, the outer regions were masked by a grey frame with a smooth transition to the face, see fig 6.4. The composition of the gallery with respect to the attributes male, beard, and glasses is shown in Table 6.1.

6.3.2 Results

Correct attribute determination rates are given in Table 6.2. The complete GFK contains 111 faces, which also serve as probe faces. Hence, if a face is analyzed it is excluded from

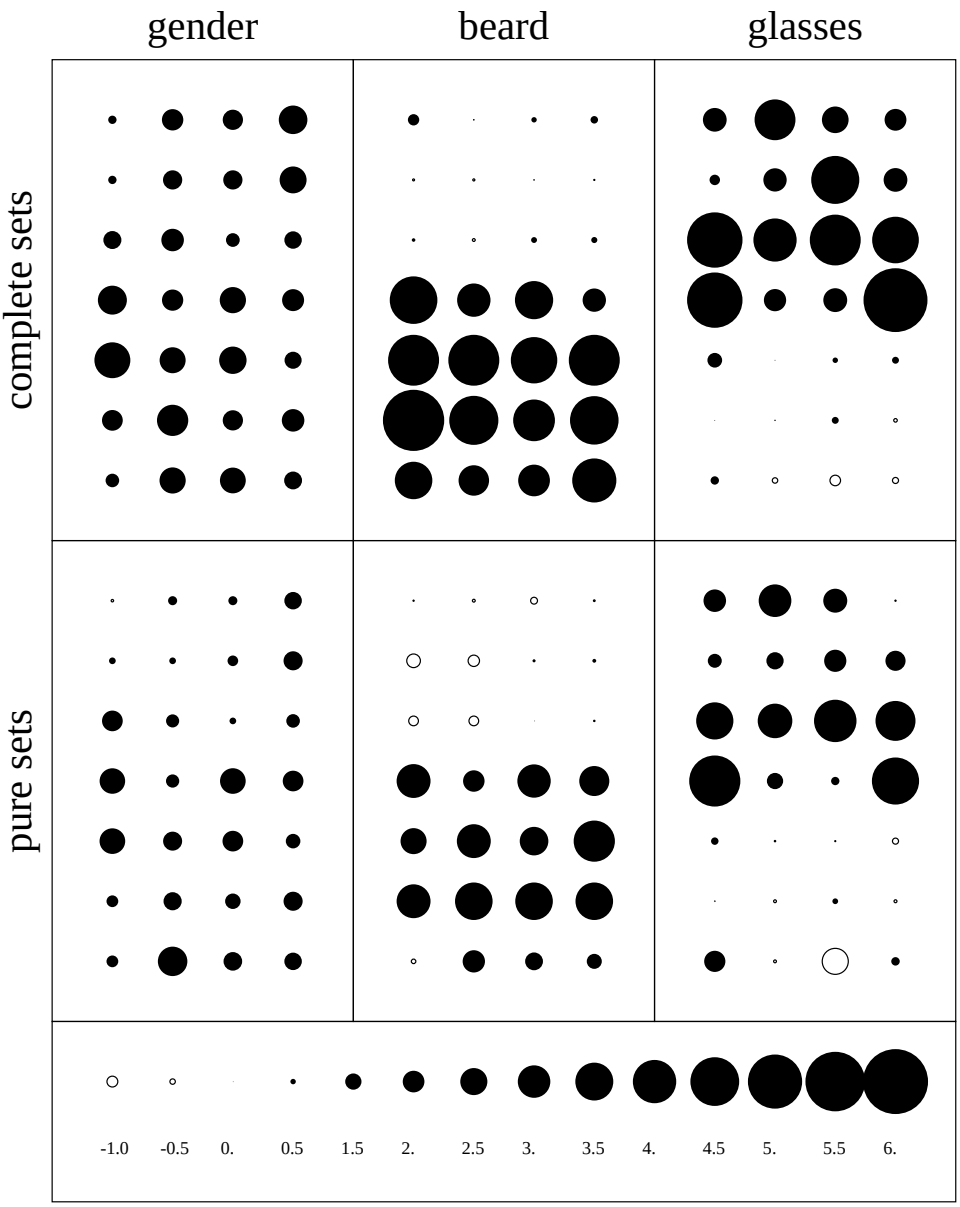

Figure 6.3: Weights β_n of the nodes. From left to right for gender determination, beard detection, and glasses detection. The weights in the top row are determined on all 111 test images and the complete GFK of 111 minus 1 models. Results on pure sets are shown in the bottom row. From left to right on beardless faces without glasses only, on males without glasses only, and on beardless males only.

	male female total	bearded beardless total	glasses no glasses total
complete	0.917 ± 0.049 (0.944)	0.839 ± 0.133 (0.905)	0.895 ± 0.067 (0.903)
sets	0.938 ± 0.058 (0.949)	0.964 ± 0.024 (0.956)	0.990 ± 0.015 (0.988)
(111/111/111)	0.924 ± 0.033 (0.946)	0.941 ± 0.027 (0.946)	0.963 ± 0.023 (0.964)
small	0.851 ± 0.098	0.848 ± 0.148	0.834 ± 0.128
sets	0.899 ± 0.070	0.964 ± 0.043	0.965 ± 0.050
(68/45/51)	0.875 ± 0.050	0.935 ± 0.054	0.919 ± 0.053
pure	0.822 ± 0.091 (0.818)	0.589 ± 0.198 (0.833)	0.800 ± 0.113 (0.778)
sets	0.840 ± 0.090 (0.857)	0.942 ± 0.046 (0.970)	0.932 ± 0.044 (0.970)
(68/45/51)	0.831 ± 0.050 (0.838)	0.857 ± 0.052 (0.933)	0.885 ± 0.043 (0.902)

Table 6.2: Correct attribute determination rates. In the first row the complete GFK of 111 faces was used for all three attributes. The images were split into a training set and a test set of 55 and 56 faces respectively. On the training set the probabilities $P(x_n|x)$ were estimated; on the test set the performance of the trained system was evaluated. The standard deviation is shown as well. In brackets the performance is given for the case where training set and test set are identical and both contain all 111 samples of the GFK. This gives an estimation for the upper bound of performance that can be obtained on this gallery with the Bayes approach. The last row gives performance results for pure sets, i.e. 68 unbearded faces without glasses for gender determination, 45 male faces without glasses for beard detection, and 51 unbearded males for glasses detection. The results degrade significantly. Part of the degradation is due to the decreased GFK size. For comparison, results are given on mixed sets of same size in the middle row.

---

the model gallery, and only 110 samples remain for the GFK. The same holds for smaller GFKs, e.g. in case of pure sets. The probe faces are usually split into a training and a test set of equal size. On the training set the relative probabilities for each node were estimated, and on the test set the correct determination performance was tested. In order to get a reliable mean perfomance and a standard deviation, 100 different training and test sets were drawn from the complete set randomly. For the results given in brackets, the training and test set were identical and of maximum size, i.e. of same size as the GFK.

Along with the results for the complete set of 111 faces, results on pure subsets are given. For gender, only unbearded faces without glasses were used, for beard only male faces without glasses, and for glasses only unbearded males, yielding GFKs of 68, 45, and 51 faces respectively. This test was mainly done to check to what extent the different attributes interfere with each other. A certain degradation can be expected from the reduced number of faces in the GFK, shown in the middle row (see also next section). Even taking this into account all results degrade. This is probably due to correlations between the different attributes. It is clear that the presence of a beard tells something about the gender. Using pure sets makes the task more difficult. The correlation coefficients are 0.356 for the attributes male and beard, 0.290 for the attributes male and glasses, and 0.161 for the attributes beard and glasses, as can be computed from Table 6.1.

Figure 6.4 shows more of the 111 sample faces. They are ordered with respect to the significance of their attributes as judged by the system when all faces were used as

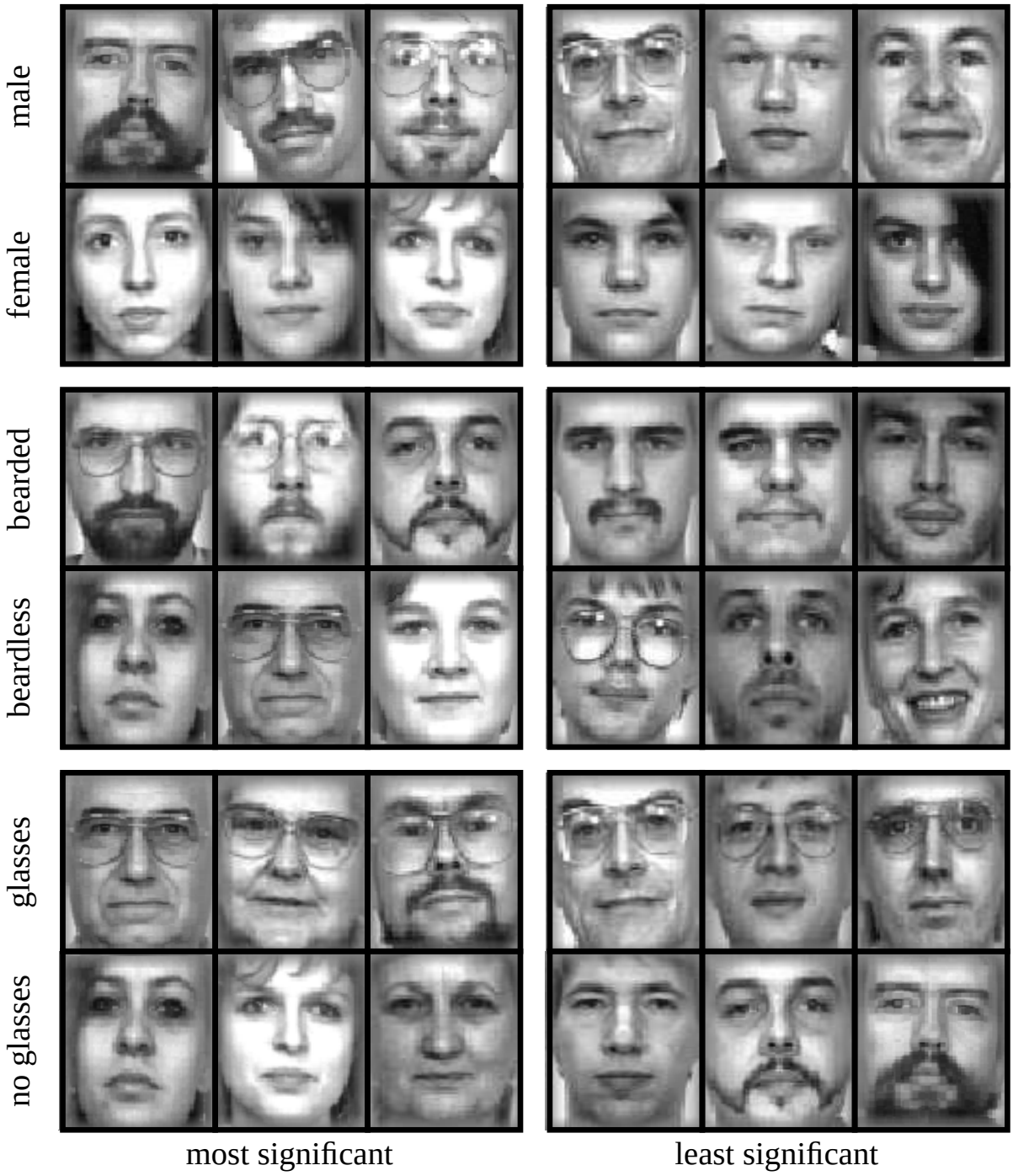

Figure 6.4: Sample faces ordered with respect to their significance for the different attributes. Most significant faces are on the left, least significant faces are on the right. From top to bottom faces are shown that are male, female, bearded, unbearded, with glasses and without glasses. All faces are shown in the same format for display purposes.

training set. The least significant females include the two youngest ones in the gallery, who were of an age where the difference between male and female faces is less obvious than for older persons. The least significant beardless face, the female, was probably misclassified due to the smile, which generated structures resembling a beard. The least significant bearded and beardless samples also reveal the difficulties and some arbitrariness in the definition of who is bearded and who is not. The samples in the bottom two rows, glasses and no glasses, allow no conclusions about the reasons why certain faces are misclassified with respect to this attribute.

6.3.3 Dependencies on Parameters

The purpose of this section is to investigate the dependencies of the correct classification rates on the parameters of the system. The matching process itself was not varied, i.e. the image graphs were generated once, and even if the size of the GFK was varied the image graphs were kept constant as obtained with the maximum size of the GFK.

First I am going to consider the system for different sizes of the training set. The smaller the training set, the greater the errors in estimating the probabilities $P(x_n|x)$. But as the left graph in Figure 6.5 shows, the typical training set size of 55 faces is sufficient to get maximum performance.

I claimed previously that increasing the size of the GFK improves the classification rate. In order to get an impression, I measured the performance with varying GFK size. The GFK always contained at least one model for each attribute. The right graph in Figure 6.5 shows how performance increases with GFK size and that it has not yet reached its maximum. Especially for gender, one can expect that correct determination rates will improve significantly with a larger GFK size. It is surprising that in case of beard detection, four models already achieve a performance of 0.85 (though matching precision would degrade significantly with only four models in the GFK). It is clear that the required size of the GFK depends not only on the performance level that one wants to achieve but also on the number of attributes that one wants to determine and on the variety of the faces.

One idea to improve the performance is to take not only the best fitting jet per node into account, but to consider the second best, third best, etc., as well. The left graph in Figure 6.6 shows the performance depending on the rank of the jets used for classification, i.e. only one jet per node was used: the best, the second best, the third best, etc. As expected, the performance degrades slowly. But one might still expect performance to increase if one takes the first several best into account. The Bayesian approach was applied to each of the first ranks at each node, providing $N \times R$ conditional probabilities per attribute, if R is the number of ranks taken into account. Results are shown in the right graph. The result is not very successful. Only in case of gender determination could a significant improvement be achieved. For glasses the performance in fact decreased. A reason might be that the node labels on different ranks are not independent of each other as assumed in the Bayesian approach.

Finally it is worth mentioning that the *phase information* is crucial for selecting the correct local experts. For identical image graphs, performance degrades significantly if the similarity function $\mathcal{S}_a$ is used istead of $\mathcal{S}_\phi$ (see Equations A.6 and A.7).

