Autonomous Robots 6, 203–221 (1999)

GripSee: A Gesture-Controlled Robot for Object Perception and Manipulation*

MARK BECKER, EFTHIMIA KEFALEA, ERIC MAËL, CHRISTOPH VON DER MALSBURG[†], MIKE PAGEL, JOCHEN TRIESCH, JAN C. VORBRÜGGEN, ROLF P. WÜRTZ AND STEFAN ZADEL
Institut für Neuroinformatik, Ruhr-Universität Bochum, D-44780 Bochum, Germany
http://www.neuroinformatik.ruhr-uni-bochum.de/ini/VDM/
Rolf.Wuertz@neuroinformatik.ruhr-uni-bochum.de

Abstract. We have designed a research platform for a perceptually guided robot, which also serves as a demonstrator for a coming generation of service robots. In order to operate semi-autonomously, these require a capacity for learning about their environment and tasks, and will have to interact directly with their human operators. Thus, they must be supplied with skills in the fields of human-computer interaction, vision, and manipulation. **GripSee** is able to autonomously grasp and manipulate objects on a table in front of it. The choice of object, the grip to be used, and the desired final position are indicated by an operator using hand gestures. Grasping is performed similar to human behavior: the object is first fixated, then its form, size, orientation, and position are determined, a grip is planned, and finally the object is grasped, moved to a new position, and released. As a final example for useful autonomous behavior we show how the calibration of the robot's image-to-world coordinate transform can be learned from experience, thus making detailed and unstable calibration of this important subsystem superfluous. The integration concepts developed at our institute have led to a flexible library of robot skills that can be easily recombined for a variety of useful behaviors.

Keywords: service robot, human-robot interaction, stereo vision, gesture recognition, hand tracking, object recognition, fixation, grasping, grip, perception, skill, behavior

1. Introduction

The majority of robots in use today perform only a very limited preordained set of actions in a highly structured and controlled environment. The next generation will need a much wider range of applicability, in particular in the form of service robots that can work in environments designed to be used by their owners, not themselves. Examples are offices, supermarkets, hospitals, and households; uncontrollable environments such as hazardous areas in technical systems or in space also call for robots that can cope with circumstances not tailored to their needs. Although some service robots for sweeping or vacuum cleaning, lawn mowing or drink dispensing are already on the market, they have a long way to go to be of practical use. Their limited range of capabilities is not a classical robotics problem, as huge progress has been made in, e.g., hardware and control software. This is not to deny the abundance of open problems in robot control, but there, at least, the problems are understood well enough to allow a mathematical formulation. What is really lacking is a convincing way of interaction with the environment. The major difficulty is neither the acquisition of information about the environment nor its manipulation—cameras, microphones, and tactile sensors on one hand and manipulators and speech synthesizers on the other have already reached high standards. Rather, it is the *interpretation* of the sensory data that poses the

*This work has been funded by grants from the DFG (Graduiertenkolleg KOGNET) and the German Federal Ministry for Education and Research (01 IN 504 E9).

[†]Also at Department of Computer Science, University of Southern California, Los Angeles, USA.

major obstacle for a robot to become truly *situated* (Suchmann, 1987) or *embedded* (Rosenschein, 1985), with all the theoretical implications (Maes, 1994). We claim that, currently, situatedness does not make much of a difference for a robot because the resulting information is simply not available to its control system. Even representations close to the raw sensory data, which have been proposed as a way out of this problem, need a level of reliability that eludes present methodology. In other words, *perception* remains the toughest problem for autonomous robots.

Following the idea of *emergent functionality*, another approach to hard problems in robotics, one has to cope with the functionality that a system provides without having a systematic way of producing a *desired* behavior. In contrast to that, our concept might be called *semi-autonomy*: a robot must dispose of a repertoire of *skills* that are carried out autonomously, the actual *control* of behavior must be left to a human operator to an appropriate degree. In our system, we have chosen to implement a gesture interface which is suited for use by technically untrained people, a decision dictated, of course, by the quest to provide a prototype for a service robot. The long-term goal of semi-autonomous robotics is the construction of an intelligent slave, although this might well turn out to be a contradiction in terms.

Returning to the problem of perception, we feel that nothing is wrong with internal symbolic representations if (and this is a big restriction) they can be constructed autonomously from rough, built-in concepts. The development of the field has shown that *building* representations is the hard part, while manipulation of and reasoning from a given representation are relatively straightforward. Chances are that novel paradigms from the repertoire of *soft computing* will be important to complement classical AI paradigms, which have their unquestioned success in processing symbolic information *once it is available*.

Thus, for theoretical reasons as well as for considerations about market demand, it appears safe to assume that the important breakthroughs in robotics will be the creation of a robot with serious perceptual capabilities. This paper introduces **GripSee**, a robot designed as a contribution towards that goal. Its environment is restricted to a table top where everyday objects can be manipulated and human operators can give commands to the robot by means of hand gestures. This choice is motivated by our focus on perception—in later stages we plan to enable **GripSee** to gradually learn representations of unknown objects by a combination of observation and manipulation, and with only a minimum of human intervention.

2. The Robot and Its Software

2.1. Design Principles

GripSee's design was motivated by the fact that visual perception can be supported to a large degree by active components: an active camera head can change the direction of view to avoid singular visual situations, a robot arm can manipulate the object to get a global impression of how it looks. Mobility would increase those possibilities, but it also causes so many new problems that we decided to leave it out initially.