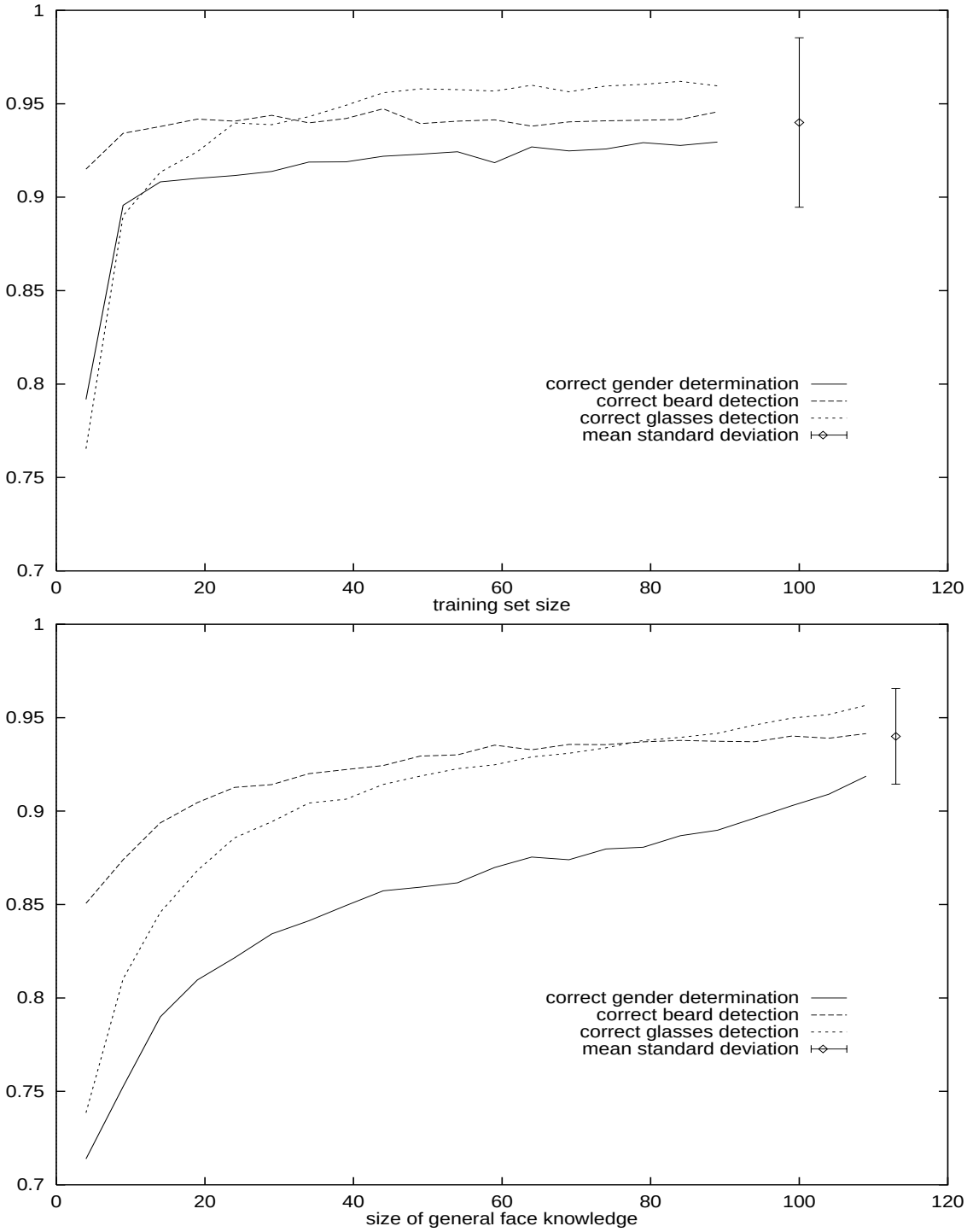

Figure 6.5: These graphs show the dependence of the mean rates of correct attribute determination depending on training set size (top graph) and GFK size (bottom graph). For the top graph, test set size was constantly 20 samples and the GFK size was 110, while the training set size varied from 4 to 89. For the bottom graph, training and test set had their standard size of 55 and 56 respectively, while the GFK size varied from 4 to 109. For each data point 500 different samples of the training and test set were chosen randomly to get a reliable mean performance.

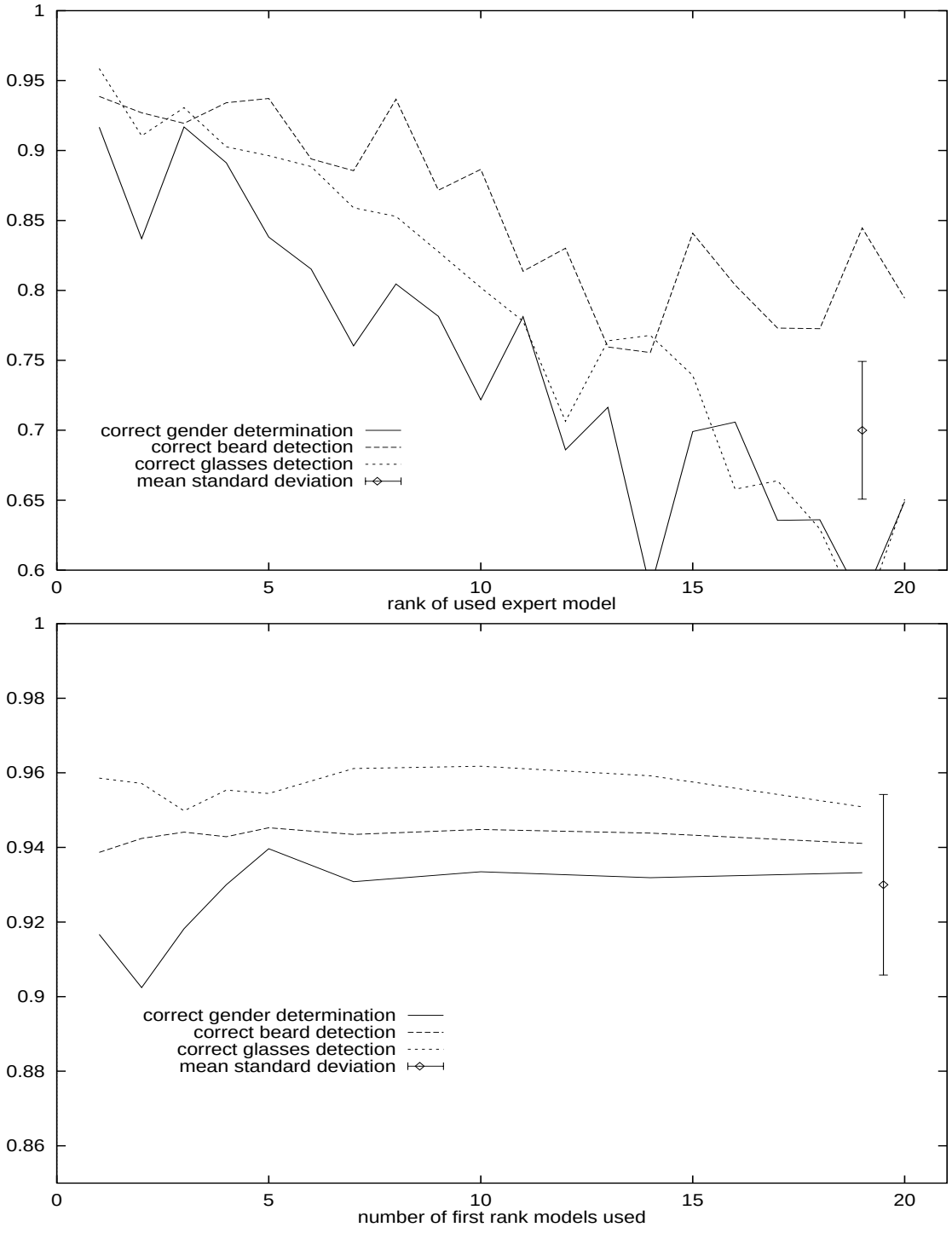

Figure 6.6: Correct attribute determination rate dependent on the rank of the used local experts (top graph) or the number of first rank local experts used (bottom graph) if several jets were evaluated per node. Although the jets of lower rank have still some reliability in determining the attributes, taking into account several jets does not improve performance significantly.

6.4 Discussion

The system presented here demonstrates successful determination of the facial attributes gender, beard, and glasses. The architecture is quite homogeneous and simple. As in the face recognition system of the previous chapter, nothing is specialized to faces, and one may expect the system to perform satisfactorily on similar tasks, such as discriminating between species of domestic animals.

6.4.1 Comparison with Other Systems

While there is a huge literature on face recognition, there are relatively few publications about artificial systems for gender determination. GOLOMB et al. (1991) employed a standard *back-propagation* network for gender determination. In 90 images of faces (45 beardless male, 45 female) the eyes were located manually and the images then rotated and scaled automatically to a standard format of 30×30 pixels. Images were compressed by an encoder back-propagation network with 40 hidden units. The output of these 40 units served as input for a gender determination network, the *SexNet*, trained with the back-propagation algorithm as well. 8 tests were performed with a training set of 80 images and 10 test images. Mean performance and standard deviation were 91.9%±8.6%. The system used limited hair information.

O'TOOLE et al. (1993) used *Principal Component Analysis* for face representation and for the discrimination of ethnic groups as well as gender. They give no performance results on gender determination.

A system based on geometrical features was presented by BRUNELLI & POGGIO (1993a). They used 168 images of 21 males and 21 females. The faces were automatically normalized with respect to rotation and scaling. Then 18 different geometrical features such as pupil-to-nose vertical distance, nose width, chin radii, and eyebrow thickness were automatically extracted, providing one 18 dimensional vector per image. No hair information was used. A Hyper Basis Function Network was trained on the data sets of all minus one person and tested on the excluded ones. The mean performance on the training sets was 92% and on the test sets 87.5%.

Though the performance of these systems is comparable or higher, the system presented has several advantages. Firstly, it is very general and conceptually not restricted to faces as is the system of BRUNELLI & POGGIO. No face specific features have to be defined. Secondly, it is local, i.e. in contrast to the PCA approach of O'TOOLE et al. localized attributes such as glasses can easily be determined and one gets information about which regions are important. Thirdly, the system presented is fully automatic: no manual alignment such as in the system of GOLOMB et al. is required. Fourthly, the *training effort* is minimal: only few training samples are required to determine the relative weights of the nodes. PCA and back-propagation are known to be expensive in terms of training. The main drawback of the system presented is that it is slow in respect to processing time. The Gabor transformation and the Elastic Graph Matching require about one minute on a Sun SPARCstation 10–512 with a 50 MHz processor. As a second disadvantage one may consider that the system presented does not reveal what is characteristic for a certain attribute (though the node weights indicate which nodes are significant). BRUNELLI & POGGIO for example could illustrate that their system considers a face as male if it has thick eyebrows, a short and wide nose, a long distance

between mouth and nose, etc. O'Toole et al. found in their system that the second eigenvector explains most of the variance for gender determination. Image eigenvectors can be directly visualized. Such information is not easily available in the system presented here.

6.4.2 Future Perspectives

So far the attribute labels of the GFK-models are binary and defined by hand. A male face can therefore be misclassified because it looks actually female or because there is a similar female face in the GFK that is very male in appearance. The attribute labels should vary contiuously from one extreme, e.g. male, to the other, female. Taking this into account might improve the determination performance. It might also be possible to let the GFK find reasonable attribute classes autonomously.

Another direction of investigation would be to apply the system to other facial attributes and to use the results for face recognition purposes. The set of possible candidates in the model gallery reduces significantly if several attributes, such as gender, age, and ethnic group are determined in advance. Another idea could be to use the phantom face representation for manipulations such as rotation in depth, or generating a different facial expression. Assume a GFK of neutral frontal views and a GFK of the same persons in a different pose are given. A single neutral frontal view of a probe face could be presented to the system, and a phantom face could be generated. Since for all models in the GFK a rotated version is present, it should be possible to generate a rotated phantom face. The question is whether the rotated phantom face would look similar to the rotated original or not. Preliminary experiments showed no significant similarity and further investigation is necessary.

It would also be interesting to apply the system to another class of objects such as domestic mammals, distinguishing between dogs, cats, sheep, and horses of different races. The task then is to abstract from several horses of different race what is typical for horses and to abstract from several dogs of different race what is typical of dogs. This would again be done by generating general knowledge about domestic mammals, all having a common graph structure. By finding the local experts for each node, the type of animal could be determined. Animals are actually a good example, since they show a complex hierarchy. It would be interesting to build an animal classification and recognition system based on the system presented here. An animal would be classified according to its phenotype. For example starting from distinguishing between birds and mammals, then differentiating between more similar species such as dogs and horses, then classifying according to the specific race, e.g. poodle versus German shepherd, and finally recognizing the individual animal, if possible.

Chapter 7

Recognizing Objects in Cluttered Scenes

Abstract: The system presented below uses Elastic Graph Matching (EGM) to match stored models into a scene of objects partially occluding each other. The similarities of the nodes with the image are evaluated in order to decide which regions are occluded and which are visible. If all objects in the scene are known, the system can process the scene from front to back. It can then take advantage of the fact that the occluding objects are already recognized and that their contours are known.

7.1 Introduction

The systems presented in the previous three chapters all perform face recognition, a task specific in the sense that faces form a class of similar objects, and the General Face Knowledge accounts explicitly for that. In this chapter I am going to present a recognition system for objects of very different character and shape[1]. Toy objects are arranged into scenes and may occlude each other significantly. They are represented by labeled graphs, and Elastic Graph Matching (EGM) serves to find the objects in the scene. The difficulty is to decide which objects are actually present in the scene and to determine their order in depth. For this task the graph structure is especially advantageous, since it allows reference to overlap regions in the image and makes explicit that some parts are occluded while others are not.

Two different algorithms will be presented. The first is appropriate for searching a known object in a scene with other objects which are unknown. The algorithm decides whether the object is present, where it is, and which parts are visible and which are occluded. The second algorithm requires that all objects in the scene are known to the system. The algorithm then analyzes the scene from front to back, taking advantage of the fact that the objects in the front are completely visible and that for the objects behind them it is already known which regions are occluded. This algorithm decides which objects are present in the scene, where they are, and in which order in depth they are arranged. It also provides information as to which regions of the models are occluded and which are visible.

[1]This chapter is in part a modified reprint of (Wiskott & von der Malsburg 1993)

7.2 The System

7.2.1 Data Structures

The total system is composed of an *image domain* $\mathcal{I}$ and a *model domain* $\mathcal{M}$. *Objects* are represented by *labeled graphs* having a square grid structure with an outline dependent on the contour of the objects (see Figure 7.2). Nodes are labeled with *jets* $\mathcal{J}$ as local features (see Appendix A). Edges which connect neighboring nodes are labeled with the difference vectors between the respective node positions $\vec{x}_n$ in the image measured in pixel units. The graphs are *rigid*, i.e. in this system I do not allow for distortions of the graphs. The similarity between the jets is defined as in Equation A.6 with the difference that the similarity is taken to the power of four to emphasize nodes with high similarity (this to be motivated later):

$$\mathcal{S}(\mathcal{J}(\vec{x}_n), \mathcal{J}'(\vec{x}_n)) = \mathcal{S}_a^4(\mathcal{J}(\vec{x}_n), \mathcal{J}'(\vec{x}_n)) \ . \tag{7.1}$$

To specify the state of the *scene analysis* system completely, it is necessary in addition to represent which regions of the image have been recognized by which model graphs, and what the *occlusion relations* between the objects are. I describe the relation of the model domain to the image domain with the help of a few binary variables that decide on the recognition status of a model and the *visibility* or *occlusion* of its individual *nodes*, plus a single position vector for the placement of the model graph in the image. These variables will be introduced in the next section.

7.2.2 Model Graph Formation

For the formation of a *model graph*, a simple *segmentation* procedure is applied. Three images are taken, each with the object in identical position on a different background. In two of the images, the background is formed by a horizontal or vertical lattice of black and white stripes approximately 3.5 pixels wide. The third image has a white background. A square lattice of points with a spacing of 7 (or, for some delicate objects, 5) pixels is selected in the image. For each of the selected lattice points $\vec{x}_n$ the similarity $\mathcal{S}(\mathcal{J}^h(\vec{x}_n), \mathcal{J}^v(\vec{x}_n))$ between jets taken from the images with horizontal and vertical stripes, respectively, is computed. These similarity values are high within the area covered by the object and low over the background. (Similarities for nodes inside an object but near its border are lower to the extent that their wavelets reach over into the background.) Now all lattice points are selected for which this similarity is above a threshold (which I chose in the range 0.12^4–0.33^4, depending on the object. A more sound treatment of occluding boundaries will have to supplant this ad hoc treatment later). The procedure may produce several mutually disconnected graphs, the largest of which is selected. (Two graphs are connected if their minimal distance is one lattice spacing.) For the graph thus obtained, lattice points are labeled with the jets taken from the third image, which has been taken with a blank background. The resulting graph is stored as a model graph in memory. This *graph formation* process has the advantage of positioning the nodes *automatically* in those regions of the object which are least sensitive to background variations. For example, the process avoids placing nodes in openwork regions of an object. I also successfully experimented with other background textures, not only horizontal and vertical stripes.

7.2.3 Matching a Model into a Scene

When *matching* a model graph against the image of a scene, a replica of its set of node positions $\{\vec{x}_n^{\mathcal{M}}\}$ is placed in the image, creating the set of points $\{\vec{x}_n^{\mathcal{I}}\}$, where $\vec{x}_n^{\mathcal{I}} = \vec{x}_n^{M} + \vec{x}^s$ with an offset vector $\vec{x}^s$ common to all nodes. The graph formed by the set of points $\{\vec{x}_n^{\mathcal{I}}\}$ is called the *image graph*. The model graph is compared to the image graph with offset $\vec{x}^s$ in terms of the *similarity* function

$$\mathcal{S}_{\mathcal{G}}(\mathcal{G}^{\mathcal{I}}, \mathcal{G}^{\mathcal{M}}) = \frac{1}{N_V} \sum_{n \in V} \mathcal{S}(\mathcal{J}_n^{\mathcal{I}}, \mathcal{J}_n^{\mathcal{M}}), \tag{7.2}$$

where $\mathcal{J}_n^{\mathcal{I}}$ is the image jet taken at position $\vec{x}_n^{\mathcal{I}}$, V is the set of *visible nodes* (visibility being defined below), and N_V is their number. Now the matching similarity (7.2) is maximized by varying $\vec{x}^s$. First, the offset is taken through all points on a square grid with a spacing of five pixels for which the replica of the model graph lies entirely within the image. Around the lattice point with maximal similarity a better maximum is then found for a finer lattice with spacing 1. The resulting image graph is taken as the candidate match. In distinction to the matching schedule described in Chapter 5, here I do not consider distortions of the image graph with respect to the model graph, and I use no phase information for the matching.

When an object is *occluded* to a large extent, the total *similarity* of its match is degraded by the many nodes that come to fall on the image of other objects and that correspondingly have only average similarity values. It is decisive that this correct match cannot be outdone by a false match in some region of the image picked such that many nodes have above-average similarity. In order to favor the correct match, I give its correctly matching nodes an advantage over the only averagely fitting nodes by raising the inner product in Equation A.6 to the fourth power. (In the experiments, evaluation of correctly analyzed scenes shows that correctly matched nodes have a mean similarity value of $\mathcal{S}_c = 0.64$, whereas with graphs matched to scenes not containing the object nodes have a mean similarity of $\mathcal{S}_w = 0.35$. Random pairs of jets have a mean similarity of $\mathcal{S}_r = 0.32$.)

7.2.4 Scene Analysis, Algorithm One

In this first simple *scene analysis* algorithm, each model graph is matched separately to the image to decide if and where it fits and to what extent it is occluded. The algorithm has the advantage that there is no need for all objects in the scene to be known to the system. The algorithm examines all graphs in the model domain. First, a graph is matched to the image. Then all nodes under a threshold of 0.52 for $\mathcal{S}$ are marked as occluded. (This parameter could be obtained automatically by collecting a similarity histogram for a large set of model jets and image jets. In my experience this histogram tends to be bimodal, and the threshold can be set near the minimum between the modes.) Since I assume that occlusion occurs for *coherent regions*, the algorithm proceeds to revise the occlusion decision for each node according to its neighborhood in the graph. For occluded nodes which have a majority of visible neighbors within the model graph, the decision is reversed, and similarly for visible nodes which have a majority of occluded neighbors. In this way all nodes are visited repeatedly, in the

arbitrary sequence inherent in the graph administration system, until no further changes occur.

The *visible region* of the model's image graph is then the set of all pixels that lie within squares of size d centered around visible nodes of the graph (d being the spacing of nodes in the model's graph, 7, or sometimes 5, pixels). If it has an average node similarity better than 0.61 and an area of at least 1300 pixels, the model is accepted for the scene. (Although not the point of this algorithm, it is convenient for display purposes to order the accepted models in depth according their mutual occlusion indices, which are computed as explained in the next section.)

7.2.5 Scene Analysis, Algorithm Two

For this algorithm to work, there must be models for all objects in the scene. Posing such a constraint has the advantage that the relative *occlusion relations* can be determined and used for a more reliable analysis of the scene. For two graphs A and B, this relation will be characterized by the *occlusion index* $\mathcal{Q}_{AB}$. When it is computed, the system may already have decided that third object(s) are occluding parts of A or B so that only part of their graphs are visible in the image. Let me define a similarity function $\mathcal{S}_A(\vec{x})$ for a model A for all pixels of the image after the model has been matched. For each pixel $\vec{x}$ of the image it gives the similarity value of the nearest node, and zero outside the visible region of the graph. Further, let R be the region of overlap between the visible parts of A and another model B. Then the occlusion index of A with respect to B is defined as

$$\mathcal{Q}_{AB} = \sum_{\vec{x} \in R} \mathcal{S}_A(\vec{x}) - \mathcal{S}_B(\vec{x}). \tag{7.3}$$

For this we have the relations $\mathcal{Q}_{AB} = -\mathcal{Q}_{BA}$, $\mathcal{Q}_{AA} = 0$, and for graphs A and B without overlap we have $\mathcal{Q}_{AB} = 0$. If $\mathcal{Q}_{AB} > 0$, A is occluding B, and if $\mathcal{Q}_{AB} \geq 0$, A is said not to be occluded by B.

To start scene analysis, all stored models are first matched to the image. Each model will yield a "match", that is, the graph placement fully inside the image with maximal graph similarity (see Equation 7.2). These graph placements will not be changed during all of the following steps. The effect of the following iterative process will result in the graduation of some of the models in two steps, first to the status of candidate, then to the status of being accepted. The process has the following steps:

Step 1 Turn those models into candidates (i) for which the average node similarity is better than 0.45, (ii) for which the visible region is bigger than 1300 pixels and (iii) which haven't yet been accepted.

Step 2 Stop if there are no candidates.

Step 3 Accept one of the candidates: First, determine the mutual occlusion indices for all pairs of candidates. Then select the "unoccluded" candidate(s), that is, those for which all occlusion indices are non-negative. If there are several, select the one with highest graph similarity (see Equation 7.2). If there is no candidate for which all occlusion indices are non-negative, pick the one that is least occluded, that is, for which the smallest occlusion index is largest.

Step 4 Insert the model just accepted in the depth sequence of all accepted models. For this, the new model has to work its way from the back of the list forward. The model is advanced one step if its occlusion index relative to the model in front of it is non-negative. (This comparison is based on the overlap of all of the new model with the visible part of the model in front of it.) Advancement stops as soon as the new model hits one with which it has a negative occlusion index.

Step 5 Now the occlusion status of the nodes is updated. Those in the newly accepted graph are occluded by the territory of the models in front of it. The nodes of the model graphs now put behind are occluded by the new one. Also, the territory of the newly accepted graph is declared invisible for all as yet uncommitted model graphs, whose visible areas and graph similarity values are modified correspondingly (see Equation 7.2). After that, proceed with Step 1.

7.3 Experiments

7.3.1 Database

The pictures of the scenes are of size 128×128 pixels and have 256 distinguishable gray levels. They are derived from 512×512 pixel frames taken with a CCD camera by low-pass filtering and subsampling. I took pictures of 30 scenes composed of 3–6 objects each. Scenes were taken at approximately 200 cm of distance, at which pictures are 42 cm wide. Distances to individual objects vary up to 10 cm, causing size variance of up to 5% relative to the images used for model graph formation. Individual objects were mostly presented in approximately the same orientation and perspective, although I also did individual experiments with slightly rotated objects (see Figure 7.2.b). To avoid visible shadows, I sometimes illuminated scenes with two light sources (although during model graph formation there had been only one light source). This fact is relevant for the issue of robustness to illumination. I made sure the objects were visible to a certain minimal extent, 1300 pixels, corresponding to an area of approximately 140 cm^2.

I created a *gallery of 13 toy objects* of which all scenes are composed. The objects are: *basket, bear, book, box, candleman, candlewoman, clock, elephant, glass-of-marbles, nutcracker, rattle, windmill,* and *zebra.* This gallery represents a certain selection. I have excluded objects that are too small, that have too little inner structure (although I included *basket, rattle*, and *glass-of-marbles* in spite of their poor inner structure), and that are not compact (although I included *clock* with its two holes and *zebra* with its thin legs). The problem with small objects was that with a resolution of 128×128 pixels the corresponding model graphs contain too little information and yield good graph similarity in wrong places. This is similarly the case with objects that are too homogeneous. For objects with openwork structure (e.g. letter scales that we tried) the extended receptive fields caused problems, being too sensitive to the background.

7.3.2 Results

Two of the scenes used in the experiments are shown in Figure 7.1. Figure 7.3 shows analyses obtained with Algorithm 1, and Figure 7.4 with Algorithm 2.

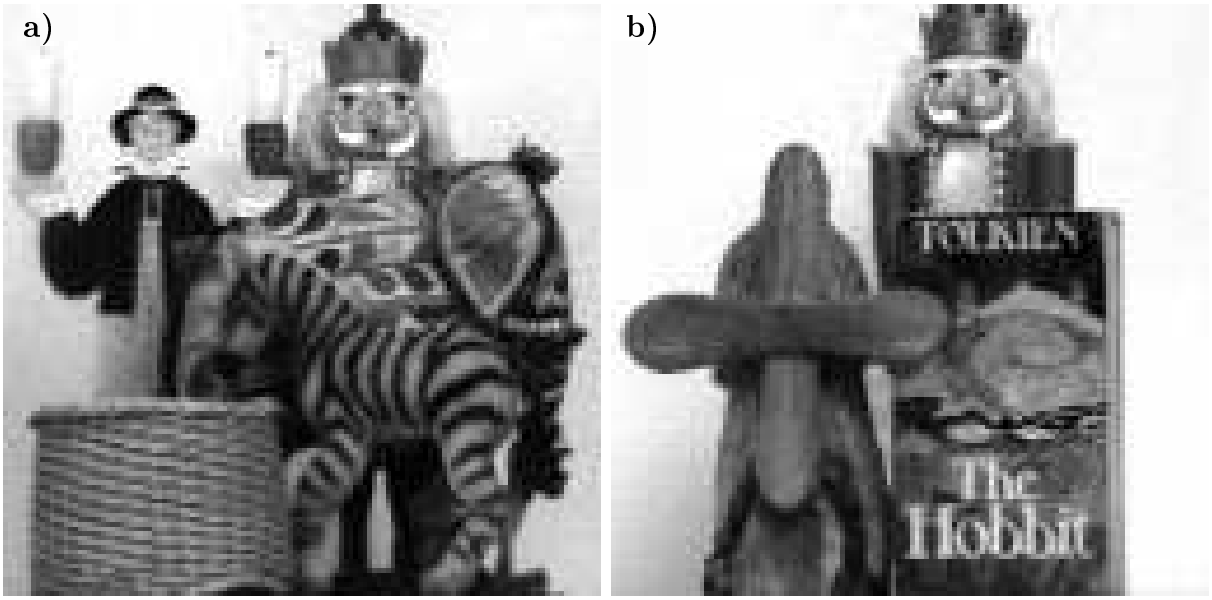

Figure 7.1: Two of the 30 scene images. **a)** contains *zebra, basket, elephant, candleman*, and *nutcracker*, **b)** contains *windmill, book*, and *nutcracker*. There are altogether 13 models in the gallery, and 121 objects in the scenes, each scene containing between three and six objects. The resolution of the images is 128^2 pixels with 256 grey levels.

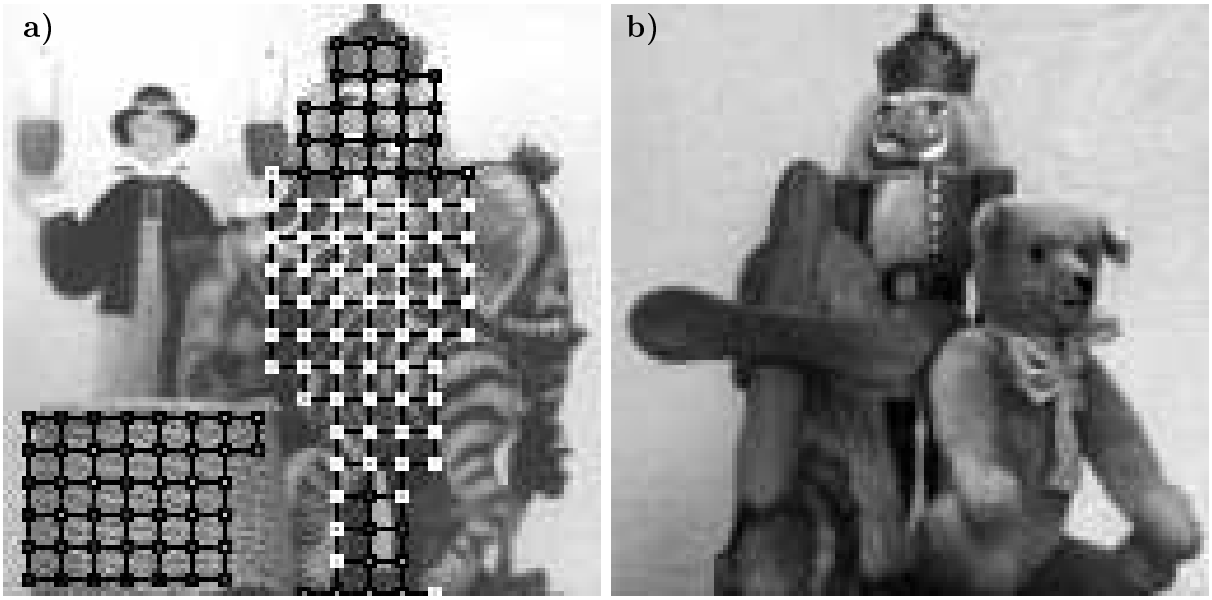

Figure 7.2: Picture of the image graphs for *basket* and *nutcracker* as matched in Scene 7.1.a. Black and white frames denote visible and occluded nodes, respectively. The state of visibility was determined by Algorithm 2. Center pixels of nodes code for similarity of the model jets to corresponding image jets, from white (low similarity) to black (high similarity). **b)** One example of a scene with rotated objects ($\approx 20°$) not included in the statistical investigation (for analysis see Figures 7.3.c and 7.4.c.)

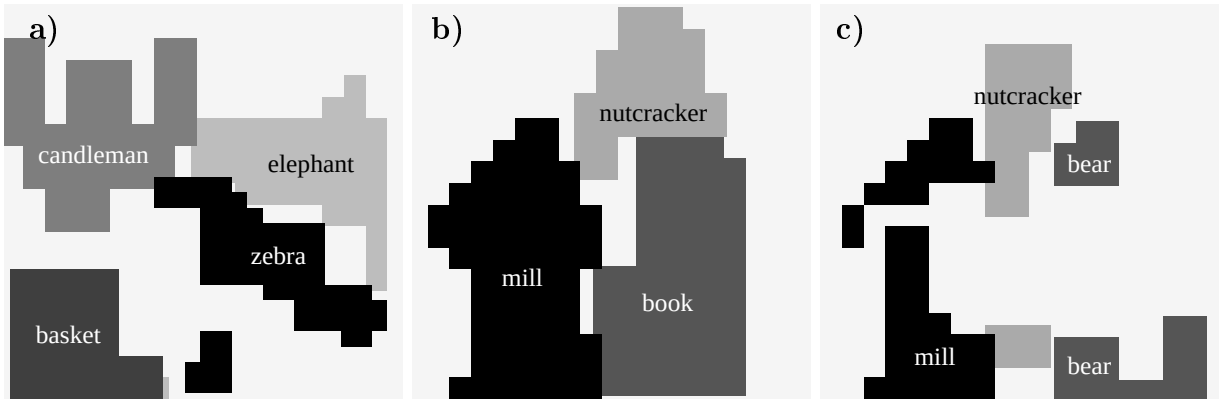

Figure 7.3: Analyses of the scenes in Figures 7.1.a, b, and 7.2.b respectively, by Algorithm 1. Visible regions of the matched model graphs are shown, from front (black) to back (light). **a)** The algorithm recognized *zebra, basket, candleman* and *elephant* in Scene 7.1.a, but it missed the *nutcracker*, not finding the lower part under the zebra and discarding the identified head region as too small in area. For the zebra, large parts are interpreted as occluded, because of perturbation of inner jets by overlap with the background. **b)** Scene 7.1.b is analyzed correctly. Altogether 80% of the 121 objects were recognized correctly while 2 models were accepted erroneously by this algorithm. **c)** shows the analysis of Scene 7.2.b. All objects are recognized correctly, but again large parts are interpreted as occluded. Although I did not investigate the robustness of the system against rotation in depth systematically, several examples such as this one suggest that this algorithm might be robust up to approximately 10°.

Algorithm 1 gave the following performance. From a total of 121 objects, 24 were not recognized correctly, leaving 80% recognized correctly. Only 2 objects were erroneously accepted. Some decisions regarding occlusion were unsatisfactory, (see, for instance, the candleman in Figure 7.3.a), which is not surprising since no interactions between objects were taken into account (and the point of Algorithm 1 was recognition only, anyway).

Algorithm 2 produced 21 completely correct analyses from among the full set of 30 scenes. For 3 scenes it made errors regarding the occlusion order, always for pairs of weakly overlapping objects. In the remaining 6 scenes, 3 models were accepted although the corresponding object was not present, and 4 objects which were present were not recognized. Among the latter, 2 were matched at the wrong position and 2 objects were matched correctly but were rejected on the basis of too poor a graph similarity value. In all, 96.7% of the objects are recognized correctly and with confidence.

Some of the errors committed by the system are instructive. In the total set of experiments, there were only 2 cases in which the best match for a model graph in the scene was found in the wrong place. One case concerned the *glass-of-marbles*, which has little internal structure and is partially transparent. In the other case, *candlewoman* was matched to the fairly similar *candleman* while the proper object was heavily occluded. To deal with this type of error, one could produce several matches for each model. The system would then have to manage more matches, but it would be more likely to find correct matches (and would be equipped to match multiple instances of the same object type).

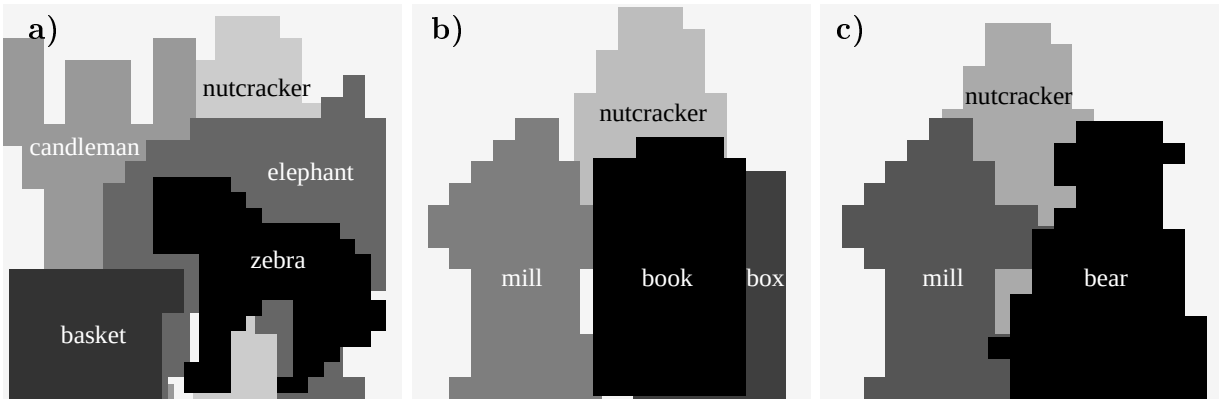

Figure 7.4: Analyses of the scenes in Figures 7.1.a, b and 7.2.b by Algorithm 2. **a)** In Scene 7.1.a, all objects and their occlusion relations have been recognized correctly. **b)** In Scene 7.1.b, a box (rightmost model) was erroneously recognized. Besides, the book mistakenly was put in front of the windmill. Altogether, 96.7% of the 121 objects were recognized correctly, and 3 models were accepted erroneously by this algorithm. **c)** The third example, Scene 7.2.b, was also recognized correctly. Algorithm 2 seems to be more robust against rotation in depth than the first. Up to approximately 15° might be of little effect. Both algorithms could be improved in this respect if graph distortions were permitted (see Chapter 5).

7.4 Discussion

The system presented here is a very natural extension of the EGM system described in (Lades et al., 1993) to the recognition of partially occluded objects and the analysis of scenes. The *labeled graph* as the fundamental data structure proved to be appropriate for visual object representation in the presence of significant occlusions. Only the status variable indicating whether a node is visible or not had to be added. It was crucial for the success of the system that graphs provide information about the location and neighborhood relations of the nodes and the attached local features. This was essential for back-labeling the scene with recognition information and with occlusion relations.

This application to scenes also reveals a weakness of the local feature representation based on *Gabor wavelets*. The Gabor wavelets of lower frequency have a non-negligible extension compared to the distance of the nodes and the extension of the objects. This becomes evident for the zebra, for which the legs were considered to be occluded by Algorithm 1, because their similarity was too much degraded by the *background* (see Figure 7.3). A solution to this problem would be to increase the resolution of the images and to emphasize more the high frequency kernels. A more fundamental solution was demonstrated by Pötzsch (1994), who showed that the influence of background to a jet can be suppressed by a linear transformation imitating the operation of cutting the image information into two half planes and keeping only one. A second disadvantage of the Gabor wavelet preprocessing is that it depends on internal texture. This system cannot deal with structureless objects which are mainly defined by their contour. For such objects an edge-based representation would be more appropriate.

The project presented here was undertaken as a pilot study to investigate the problems involved in analyzing cluttered scenes. The system still has to be reformulated in a fully neural system, based on Dynamic Link Matching.

Chapter 8

Conclusion

One of the intentions of this work was to show that the labeled graph is a powerful and flexible data format providing the syntactical structure missing in the vector format typically used in conventional neural net applications. The syntactical links between elementary features play a crucial role in the applications presented on several levels. They were either made explicit or they were implicit in the chosen data formats, but would have to be realized explicitly in a mature system.

Firstly, the individual coefficients of the jet representation are implicitly bundled by links to generate a description of a local patch of grey values. Secondly, nodes labeled with jets were linked together by edges in order to build a graph representing individual objects. Thirdly, the graphs can be matched to an image by dynamically establishing connections from the nodes of the graph to a subset of nodes in the image. (These three aspects of syntactical linking have already been used in previous systems (Buhmann et al., 1989; Lades et al., 1993). The following types of syntactical structures are newly introduced in this work.) In the fourth place, sets of jets were attached to nodes and serve as a collection of alternatives if the appearance of a certain object part, e.g. an eye, varies significantly. In the fifth place, context information about different properties of the objects was incorporated and expressed by attached attributes (male, bearded, etc.). And finally, model graphs that were matched to a scene competed with each other through inhibition between overlapping nodes.

This is only a relatively small number of examples of how syntactical links may be used for perceptual tasks, and one major goal for future research will be to investigate other possibilities of useful syntactically linked structures, some of which have already been mentioned in Chapter 2.

A second purpose of this work was to demonstrate that labeled graphs can actually be processed in a neural architecture and that serious recognition tasks can be performed on this basis. For the first time a complete DLM face recognition system has been developed, able to recognize faces against a gallery of more than one hundred model faces. Nevertheless DLM is relatively slow and cumbersome compared to the algorithmic version, the EGM. Several features of the algorithmic models have yet to be implemented in a fully neural system, especially the recognition of occluded objects in cluttered scenes.

All systems presented here, the one based on DLM as well as those based on EGM, have various shortcomings. Firstly, they are extreme in the sense that they build object representations directly from low-level features. It is necessary to introduce some mid-level features that can mediate between models and image information. Secondly, no low-

level segmentation cues are used to improve speed and reliability of object recognition. So far the objects have either been presented in front of a homogeneous background, or the segmentation was only guided top-down by object knowledge. Thirdly, the generation of graph representations is very artificial. Model graphs are either defined by hand, or a very primitive segmentation procedure is used to determine the contour of a new object. Finally, more control structure is required to enable the system to build a knowledge database autonomously without user interaction.

Conceptionally, it will be necessary to develop further the ideas sketched in Chapter 2. The goal is to define a small set of fundamental operations on graphs that suffice to let complex structures emerge in an autonomous system able to learn from sensory input, to organize knowledge about its environment, and finally to generate useful action in terms of a given goal.

Appendix A

Preprocessing with Gabor Wavelets

Abstract: Gabor wavelets have the shape of plane waves restricted by a Gaussian envelope function. Convolving an image with a whole family of Gabor wavelets of different size and orientation provides a set of complex coefficients at each pixel. This set is called a jet $\mathcal{J}$ and represents a local patch of grey values. Due to the wave character of the kernels, the coefficients have a phase ϕ varying with the characteristic frequency of the kernel and a slowly changing magnitude a. Jets can be compared by similarity functions $\mathcal{S}_\phi$ and $\mathcal{S}_a$. If phase information is taken into account ($\mathcal{S}_\phi$), the spatial distance or disparity $\vec{d}$ between two jets taken from approximately the same object location can be estimated.

A.1 Gabor Wavelet Transformation

A *jet* is a special type of *local feature* describing a small patch of grey values in an image $\mathcal{I}(\vec{x})$ around a given pixel $\vec{x} = (x, y)$. It is based on a wavelet transform, defined as a convolution

$$\mathcal{J}_j(\vec{x}) = \int \mathcal{I}(\vec{x}')\psi_j(\vec{x} - \vec{x}')d^2\vec{x}' \tag{A.1}$$

with a family of *Gabor kernels*

$$\psi_j(\vec{x}) = \frac{k_j^2}{\sigma^2} \exp\left(-\frac{k_j^2 x^2}{2\sigma^2}\right)\left[\exp(i\vec{k}_j\vec{x}) - \exp\left(-\frac{\sigma^2}{2}\right)\right], \tag{A.2}$$

having the shape of plane waves with *wave vector* $\vec{k}_j$ restricted by a Gaussian envelope function. I employ a discrete set of 5 different *frequencies*, index $\nu = 0, ..., 4$, and 8 *orientations*, index $\mu = 0, ..., 7$,

$$\vec{k}_j = \begin{pmatrix} k_{jx} \\ k_{jy} \end{pmatrix} = \begin{pmatrix} k_\nu \cos\varphi_\mu \\ k_\nu \sin\varphi_\mu \end{pmatrix}, \quad k_\nu = 2^{-\frac{\nu+2}{2}}\pi, \ \varphi_\mu = \mu\frac{\pi}{8}, \tag{A.3}$$

with index $j = \mu + 8\nu$. By this sampling the frequency space is evenly covered within a reasonable band-pass. The Gauss width is σ/k with $\sigma = 2\pi$, and the kernels are *DC-free*, i.e. the integral $\int \psi_j(\vec{x})d^2\vec{x}$ vanishes. Since this is a wavelet transform, the family of kernels is *selfsimilar* in the sense that all kernels can be generated from one *mother wavelet* by dilation and rotation.

A jet $\mathcal{J}$ is defined as the set $\{\mathcal{J}_j\}$ of 40 complex coefficients

$$\mathcal{J}_j = a_j \exp(i\phi_j) \tag{A.4}$$

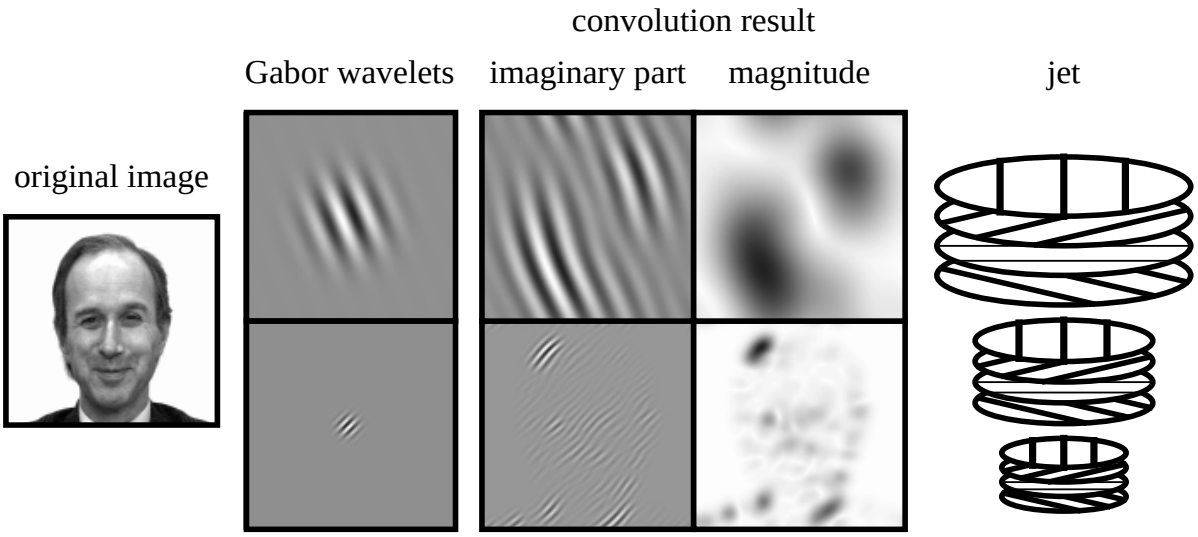

Figure A.1: The visual preprocessing is based on the Gabor wavelet transform. The wavelets have the shape of plane waves (5 different frequencies $\times$ 8 different orientations) restricted by a Gaussian envelope function. A convolution yields 40 complex coefficients referred to as a jet. The phase of the coefficients varies with the main frequency (see imaginary part) and their magnitude varies slowly.

---

with amplitudes $a_j(\vec{x})$ slowly varying with position and phases $\phi_j(\vec{x})$ varying with the spatial frequency given by the characteristic wave vector $\vec{k}_j$ (see Figure A.1).

Gabor wavelets were chosen for their technical properties and biological relevance. Since they are DC-free, they provide *robustness* against varying brightness in the image. Robustness against varying contrast can be obtained by normalizing the jets. The limited localization in space and frequency yields a certain amount of robustness against translation, distortion, rotation, and scaling. Only the phase changes drastically with translation, but that can be used for estimating displacement, as will be shown later. A disadvantage of the large kernels is their sensitivity to background variations. But as was shown by PÖTZSCH (1994), if the object contour is known, the influence of the background can be suppressed. Finally, the Gabor wavelets are closely related to the receptive fields of simple cells in the vertebrate visual cortex (POLLEN & RONNER, 1981; JONES & PALMER, 1987; DEVALOIS & DEVALOIS, 1988).

A.2 Saliency

Saliency indicates whether an image location is considered to be interesting solely on the basis of low level information (see for example MANJUNATH et al., 1992). I use the norm of the jet as a simple *saliency* measure:

$$\mathcal{N}(\mathcal{J}) = \sqrt{\sum_j a_j^2}, \tag{A.5}$$

i.e. a jet is salient if it represents rich textural structure of high contrast. This saliency is used in Chapter 4 to initialize the attention layer.

A.3 Comparing Jets

The quick phase variations cause problems. Jets taken from an image few pixels apart from each other have very different coefficients, although they represent almost the same local feature. I therefore either ignore the phase or compensate for its variations explicitely. The first leads to the *jet similarity* function

$$\mathcal{S}_a(\mathcal{J},\mathcal{J}') = \frac{\sum_j a_j a'_j}{\sqrt{\sum_j a_j^2 \sum_j a_j'^2}} \tag{A.6}$$

already used by BUHMANN et al. (1992) and LADES et al. (1993). With $\mathcal{J}$ a fixed jet, and $\mathcal{J}' = \mathcal{J}'(\vec{x})$ the jets at positions $\vec{x}$ in an image, $\mathcal{S}_a(\mathcal{J},\mathcal{J}'(\vec{x}))$ is a smooth function, and its local optima have large attractor basins suitable for very simple methods to search for them (see Figure A.2). Typically gradient descent or diffusion processes converge rapidly and reliably.

Using *phase information* has two potential advantages. Firstly, phase information can help to discriminate between patterns with similar amplitudes, and secondly, since phase varies so quickly with location, it provides a means to locate jets in an image precisely. In the following I assume that the two jets $\mathcal{J}$ and $\mathcal{J}'$ refer to similar object locations with a small relative displacement $\vec{d}$. The phase shifts can then approximately be compensated for by the term $\vec{d}\vec{k}_j$, and the similarity can be defined as

$$\mathcal{S}_\phi(\mathcal{J},\mathcal{J}') = \frac{\sum_j a_j a'_j \cos(\phi_j - \phi'_j - \vec{d}\vec{k}_j)}{\sqrt{\sum_j a_j^2 \sum_j a_j'^2}}. \tag{A.7}$$

Before computing the similarity, the displacement $\vec{d}$ has to be estimated. This can be done by maximizing $\mathcal{S}_\phi$ in its Taylor expansion, as will be explained in the following section. The great advantage of this second similarity function is actually that it yields this displacement information. Profiles of similarities and estimated displacements are shown in Figure A.2.

A.4 Disparity Estimation

In order to estimate the displacement vector $\vec{d} = (d_x, d_y)$, I have adopted a method used for disparity estimation (THEIMER & MALLOT, 1994) based on (FLEET & JEPSON, 1990). The idea is to maximize the similarity $\mathcal{S}_\phi$ in its Taylor expansion:

$$\mathcal{S}_\phi(\mathcal{J},\mathcal{J}') \approx \frac{\sum_j a_j a'_j [1 - 0.5(\phi_j - \phi'_j - \vec{d}\vec{k}_j)^2]}{\sqrt{\sum_j a_j^2 \sum_j a_j'^2}}\ . \tag{A.8}$$

Setting $\frac{\partial}{\partial d_x}\mathcal{S}_\phi = \frac{\partial}{\partial d_y}\mathcal{S}_\phi = 0$ then leads to

$$\underbrace{\sum_j a_j a'_j k_{jx}(\phi_j - \phi'_j)}_{\Phi_x} = d_x \underbrace{\sum_j a_j a'_j k_{jx} k_{jx}}_{\Gamma_{xx}} + d_y \underbrace{\sum_j a_j a'_j k_{jx} k_{jy}}_{\Gamma_{xy}}, \tag{A.9}$$

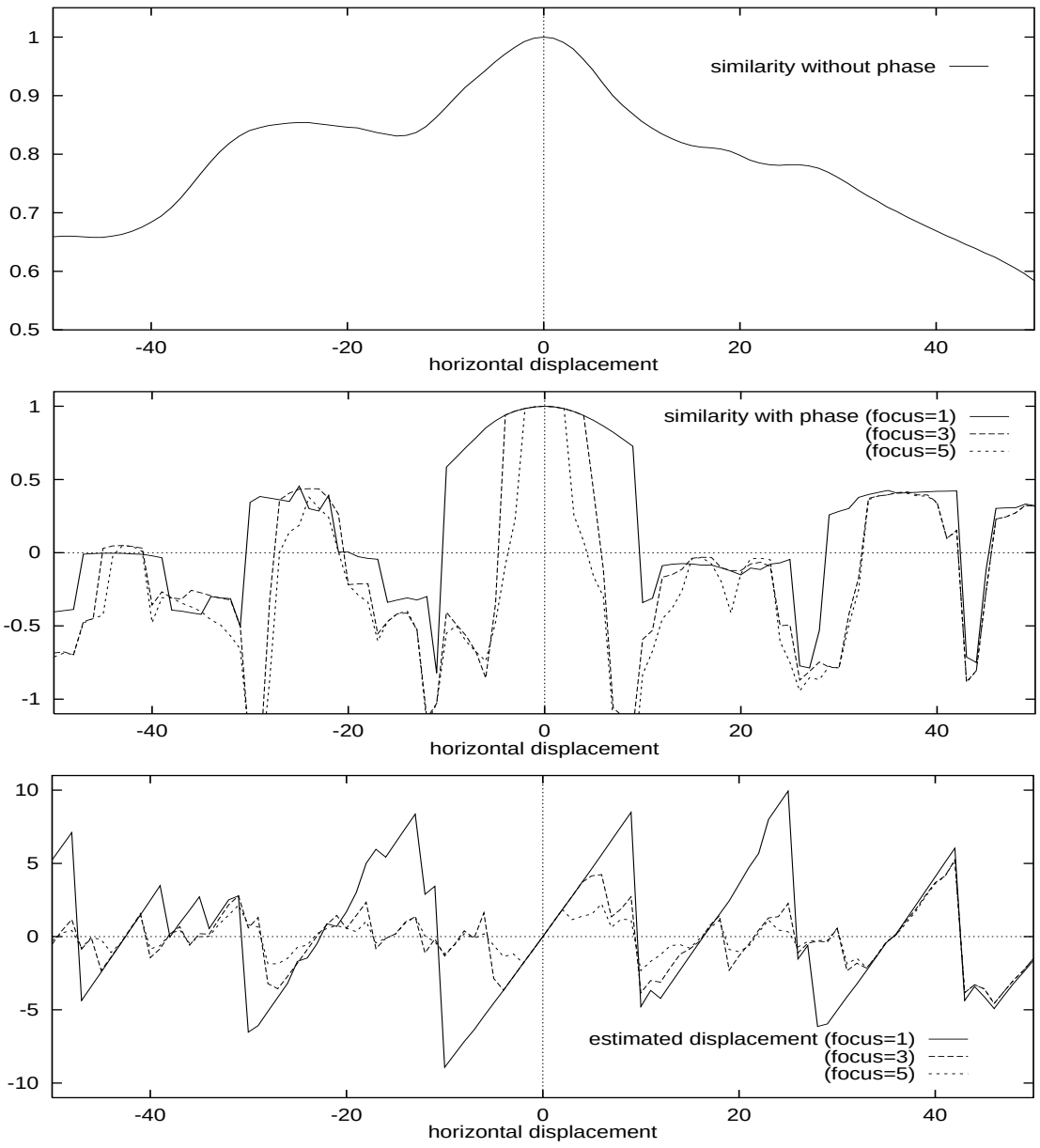

Figure A.2: a) Similarity $\mathcal{S}_a(\mathcal{J}(\vec{x}_1), \mathcal{J}'(\vec{x}_0))$ with jet $\mathcal{J}'$ taken from the left eye of the face shown in Figure A.1, and jet $\mathcal{J}$ taken from pixel positions of the same horizontal line, $\vec{x}_1 = \vec{x}_0 + (d_x, 0), d_x = -50, ..., 50$. The similarity potential is smooth and has a large attractor basin. **b)** Similarity $\mathcal{S}_\phi(\mathcal{J}(\vec{x}_1), \mathcal{J}'(\vec{x}_0))$ and **c)** estimated displacement $\vec{d}(\mathcal{J}(\vec{x}_1), \mathcal{J}'(\vec{x}_0))$ for the same jets as in a). The focus varies from 1 to 5. The similarity potentials have many more local optima, especially for a high focus value. The right eye is 24 pixels away from the left eye, generating a local maximum for both similarity functions close to $d_x = -24$. The estimated displacement is very precise around the 0-position and rougher at other local optima, especially at the other eye. The maximum displacements are half the wavelength of the highest frequency kernel taken into account for the first displacement estimation (about 8, 4, and 2 pixels for focus values 1, 3, and 5 respectively).

$$\underbrace{\sum_j a_j a'_j k_{jy}(\phi_j - \phi'_j)}_{\Phi_y} = d_x \underbrace{\sum_j a_j a'_j k_{jy} k_{jx}}_{\Gamma_{yx}=\Gamma_{xy}} + d_y \underbrace{\sum_j a_j a'_j k_{jy} k_{jy}}_{\Gamma_{yy}}, \tag{A.10}$$

which can be solved for $\vec{d}$ if the determinant $\Gamma_{xx}\Gamma_{yy} - \Gamma_{xy}\Gamma_{yx}$ does not vanish:

$$\vec{d}(\mathcal{J}, \mathcal{J}') = \begin{pmatrix} d_x \\ d_y \end{pmatrix} = \frac{1}{\Gamma_{xx}\Gamma_{yy} - \Gamma_{xy}\Gamma_{yx}} \begin{pmatrix} \Gamma_{yy} & -\Gamma_{yx} \\ -\Gamma_{xy} & \Gamma_{xx} \end{pmatrix} \begin{pmatrix} \Phi_x \\ \Phi_y \end{pmatrix}. \tag{A.11}$$

This equation yields a straightforward method for estimating the displacement or disparity between two jets taken from object locations close enough that their Gabor kernels are highly overlapping. Without further modifications, this equation can determine displacements up to half the wavelength of the highest frequency kernel, which would be two pixels for $k_{max} = \pi/2$. The range can be increased by using low frequency kernels only. For the largest kernels the estimated displacement may be 8 pixels. One can then proceed with the next higher frequency level and refine the result, possibly by correcting the phases of the higher frequency coefficients by multiples of 2π according to the disparity estimated on the lower frequency. I have referred to the number of frequency levels used for the first displacement estimation as *focus*. A focus of 1 indicates that only the lowest frequency level is used and that the estimated displacement may be up to 8 pixels. A focus of 5 indicates that all five levels are used, and the disparity may only be up to 2 pixels. If one has access to the whole image of jets, one can also work iteratively. Assume a jet $\mathcal{J}$ is given for which the accurate position is needed in an image around a starting point $\vec{x}_0$. Comparing $\mathcal{J}$ with the jet $\mathcal{J}_0 = \mathcal{J}(\vec{x}_0)$ gives an estimated displacement of $\vec{d_0} = \vec{d}(\mathcal{J}, \mathcal{J}(\vec{x}_0))$. Then a jet $\mathcal{J}_1$ is taken from position $\vec{x}_1 = \vec{x}_0 + \vec{d_0}$ and the displacement is estimated again. But since the new location is closer to the correct position, the new displacement $\vec{d_1}$ will be smaller and can be estimated more accurately. This procedure will eventually converge with a remaining subpixel displacement. This is the iterative scheme that is used in the matching process described in Chapter 5.

Appendix B

Zusammenfassung in deutscher Sprache

B.1 Einleitung

In der Neuroinformatik dominiert die Vektorrepräsentation zur Darstellung von Eingabedaten. Ein Bild ist in technischen Systemen zunächst ein Feld von Pixeln, wobei jedes Pixel eine Position und einen Grauwert hat. In der typischen Vektordarstellung werden die Positionen nur als eindeutige Adressen verwendet, die repräsentierten räumlichen Beziehungen gehen jedoch verloren. Das hat fatale Konsequenzen, was in Abbildung 1.1 an einem einfachen Beispiel demonstriert wird. Berücksichtigt man Nachbarschaftsbeziehungen, dann wird man die beiden rechten Bilder als gegeneinander verschobene Kopien erkennen. Mit der Hamming-Distanz als einem vektoriellen Abstandsmaß wird man jedoch zu dem Ergebnis kommen, daß die linken beiden Bilder einander ähnlicher sind.

Die Vektorrepräsentation von Bilddaten in der Neuroinformatik ist ein konzeptionelles Problem, das auch durch geeignete Normierung der Eingabebilder oder durch Modelle wie dem Neokognitron nicht befriedigend gelöst wird. Die vorliegende Arbeit beschäftigt sich mit etikettierten Graphen als einem alternativen Konzept zur Objektrepräsentation, das explizit Relationen zwischen Elementen kodieren kann. So werden Bilder aus lokalen Merkmalen *und* deren räumlichen Beziehungen zu Graphen zusammengefaßt.

Neben den etikettierten Graphen werden die Prinzipien der dynamischen Graphenanpassung erläutert. Dynamische Graphenanpassung ist ein von VON DER MALSBURG entwickeltes neuronales Konzept zum Vergleichen und Anpassen von etikettierten Graphen. In vier Anwendungen wird die Leistungsfähigkeit dieser Konzepte demonstriert. Alle Anwendungen basieren auf einer visuellen Vorverarbeitung, die durch die Gabor Wavelet-Transformation beschrieben wird.

B.2 Etikettierte Graphen zur Objektrepräsentation

Wahrnehmung erfordert zunächst eine geeignete Repräsentation der Reizmuster. Dabei gehe ich davon aus, daß jeder Sinn im wesentlichen in drei Unterräumen organisiert ist, und daß jeder Reiz als etikettierter Graph (labeled graph) repräsentiert werden kann. Der erste der drei Unterräume ist der Sinnesraum (sensory space), er ist die zweidimensionale Retina für das visuelle System, die Cochlea im auditorischen System, oder die Hautoberfläche unseres Tastsinnes. Im Sinnesraum spielen Relationen oder relationale Mermale (relational features), wie z.B. Abstände, die wesentliche Rolle. Menschen sind sehr genau in der Beurteilung ob drei Punkte auf einer Geraden liegen, ignorieren aber weitgehend den Ort des Musters auf der Retina. Die Relationen im Sinnesraum werden durch Kanten im Graphen dargestellt, z.B. etikettiert mit Abstandsinformationen. Der zweite Unterraum ist der Merkmalsraum (feature space). Merkmale, oder besser lokale Merkmale (local features), sind im visuellen System z.B. Farbe, Textur oder Orientierung einer Kante. Hier spielt die absolute Empfindung eine größere Rolle als Relationen. Entsprechend werden die Merkmale durch Knoten dargestellt, z.B. etikettiert mit Farb- oder Texturinformation. Der dritte Unterraum eines jeden Sinnes ist die Zeit (time). Sie spielt in verschiedener Hinsicht eine besondere Rolle. Erwähnt werden soll hier nur, daß Zeit ein starker Hinweis auf Kausalität ist. Sie ist außerdem allen Sinnen gemeinsam und vermutlich der wesentliche Schlüssel zur Integration der verschiedenen Sinne.

Zur Bildung von etikettierten Graphen von Reizmustern müssen Knoten aus dem Bild ausgewählt und durch Kanten zu Graphen verbunden werden. Im einfachsten Fall werden die Knoten auf einem regelmäßigen Gitter angeordnet. Der Abstand der Knoten hängt dann nur von der Ausdehnung und Komplexität der verwendeten Merkmale ab. Aufwendiger, aber geeigneter, ist die Auswahl von sogenannten auffallenden Punkten (salient points), von denen man annimmt, daß sie wichtige Information tragen. Es ist jedoch schwierig, diese Punkte allein auf der Basis von lokaler Bildinformation und ohne Objektwissen zu definieren. Für die Auswahl geeigneter Kanten zwischen den Knoten dienen Gruppierungshinweise wie sie aus der Psychophysik bekannt sind. Hier ist vor allem das Prinzip der Nähe (proximity) zu nennen. Das kann sich auf alle drei Unterräume beziehen: räumliche Nähe im Sinnesraum, Ähnlichkeit im Merkmalsraum und Koinzidenz in der Zeit (siehe Abbildung 2.2). In allen drei Unterräumen werden Knoten mit identischen Eigenschaften stark miteinander verbunden, und die Verbindung wird mit zunehmendem Abstand schwächer. Überträgt man die induzierten Verbindungen auch auf alle benachbarten Knoten, so entsteht ein stark verknüpfter Graph des Reizmusters, der als Basis für alle weiteren Schritte dient (siehe Abbildung 2.3).

Sind Graphen von Reizmustern gebildet und abgespeichert, so möchte man als nächstes Graphen miteinander vergleichen. Dabei werden zwei Graphen als ähnlich angenommen, wenn sie sowohl in ihren lokalen als auch in ihren relationalen Merkmalen ähnlich sind. Den entsprechenden Prozeß des Vergleichens nennt man Graphenanpassung (graph matching) (siehe Abbildung 2.4). Es ist auch denkbar, daß Graphen im wesentlichen aufgrund der gemeinsamen Struktur und unabhängig von den lokalen Merkmalen miteinander verglichen werden können. Das würde auch den Vergleich von Graphen unterschiedlichen Ursprungs ermöglichen, z.B. aus dem visuellen und dem auditorischen System. Ein solcher Prozeß kann als Analogiebildung interpretiert werden

(siehe Abbildung 2.5). Mit der Graphenanpassung lassen sich auch Teilgraphen vergleichen und eine einfache Strukturierung der gespeicherten Graphen vornehmen. Stellt man fest, daß Teile einer Menge von Graphen identisch sind, so bildet man einen sog. Fusionsgraphen (fusion graph), in dem die gemeinsamen Teilgraphen nur einmal repräsentiert sind (siehe Abbildung 2.6). Der Fusionsgraph hat zum einen den Vorteil, daß er weniger Speicherkapazität als die Einzelgraphen erfordert. Zum anderen hat er Generalisierungsfähigkeiten, da die Teilstücke nun auch in anderen Kombinationen zusammengesetzt werden können. Damit gekoppelt ist das Problem, daß der Fusionsgraph die ursprünglichen Graphen nicht mehr eindeutig kodiert. Diese Eindeutigkeit läßt sich jedoch durch Einführung von sogenannten Kardinalknoten (cardinal cells) wieder herstellen. Geschieht dies in geeigneter Weise, können sowohl Generalisierungsfähigkeit als auch die Eindeutigkeit der ursprünglichen Graphen in kontrollierter Weise miteinander kombiniert werden (siehe Abbildung 2.7).

In den in dieser Arbeit vorgestellten Anwendungen geht es um Gesichtserkennung und Szenenanalyse. Beides erfordert spezielle Graphenstrukturen. In der Gesichtserkennung ist es vorteilhaft, die gemeinsame Struktur von Gesichtern auszunutzen. Die Gesichtsgraphen für eine Ansicht (Frontalansicht, Halbprofil oder Profil) werden entsprechend alle die gleiche Struktur haben, z.B. einen Knoten auf dem rechten Auge, einen Knoten auf der Nasenspitze, usw. Dies ermöglicht es, die Gesichtsgraphen stapelartig zu kombinieren, wobei etwa alle rechten Augenknoten miteinander verbunden werden, ebenso alle Nasenknoten, usw. Dieser Fusionsgraph repräsentiert das gesamte Wissen des Systems über Gesichter und wird entsprechend *allgemeines Gesichtswissen* (general face knowledge) genannt. Es ermöglicht auch, neue Gesichter aus schon bekannten zusammenzusetzen. Zur weiteren Analyse von Gesichtern wird das allgemeine Gesichtswissen mit Kontextinformation versehen. In der vorgestellten Anwendung bezieht sich das auf das Geschlecht der Personen, ob sie bärtig sind und ob sie eine Brille tragen (siehe Abbildung 2.8 links). Für die Szenenanalyse sind die Objekte sehr unterschiedlicher Natur und daher nicht in einem Fusionsgraphen kombinierbar. Jedoch müssen die Graphen zur Analyse einer Szene miteinander um Bildfläche konkurrieren, wenn man annimmt, daß an jedem Ort nur ein Objekt sichbar sein kann. Dabei muß berücksichtigt werden, daß sich die Objekte nicht gegenseitig durchdringen können. Das kann durch die Forderung einer eindeutigen Tiefenreihenfolge erreicht werden (siehe Abbildung 2.8 rechts).

B.3 Prinzipien der dynamischen Graphenanpassung

Neuronale Netzwerke scheinen zunächst ungeeignet zu sein, etikettierte Graphen zu verarbeiten. Das liegt im wesentlichen daran, daß konventionelle neuronale Netze Relationen zwischen Neuronen nur durch deren synaptische Verbindungsstärke ausdrücken können, die zudem nur auf einer langsamen Zeitskala durch Lernen veränderlich ist. VON DER MALSBURG hat in der von ihm vorgeschlagenen *Dynamic Link Architecture* die konventionellen neuronalen Netze konzeptionell um die Möglichkeit des dynamischen Bindens von Neuronen durch Korrelation ihrer Zeitsignale und um schnell und reversibel schaltende Synapsen erweitert. Beide Konzepte zusammen ermöglichen die dynamische Graphenanpassung (dynamic link matching), ein Prozeß zum aufeinander Abbilden und Vergleichen von etikettierten Graphen.

Ein etikettierter Graph wird repräsentiert durch eine Schicht von Neuronen. Jedem Neuron ist ein lokales Merkmal zugeordnet, und laterale Verbindungen induzieren eine Metrik und somit Abstände zwischen den Neuronen. In der Graphenanpassung sollen zwei solche Schichten entsprechend der Ähnlichkeit ihrer repräsentierten Muster aufeinander abgebildet werden. Zu Beginn sind beide Schichten vollständig miteinander verschaltet, beschrieben durch die Verbindungsmatrix. Schließlich soll jedoch jedes Neuron der einen Schicht mit nur einem Neuron der anderen Schicht verbunden sein (siehe Abbildung 3.1). Die dynamische Graphenanpassung basiert auf folgenden vier Prinzipien (siehe Abbildungen 3.2 bis 3.5): Erstens, die lateralen Verbindungen einer Schicht induzieren eine Dynamik, die Nachbarschaften durch Korrelationen ausdrückt, benachbarte Neurone haben korrelierte Zeitsignale, entfernte Neurone feuern unkorreliert. Zweitens, sind zwei Schichten mit einer Identitätsabbildung verbunden, so werden sich die Aktivitätsdynamiken beider Schichten synchronisieren. Korrespondierende Neurone feuern korreliert. Drittens, die Synchronisation ist robust gegen Rauschen, Verzerrungen und Teilverdeckungen. Sie ist ohnehin invariant gegen Translation, Rotation, und Spiegelung. Diese Störungen werden immer auftreten, wenn die Schichten reale Bilder repräsentieren. Viertens, die Verbindungsstruktur kann sich aufgrund der induzierten Korrelationen zu einer eins-zu-eins Abbildung (one-to-one mapping) entwickeln. Dies ist ein Wechselwirkungsprozeß, da die Korrelationen bei entwickelter Verbindungsstruktur ihrerseits auch verbessert werden (siehe Abbildung 3.6).

Die dynamische Graphenanpassung ist einer von wenigen Ansätze zur translationsinvarianten Objekterkennung in neuronaler Architektur. Trotz ihrer Möglichkeiten, die im nächsten Abschnitt kurz erläutert wird, kann sie die Leistungsfähigkeit unseres visuellen Systems aus zwei Gründen nicht erklären: Erstens, die dynamische Graphenanpassung ist zu langsam. Die enorm kurzen Erkennungszeiten unseres visuellen Systems lassen sich durch die relativ langsame Aktivitäts- und Verbindungsdynamik nicht erklären. Man kann jedoch annehmen, daß die dynamische Graphenanpassung in einem frühen Entwicklungsstadium zur Objekterkennung verwendet wird, und daß sich später effizientere Mechanismen entwickeln, die jedoch ein hohes Maß an visueller Erfahrung erfordern. Zweitens, die anfängliche vollständige Verschaltung zwischen den Schichten erfordert zu viele Verbindungen. Die Lösung dieses Problems liegt offensichtlich in der Einführung von hierarchischen Strukturen, wie schon ansatzweise gezeigt wurde.

B.4 Gesichtserkennung mit dynamischer Graphenanpassung

Als Aktivitätsdynamik der dynamischen Graphenanpassung wurde bisher eine verwendet, die stationäre Aktivitätsflecken erzeugt. Dabei wurde die gesamte Dynamik auf recht künstliche Weise kontrolliert und mit den folgenden vier Schritten iteriert: Erzeugung eines Aktivitätsflecks auf einer Schicht, initiiert durch Rauschen; Erzeugung eines Aktivitätsflecks auf der anderen Schicht aufgrund der durch die Verbindungsmatrix propagierten Aktivität des ersten Flecks; Anwendung eines Lernschrittes für die Verbindungsstruktur; Zurücksetzen der Schichtaktivitäten auf Null. Außerdem ist die dynamische Graphenanpassung bisher noch nicht zu einem vollständigen Erkennungssystem entwickelt worden. Es wurden meist wenige Modelle, typischerweise drei, verwendet, und die Erkennungsentscheidung wurde aufgrund von Größen, z.B. der gemittelten

Verbindungsstärke, getroffen, die in einem biologischen System nicht direkt zugänglich sind. Es war das Anliegen dieser Arbeit, die dynamische Graphenanpassung zu einem vollständigen Erkennungssystem mit kontinuierlicher und autonomer Dynamik zu entwickeln.

Das Ziel der kontinuierlichen Dynamik wurde durch Einführung von verzögerter Selbsthemmung (delayed self-inhibition) erreicht. Durch die Selbsthemmung kann der vormals stationäre Aktivitätsfleck nicht mehr an einem Ort stehen bleiben, sondern er muß ständig auf benachbarte Bereiche ausweichen, da dort die Selbsthemmung noch gering ist. Das führt zu einer kontinuierlichen Bewegung, in der der Aktivitätsfleck die gesamte neuronale Schicht abtastet (siehe Abbildung 4.2). Diese starke Eigendynamik der Aktivitätsflecken führt natürlich auch zu Problemen. Insbesondere ist die Synchronisation der Aktivitätsflecken auf verschieden großen Schichten erschwert. Daher habe ich einen Aufmerksamkeitsfleck (attention blob) eingeführt, der die Bewegungsfreiheit des laufenden Flecks auf der größeren Schicht einschränkt. Der Aufmerksamkeitsfleck kann seinerseits aber auch von dem Aktivitätsflecken verschoben werden, z.B. in den Bereich des abgebildeten Objektes (siehe Abbildungung 4.4 und 4.5). Die dynamische Graphenanpassung geschieht parallel zwischen dem Bild und einer Galerie von Modellen. Zur eigentlichen Erkennung des richtigen Modells wird dessen Gesamtaktivität verwendet. Das richtige Modell ist dem Bild am ähnlichsten und kooperiert daher am erfolgreichsten, was zu einer erhöhten Gesamtaktivität führt. In einfachen Fällen kann schon sehr früh das richtige Modell bestimmt werden (siehe Abbildung 4.6 oben). In anderen Fällen müssen sich die Verbindungsmatrizen erst stark organisiert haben, bevor sich das richtige Modell durchsetzen kann (siehe Abbildung 4.6 unten). Erkennungsleistungen unter verschiedenen Bedingungen für Galerien von bis zu 111 Modellen sind in Tabelle 4.3 angegeben.

Drei weitere Veränderungen gegenüber dem ursprünglichen System sind von Bedeutung: Erstens wurden die Schichten wechselseitig, anstatt wie bisher unidirektional, miteinander verbunden. Das hat zum einen den Vorteil, daß sich die Aktivitätsflecken leichter synchronisieren. Zum anderen ist das für das Erkennungssystem notwendig: Die Modelle müssen das Bild beeinflussen, um den Aufmerksamkeitsfleck richtig auf dem Objekt zu positionieren. Das Bild muß die Modelle anregen, um eine Unterscheidung zwischen ähnlichen und unähnlichen Modellen zu erlauben. Zweitens wurden die Modelle untereinander derartig verknüpft, daß die Aktivitätsflecken in allen Modellen immer synchron laufen und zu einem Zeitpunkt am gleichen Ort im Gesicht sind, z.B. das rechte Auge oder die Nasenspitze. Diese Struktur kommt dem allgemeinen Gesichtswissen schon sehr nahe. Ohne diese Zwangssynchronisation in der Modelldomäne würde die Synchronisation des Bildes mit den Modellen weitgehend von zufälligen Anfangsbedingungen abhängen. Drittens werden die Neuronen nicht, wie üblich, durch die Summe der eingehenden Signale angeregt, sondern durch das Maximum. Die Summe vermischt nämlich ein korrektes Signal mit vielen falschen, während das Maximum mit einer relativ hohen Wahrscheinlichkeit das richtige Signal und nur dieses selektiert. Ein weiterer Vorteil des Maximums ist, daß der dynamische Bereich der äußeren Anregung während des Selbstorganisationsprozesses gleich bleibt. Es müssen also keine Parameter nachgeregelt werden.

B.5 Gesichtserkennung mit elastischer Graphenanpassung

Zur dynamischen Graphenanpassung gibt es eine algorithmisch ausgerichtete Variante, die elastische Graphenanpassung (elastic graph matching). Sie ist sehr viel schneller und flexibler als ihr neuronales Gegenstück und daher angemessener für technische Anwendungen. Hier ist die Aufgabe wieder Gesichtserkennung. Anstatt durch neuronale Schichten werden die Gesichter direkt durch etikettierte Graphen repräsentiert, die Knoten werden mit Jets als lokalen Merkmalen etikettiert, und die Kanten tragen Informationen über den Abstand der verbundenen Knoten (siehe Abbildung 5.1). Es werden zwei Prozesse unterschieden, das Bilden eines neuen Gesichtsgraphen durch elastische Graphenanpassung und die eigentliche Gesichtserkennung, bei der der neue Gesichtsgraph mit einer Galerie von Modellgraphen verglichen wird. Das letztere geschieht einfach aufgrund der gemittelten Ähnlichkeit korrespondierender Jets. Die Geometrie der Graphen spielt dabei keine Rolle. Die elastische Graphenanpassung dagegen beruht auf einem relativ aufwendigen Optimierungsprozeß, in dem versucht wird, denjenigen Teilgraphen aus einem Bild auszuwählen, der eine möglichst hohe Jetähnlichkeit mit den Modellgraphen hat unter der Nebenbedingung, daß der Bildgraph geometrisch nicht zu stark verzerrt sein darf gegenüber der mittleren Geometrie der Modellgraphen. Um eine möglichst hohe Präzision zu erreichen, wird für die Graphenanpassung die Jet-Vergleichsfunktion unter Berücksichtigung der Phaseninformation verwendet.

In der elastischen Graphenanpassung spielt das allgemeine Gesichtswissen eine besondere Rolle. In dieser Graphenstruktur ist das allgemeine Wissen des Systems um die verschiedenen möglichen Erscheinungsformen von Gesichtern zusammengefaßt (siehe Abbildung 5.2). Die Kanten sind wieder mit Abstandsinformationen etikettiert, jedoch gemittelt über alle Modelle des allgemeinen Gesichtswissens. Den Knoten sind die Jets aller Modelle zugeordnet. Dabei entspricht jeder Knoten einem bestimmten Punkt im Gesicht, z.B. der Nase, dem linken Auge, oder einem Mundwinkel. Bei der elastischen Graphenanpassung kann jeweils ein Jet pro Knoten angesprochen werden, und man wählt jeweils den am besten passenden aus. Auf diese Weise kann die volle kombinatorische Vielfalt des allgemeinen Gesichtswissens genutzt werden. Ergebnisse der elastischen Graphenanpassung sind in Abbildung 5.3 gezeigt.

Das System ist auch in der Lage, Gesichter in sehr verschiedener Ansicht miteinander zu vergleichen, z.B. Frontalansicht mit Halbprofil. Dazu ist es notwendig, objektangepaßte Graphen (object-adapted graphs) zu definieren. Die Knoten beziehen sich auf gleiche Punkte im Gesicht, unabhängig von der Ansicht. So gibt es in jeder Ansicht Augenknoten, Nasenknoten, usw. Die Struktur der Graphen sowie die Korrespondenzen zwischen Knoten, die zu gleichen Punkten im Gesicht gehören, wurden per Hand definiert. Im Vergleich zweier Gesichter verschiedener Pose werden dann natürlich nur korrespondierende Knoten verglichen. Die Erkennungsraten liegen jedoch für unterschiedliche Posen relativ niedrig, wie beim Menschen auch (siehe Tabelle 5.1). Zur Beurteilung, ob ein Gesicht zuverlässig erkannt wurde, habe ich ein schon früher entwickeltes Konfidenzmaß (confidence measure) verwendet.

Die drei wesentlichen Neuerungen gegenüber dem vorangegangenen System sind die Verwendung von Phaseninformation bei der Graphenanpassung, die Einführung des allgemeinen Gesichtswissens, und die Verwendung von objektangepaßten Graphen zum Vergleich von Gesichtern verschiedener Pose.

B.6 Phantombilder und Bestimmung von Gesichtsmerkmalen

Das Ergebnis der elastischen Graphenanpassung ist nicht nur der Bildgraph, sondern auch die Information, für welchen Knoten welcher Jet und damit auch welches Modell des allgemeinen Gesichtswissens am besten paßt. Diese Information soll nun für eine weitergehende Analyse eines Gesichtes genutzt werden. Zunächst kann man ein Phantombild erzeugen. Dazu werden die lokalen Grauwertverteilungen, die zu den ausgewählten Jets gehören, mit weichen Übergängen aneinandergefügt. Es wird also keinerlei Grauwertinformation des Originalbildes verwendet. Die Phantombilder sehen den Originalen recht ähnlich (siehe Abbildung 6.2). Man kann also davon ausgehen, daß für ein weibliches Gesicht die elastische Graphenanpassung im wesentlichen Jets von weiblichen Modellen auswählt. Gleiches gilt für männliche Gesichter oder Gesichter mit Brille oder Bart (bei letzteren aber nur für die oberen bzw. unteren Knoten). Die Merkmale der Modelle lassen sich so auf das Originalgesicht übertragen und dessen Merkmale damit ermitteln. Das Prinzip ist in Abbildung 6.1 illustriert. Die Erkennungsrate für die Gesichtsmerkmale Geschlecht, Brille und Bart sind in Tabelle 6.2 angegeben. Anwendung der Bayesschen Formel gibt außerdem einen Hinweis darauf, welche Knoten für die Merkmalsbestimmungen besonders zuverlässig sind (siehe Abbildung 6.3).

Die Erkennungsraten sind etwas niedriger bis vergleichbar mit anderen neuronalen Modellen zur Bestimmung des Geschlechtes eines Gesichts. Jedoch hat die vorgestellte Methode einige grundlegende Vorteile. Sie ist sehr allgemein, erfordert also keine manuelle Definition von Merkmalen, die für die Aufgabe geeignet sind. Das System sollte ohne weitere Modifikationen auf andere Aufgaben übertragbar sein, wie z.B. die Bestimmung von emotionalen Gesichtsausdrücken oder die Unterscheidung verschiedener Haustierrassen (Hund, Katze, Schaf). Bedingung ist nur eine in Bezug auf Gestalt und Pose konsistente Darstellung der zu bestimmenden Objekte. Das System erfordert außerdem nur ein Minimum an Trainingsaufwand. Die alternativen Modelle, wie z.B. Backpropagation oder Systeme basierend auf einer Hauptachsentransformation, sind bekannt für ihren großen Trainingsaufwand, sowohl in bezug auf die Anzahl der Trainingsbeispiele als auch in bezug auf die Rechenzeiterfordernisse.

B.7 Erkennung von teilverdeckten Objekten

Die Anwendungen der vorangegangenen drei Abschnitte beziehen sich auf Gesichtserkennung. In diesem Abschnitt ist die Aufgabe eine ganz andere. Verschiedene Spielzeugobjekte werden zu Szenen zusammengestellt und können sich dabei weitgehend überdecken (siehe Abbildung 7.1). Das vorgestellte System soll die Objekte trotz der Verdeckungen erkennen. Hier erweist sich die Repräsentation der Objekte durch etikettierte Graphen als besonders vorteilhaft, da sie in natürlicher Weise erlaubt, verschiedene Teile eines Objektes verschieden zu behandeln. Die Knoten müßen lediglich um eine Statusvariable ergänzt werden, die bezeichnet, ob der Knoten als sichtbar oder verdeckt angenommen werden soll. Auch kann für zwei überlappende Graphen gezielt bestimmt werden, welcher im Überlappbereich besser in das Bild paßt. Danach richtet sich die Hypothese, welches der beiden Objekt als verdeckt angenommen wird.

Es werden zwei Algorithmen zur Objekterkennung in einer Szene vorgestellt. Beiden

geht die Graphenanpassung voraus, die für alle Objekte einer kleinen Galerie von 13 Objekten jeweils den wahrscheinlichsten Ort im Bild ermittelt. Der erste Algorithmus behandelt jeden Graphen einzeln und bestimmt aufgrund der Ähnlichkeiten der einzelnen Knoten mit dem Bild, welche Regionen des Graphen voraussichtlich verdeckt und welche sichtbar sind. Ist die Gesamtähnlichkeit zu gering oder der als sichtbar angenommene Bereich zu klein, so wird das Modell ganz verworfen. Dieser Algorithmus erreicht eine Erkennungsrate von 80% und ist geeignet, wenn bekannte Objekte unter unbekannten Objekten erkannt werden soll (siehe Abbildung 7.3). Kann man voraussetzen, daß alle Objekte der Szene bekannt sind, so kann man die Szene von vorne nach hinten abarbeiten. Das hat den Vorteil, daß die vorderen Objekte sicher nicht verdeckt sind, und daß für weiter hinten liegende Objekte genau bekannt ist, welche Bildbereiche schon durch Objekte im Vordergrund besetzt sind. Die Erkennungsrate ist mit 96.7% entsprechend höher (siehe Abbildung 7.4).

B.8 Diskussion

Absicht der vorliegenden Arbeit war es, zu demonstrieren, daß der etikettierte Graph ein leistungsfähiges und flexibles Datenformat ist, das die syntaktische Struktur beinhaltet, die der Vektorrepräsentation fehlt. Die Knotenrelationen spielten in den verschiedenen Anwendungen eine wichtige Rolle. Sie waren entweder explizit oder implizit in den jeweiligen Datenstrukturen realisiert.

Erstens wurden die individuellen Koeffizienten der Gabor Wavelet-Transformation zu Jets zusammengebunden. Zweitens wurden Knoten durch Kanten zu Graphen organisiert. Drittens arbeiten sowohl die dynamische als auch die elastische Graphenanpassung mit Verbindungen zwischen einem oder mehreren Modellgraphen und einem Bild, um einen neuen Bildgraphen zu generieren. (Diese Aspekte syntaktischer Verbindungen sind schon in früheren Arbeiten verwendet worden. In dieser Arbeit neu hinzugekommen sind die folgenden.) Viertens, eine Menge von Jets, die als Alternativen für ein und denselben Objektpunkt fungieren können, wurden im allgemeinen Gesichtswissen gemeinsam an einen Knoten gebunden. Fünftens wurde das allgemeine Gesichtswissen um Kontextinformation erweitert, die Jets gleichen abstrakten Merkmals miteinander verbindet. Sechstens schließlich gab es zwischen den Knoten von Objektgraphen inhibitorische Verbindungen, wenn die Knoten um den gleichen Bildbereich konkurrierten. Dies ist nur eine relativ kleine Zahl von möglichen Beziehungen zwischen Knoten. Man wird weitere entwickeln und Wege finden müssen, wie sich die Relationen geeignet selbstorganisieren können.

Das zweite Anliegen der Arbeit war, zu zeigen, daß die dynamische Graphenanpassung ein leistungsfähiges neuronales Konzept zur Verarbeitung von etikettierten Graphen darstellt. Zum erstenmal wurde auf dieser Basis ein vollständiges Erkennungssystem entwickelt, das in der Lage ist, Gesichter gegen eine Galerie von über hundert Modellen zu erkennen. Die elastische Graphenanpassung ist aber immer noch deutlich langsamer und unflexibler als die elastische Graphenanpassung und viele Apekte der vorgestellten Anwendungen müßen noch in neuronalem Stile entwickelt werden.

B.9 Anhang A: Visuelle Vorverarbeitung mit Gabor Wavelets

Gabor-Funktionen haben die Form von Wellenpaketen (wavelet): ebene Wellen unter einer einhüllenden Gaußglocke. In der Gabor-Wavelettransformation wird ein Bild mit einer ganzen Familie von Gabor Funktionen gefaltet. Die Gabor Kerne haben alle die gleiche Form und unterscheiden sich nur in Größe und Orientierung. In der vorliegenden Arbeit werden fünf Größen (Frequenzen) und acht Orientierungen, d.h. 40 Kerne, verwendet. Das Ergebnis sind 40 komplexe Koeffizienten an jedem Pixel des Bildes. Da die Kerne wellenartig sind, können den Koeffizienten Amplitude und Phase zugeschrieben werden. Die Amplitude ändert sich nur langsam mit dem Ort, die Phase variiert mit der Raumfrequenz der Welle. Die Koeffizienten eines Pixels werden zusammenfassend als Jet bezeichnet. Ein Jet ist eine kompakte und flexible Beschreibung einer Grauwertumgebung (siehe Abbildung A.1).

Jets werden in zweierlei Hinsicht ausgewertet. Erstens kann die Ähnlichkeit zwischen zwei Jets bestimmt werden. Dazu dient das normierte Skalarprodukt, ohne oder mit Berücksichtigung der Phaseninformation. Zweitens kann die Phaseninformation verwendet werden, um den räumlichen Abstand zweier Jets an Nachbarpunkten eines Objektes abzuschätzen. Dies ist für Stereobilder als Disparitätsschätzung bekannt.

Bibliography

Amari, S. (1977). Dynamics of pattern formation in lateral-inhibition type neural fields. *Biological Cybernetics*, 27:77–87.

Amari, S. (1980). Topographic organization of nerve fields. *Bulletin of Mathematical Biology*, 42:339–364.

Amari, S. (1989). Dynamical stability of formation of cortical maps. In Arbib, M. A. and Amari, S., editors, *Dynamic Interactions in Neural Networks: Models and Data*. Springer-Verlag, New York.

Anderson, C. H. and van Essen, D. C. (1993). Dynamic neural routing circuits. In Brogan, D., Gale, A., and Carr, K., editors, *Visual Search 2*. Taylor & Francis, London.

Aviitzhak, H., Diep, T., and Garland, H. (1995). High-accuracy optical character-recognition using neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(2):218–224.

Behrmann, K.-O. (1993). Leistungsuntersuchungen des "Dynamischen Link-Matchings" und Vergleich mit dem Kohonen-Algorithmus. Diploma thesis, internal report IR-INI 93–05, Fakultät für Physik und Astronomie, Ruhr-Universität Bochum, D-44780 Bochum.

Bienenstock, E. and Doursat, R. (1991). Issues of representation in neural networks. In Gorea, A., editor, *Representations of Vision: Trends and Tacit Assumptions in Vision Research*, pages 47–67. Cambridge University Press, Cambridge.

Bienenstock, E. and von der Malsburg, C. (1987). A neural network for invariant pattern recognition. *Europhysics Letters*, 4:121–126.

Block, H. (1962). The perceptron: a model for brain functioning. i. *Reviews of Modern Physics*, 34:123–135. Also appeared in *Neurocomputing*, J.A. Anderson and E. Rosenfeld, Eds., MIT Press, Massachusetts, pp. 138–150.

Boff, K. R., Kaufman, L., and Thomas, J. P., editors (1986). *Handbook of Perception and Human Performance*. John Wiley and Sons, New York.

Bruce, V., Valentine, T., and Baddeley, A. (1987). The basis of the 3/4 view advantage in face recognition. *Applied Cognitive Psychology*, 1:109–120.

Brunelli, R. and Poggio, T. (1993a). Caricatural effects in automated face perception. *Biological Cybernetics*, 69:235–241.

Brunelli, R. and Poggio, T. (1993b). Face recognition: Features versus templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(10):1042–1052.

Buhmann, J., Lange, J., and von der Malsburg, C. (1989). Distortion invariant object recognition by matching hierarchically labeled graphs. In *IJCNN International Conference on Neural Networks, Washington*, pages 155–159. IEEE.

Buhmann, J., Lange, J., von der Malsburg, C., Vorbrüggen, J. C., and Würtz, R. P. (1992). Object recognition with Gabor functions in the dynamic link architecture: Parallel implementation on a transputer network. In Kosko, B., editor, *Neural Networks for Signal Processing*, pages 121–159. Prentice Hall, Englewood Cliffs, NJ 07632.

Chalmers, D. J., French, R. M., and Hofstadter, D. R. (1992). High-level perception, representation, and analogy: A critique of artificial intelligence methodology. *Journal of Experimental and Theoretical Artificial Intelligence*, 4:185–211.

de Edson, C., Filho, B., and Bisset, D. L. (1990). Applying the ART1 architecture to a pattern recognition task. In Eckmiller, R., Hartmann, G., and Hauske, G., editors, *Parallel Processing in Neural Systems and Computers*, pages 343–349. North-Holland, Amsterdam.

DeValois, R. and DeValois, K. (1988). *Spatial Vision*. Oxford Press.

Ehrig, H., editor (1991). *Graphs grammars and their application to computer science: 4th international workshop, Bremen, Germany, March 5–9,1990*, Berlin. Springer.

Eigen, M. (1978). The hypercycle. *Naturwissenschaften*, 65:7–41.

Fleet, D. J. and Jepson, A. D. (1990). Computation of component image velocity from local phase information. *International Journal of Computer Vision*, 5(1):77–104.

Fu, K. S. (1982). *Syntactic Pattern Recognition and Applications*. Prentice Hall, Englewood Cliffs, NJ 07632.

Fukushima, K., Miyake, S., and Ito, T. (1983). Neocognitron: a neural network model for a mechanism of visual pattern recognition. *IEEE Transactions on Systems, Man, and Cybernetics*, 13:826–834. Also appeared in *Neurocomputing*, J.A. Anderson and E. Rosenfeld, Eds., MIT Press, Massachusetts, pp. 526–534.

Golomb, B., Lawrence, D., and Sejnowski, T. (1991). Sexnet: a neural network identifies sex from human faces. In Touretzky, D. and Lippman, R., editors, *Advances in Neural Information Processing Systems 3*. Morgan Kaufmann, SanMateo, CA.

Häussler, A. and von der Malsburg, C. (1983). Development of retinotopic projections — an analytical treatment. *J. Theor. Neurobiol.*, 2:47–73.

Jones, J. and Palmer, L. (1987). An evaluation of the two dimensional Gabor filter model of simple receptive fields in cat striate cortex. *J. of Neurophysiology*, 58:1233–1258.

Kalocsai, P., Biederman, I., and Cooper, E. E. (1994). To what extent can the recognition of unfamiliar faces be accounted for by a representation of the direct output of simple cells. In *Proceedings of the Association for Research in Vision and Ophtalmology, ARVO*, Sarasota, Florida.

Kidder, J. N. and Seligson, D. (1993). Fast recognition of noisy digits. *Neural Computation*, 5(6):885–892.

Kirby, M. and Sirovich, L. (1990). Application of the Karhunen-Loève procedure for the characterization of human faces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(1):103–108.

Kohonen, T. (1972). Correlation matrix memories. *IEEE Transactions on Computers*, C-21:353–359. Also appeared in *Neurocomputing*, J.A. Anderson and E. Rosenfeld, Eds., MIT Press, Massachusetts, pp. 174–180.

Kohonen, T. (1982). Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, 43:59–69.

Kohonen, T. (1987). Adaptive, associative, and self-organizing functions in neural computing. *Applied Optics*, 26(23):4910–4918.

Konen, W., Maurer, T., and von der Malsburg, C. (1994). A fast dynamic link matching algorithm for invariant pattern recognition. *Neural Networks*, 7(6/7):1019–1030.

Konen, W. and von der Malsburg, C. (1992). Unsupervised symmetry detection: A network which learns from single examples. *Artificial Neural Networks*, 2:121–125.

Konen, W. and von der Malsburg, C. (1993). Learning to generalize from single examples in the dynamic link architecture. *Neural Computation*, 5:719–735.

Konen, W. and Vorbrüggen, J. C. (1993). Applying dynamic link matching to object recognition in real world images. In Gielen, S. and Kappen, B., editors, *Proceedings of the International Conference on Artificial Neural Networks, ICANN*, pages 982–985, London. Springer-Verlag.

König, P. and Schillen, T. B. (1991). Stimulus-dependent assembly formation of oscillatory responses: I. synchronization. *Neural Computation*, 3:155–166.

Kosko, B. (1987). Adaptive bidirectional associative memories. *Applied Optics*, 26(23):4947–4960.

Krüger, N. (1994). personal communication.

Krüger, N. (1995). Learning weights in discrimination functions using a priori constraints. Accepted for 17. Symposium der deutschen Arbeitsgemeinschaft für Mustererkennung (DAGM).

Lades, M., Vorbrüggen, J. C., Buhmann, J., Lange, J., von der Malsburg, C., Würtz, R. P., and Konen, W. (1993). Distortion invariant object recognition in the dynamic link architecture. *IEEE Transactions on Computers*, 42(3):300–311.

Lanitis, A., Taylor, C., and Cootes, T. (1995). An automatic face identification system using flexible appearance models. *Image and Vision Computing*, 13(5):393–401.

LeCun, Y., Boser, B., Denker, J., Henderson, D., Howard, R., Hubbard, W., and Jackel, L. (1989). Backpropagation applied to handwritten Zip code recognition. *Neural Computation*, 1(4):541–551.

Manjunath, B., Chellappa, R., and von der Malsburg, C. (1992). A feature based approach to face recognition. Technical Report CAR-TR-604 or CS-TR-2834, Computer Vision Laboratory, University of Maryland, Colledge Park, MD 20742–3411.

Mao, M. W. and Kuo, J. B. (1992). A coded block adaptive neural network system with a radical-partitioned structure for large-volume chinese characters recognition. *Neural Networks*, 5:835–841.

Martin, G. L. (1993). Centered-object integrated segmentation and recognition of overlapping handprinted characters. *Neural Computation*, 5(3):419–429.

Maurer, T. and von der Malsburg, C. (1995). Single-view based recognition of faces rotated in depth. In *Proceedings of the International Workshop on Automatic Face- and Gesture-Recognition*, Zürich.

Olshausen, B. A. (1994). *Neural Circuits for Forming Invariant Representations of Visual Objects*. PhD thesis, California Institute of Technology, Pasadena, California.

Oram, M. W. and Perrett, D. I. (1994). Modeling visual recognition from neurobiological constraints. *Neural Networks*, 7(6/7):945–972.

O'Toole, A., Abdi, H., Deffenbacher, K., and Valentin, D. (1993). Low-dimensional representation of faces in higher dimensions of the face space. *Journal of the Optical Society of America A*, 10(3):405–411.

Pollen, D. A. and Ronner, S. F. (1981). Phase relationship between adjacent simple cells in the visual cortex. *Science*, 212:1409–1411.

Pöppel, E. (78). Time perception. In Held, R., Leibowitz, H., and Teuber, H.-L., editors, *Perception*, chapter 23, pages 713–729. Springer, Berlin Heidelberg.

Pötzsch, M. (1994). Die Behandlung der Wavelet-Transformation von Bildern in der Nähe von Objektkanten. Diploma thesis, Fachbereich Physik, Universität Dortmund. Also published as internal report IR-INI 94-04 at the Institut für Neuroinformatik, Ruhr-Universität Bochum.

Reiser, K. (1991). Learning persistent structure. Doctoral thesis, Res. Report 584, Hughes Aircraft Co., 3011 Malibu Canyon Rd. Malibu, CA 90265.

Rinne, M. (1995). Matchen von Kantenbildern mit einem dynamischen Neuronennetz. Diploma thesis, internal report IR-INI 95-06, Institut für Neuroinformatik, Ruhr-Universität Bochum, D-44780 Bochum.

Samal, A. and Iyengar, P. A. (1992). Automatic recognition and analysis of human faces and facial expressions: A survey. *Pattern Recognition*, 25(1):65–77.

Schwarz, A. (1995). Erkennung von Objekten in Grauwertbildern mit laufenden nichtlinearen Wellen. Diploma thesis, Fakultät für Physik und Astronomie, Ruhr-Universität Bochum, D-44780 Bochum.

Sejnowski, T. J., Kienker, P. K., and Hinton, G. E. (1986). Learning symmetry group with hidden units: Beyond the perceptron. *Physica*, 22D:260–275.

Sirovich, L. and Kirby, M. (1987). Low-dimensional procedure for the characterization of human faces. *Journal of the Optical Society of America A*, 4(3):519–524.

Subramaniam, S., Biederman, I., Kalocsai, P., and Madigan, S. (1995). Accurate identification, but chance forced-choice recognition for rsvp pictures. In *Proceedings of the Association for Research in Vision and Ophtalmology, ARVO*, Ft. Lauderdale, Florida.

Theimer, W. M. and Mallot, H. A. (1994). Phase-based binocular vergence control and depth reconstruction using active vision. *CVGIP: Image Understanding*, 60(3):343–358.

Ting, C. and Chuang, K.-C. (1993). An adaptive algorithm for neocognitron to recognize analog images. *Neural Networks*, 6(2):285–299.

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

Valentin, D., Abdi, H., O'Toole, A. J., and Cottrell, G. W. (1994). Connectionist models of face processing: A survey. *Pattern Recognition*, 27(9):1209–1230.

van Essen, D. C., Anderson, C. H., and Olshausen, B. A. (1994). Dynamic routing strategies in sensory, motor, and cognitive processing. In Koch, C. and Davis, J., editors, *Large Scale Neural Theories of the Brain*. MIT Press.

von der Malsburg, C. (1973). Self-organization of orientation sensitive cells in the striate cortex. *Kybernetik*, 14:85–100.

von der Malsburg, C. (1979). Development of ocularity domains and growth behaviour of axon terminals. *Biological Cybernetics*, 32:49–62.

von der Malsburg, C. (1981). The correlation theory of brain function. Internal report, 81-2, Max-Planck-Institut für Biophysikalische Chemie, Postfach 2841, 3400 Göttingen, FRG.

VON DER MALSBURG, C. (1983). How are nervous structures organized? In BAŞAR, E., FLOHR, H., HAKEN, H., AND MANDELL, A., editors, *Synergetics of the Brain — Preceedings of the International Symposium on Synergetics*, pages 238–249. Springer, Berlin Heidelberg.

VON DER MALSBURG, C. (1985). Nervous structures with dynamical links. *Ber. Bunsenges. Phys. Chem.*, 89:703–710.

VON DER MALSBURG, C. (1986). Am I thinking assemblies? In PALM, G. AND AERTSEN, A., editors, *Proceedings of the Trieste Meeting on Brain Theory, October 1984*, pages 161–176. Springer, Berlin Heidelberg.

VON DER MALSBURG, C. (1988). Pattern recognition by labeled graph matching. *Neural Networks*, 1:141–148.

VON DER MALSBURG, C. (1994). personal communication.

VON DER MALSBURG, C. AND BIENENSTOCK, E. (1987). A neural network for the retrieval of superimposed connection patterns. *Europhys. Lett.*, 3(11):1243–1249.

VON DER MALSBURG, C. AND BUHMANN, J. (1992). Sensory segmentation with coupled neural oscillators. *Biological Cybernetics*, 67:233–242.

VON DER MALSBURG, C. AND WILLSHAW, D. (1977). How to label nerve cells so that they can interconnect in an ordered fashion. *Proc. Natl. Acad. Sci. (USA)*, 74:5176–5178.

VORBRÜGGEN, J. C. (1994). *Zwei Modelle zur datengetriebenen Segmentierung visueller Daten*, volume 47 of *Reihe Physik*. Verlag Harri Deutsch, Frankfurt a. Main. PhD thesis.

WAGNER, B. AND VON DER MALSBURG, C. (1995). Stability analysis of a two-dimensional self-organizing retinotopy system. in preparation.

WANG, D., HAASE, H., AND VON DER MALSBURG, C. (1990). A framework of knowledge representation in the brain based on the dynamic link structure. unpublished manuscript.

WIDROW, B. AND HOFF, M. E. (1960). Adaptive switching circuits. In *1960 IRE WESCON Convention Record*, pages 96–104. IRE. Also appeared in *Neurocomputing*, J.A. Anderson and E. Rosenfeld, Eds., MIT Press, Massachusetts, pp. 126–134.

WIDROW, B., WINTER, R., AND BAXTER, R. (1988). Layered neural nets for pattern recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 36(7):1109–1118.

WILLSHAW, D. AND VON DER MALSBURG, C. (1976). How patterned neural connections can be set up by self-organization. *Proc. R. Soc. London*, B194:431–445.

WILLSHAW, D. AND VON DER MALSBURG, C. (1979). A marker induction mechanism for the establishment of ordered neural mappings; its application to the retinotectal problem. *Trans. R. Soc. London*, B287:203–243.

Wiskott, L., Fellous, J.-M., Krüger, N., and von der Malsburg, C. (1995). Face recognition and gender determination. In *Proceedings of the International Workshop on Automatic Face- and Gesture-Recognition, IWAFGR 95*, pages 92–97, Zurich.

Wiskott, L. and von der Malsburg, C. (1993). A neural system for the recognition of partially occluded objects in cluttered scenes. *Int. J. of Pattern Recognition and Artificial Intelligence*, 7(4):935–948. Also appeared in *Advances in Pattern Recognition Systems using Neural Networks Technologies*, Series in Machine Perception and Artificial Intelligence, Vol. 7, Eds. Guyon, I. and Wang, P.S.P., IJPRAI, World scientific, February 1994.

Würtz, R. P. (1995). *Multilayer Dynamic Link Networks for Establishing Image Point Correspondences and Visual Object Recognition*, volume 41 of *Reihe Physik*. Verlag Harri Deutsch, Frankfurt a. Main. PhD thesis.

Yuille, A. L. (1991). Deformable templates for face recognition. *Journal of Cognitive Neuroscience*, 3(1):59–70.

Index