As modeling visual perception almost invariably means modeling *human* visual perception, we have chosen to let the robot components, specifically the arrangement and kinematics of camera head and robot manipulator, closely resemble the human eye, head and arm arrangement (Fig. 1). This includes kinematic redundancy to enable obstacle avoidance and leave some flexibility in grasping and manipulation. This arrangement has the additional advantage that the arm can avoid occluding its own workspace. The camera system imitates primate eyes in supplying a stereo system with a "fovea" for each eye, i.e., an area of high resolution and color sensitivity, but necessarily small field of view for those visual tasks requiring high spatial precision. Each fovea is surrounded by a "periphery" with lower resolution, but a large field of view for tasks such as motion detection, obstacle avoidance, and saccade planning.

With regard to the control system, all individual robot components are controlled by autonomous agents that communicate with each other and with other agents that implement specific *skills*. Overall control is maintained by a separate agent that defines task timelines and contains a user interface. On the implementation level, each agent consists of one executable (see Section 2.3 for details and Fig. 2 for an overview of all components and their interactions).

2.2. Hardware

The robot hardware is shown in Fig. 1 and consists of the following components:

- A modular robot arm with seven degrees of freedom (DoF), kinematics similar to a human arm, and a parallel jaw gripper;

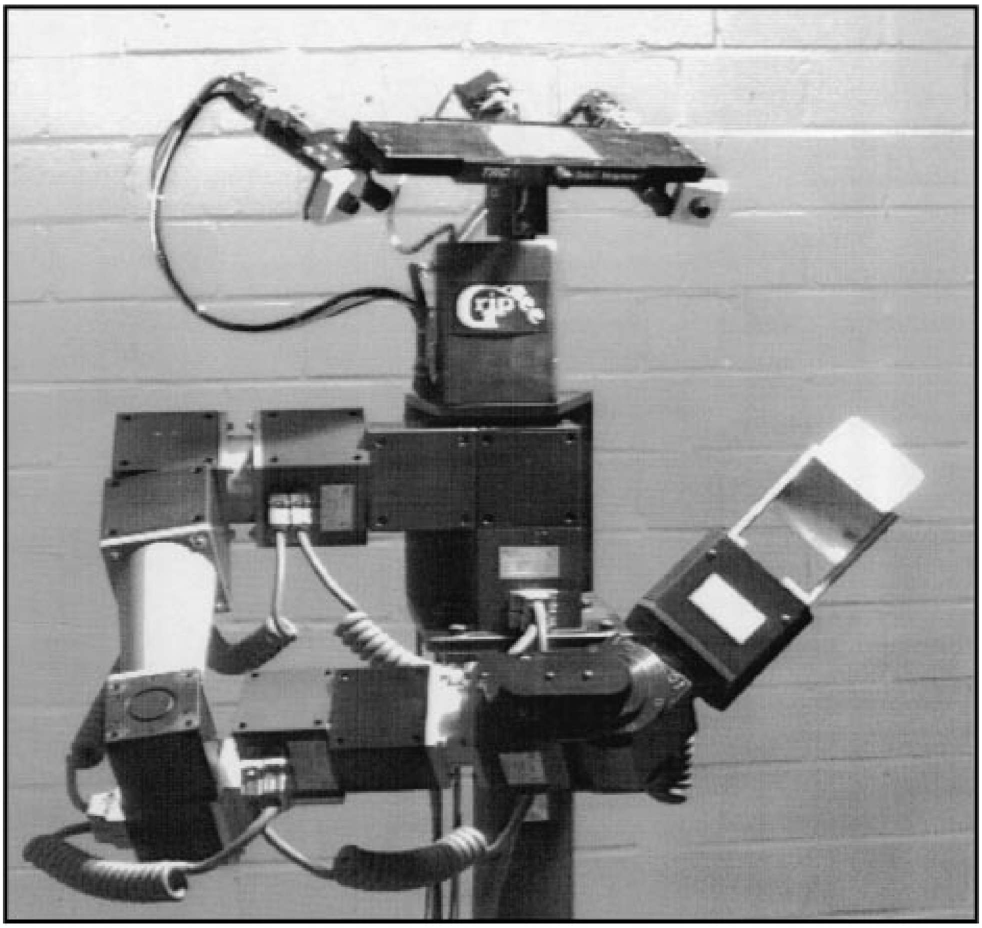


Figure 1. **GripSee**'s arm and camera head. The design closely resembles the structure of a human body.

- a dual stereo camera head with three DoF (pan, tilt, and vergence) and a stereo basis of 30 cm for two camera pairs with different fields of view (horizontally 56° with color and 90° monochrome, respectively);
- a computer network composed of two Pentium PCs under QNX and a Sun UltraSPARC II workstation under Solaris.

Image acquisition is done by two color framegrabber boards controlled by one of the PCs, which also controls the camera head and performs real-time image processing, e.g., hand tracking (see Section 3.1.1). The second PC controls the robot arm. Since image data has to be transferred between the processors, they are networked with FastEthernet to achieve sufficient throughput and low latencies. **GripSee**'s hardware is very similar to the one of **Arnold** (Bergener et al., 1997; Bergener and Dahm, 1997), which is used by another research group in our institute and is mounted on a mobile platform.

2.3. Software Structure

Our software is based on the C++-library *FLAVOR* (Flexible Library for Active Vision and Object Recognition) developed at our institute. FLAVOR comprises functionality for the administration of arbitrary images and other data types, libraries for image processing, object representation, graph matching, image segmentation, robotics, and interprocess communication. It supports flexibility, rapid prototyping, and safe and correct coding (Rinne et al., 1998).

As shown in Fig. 2, each component of the hardware (arm with gripper, camera head, image acquisition) is controlled by a separate server program that supports multiple clients. Each module performing an image

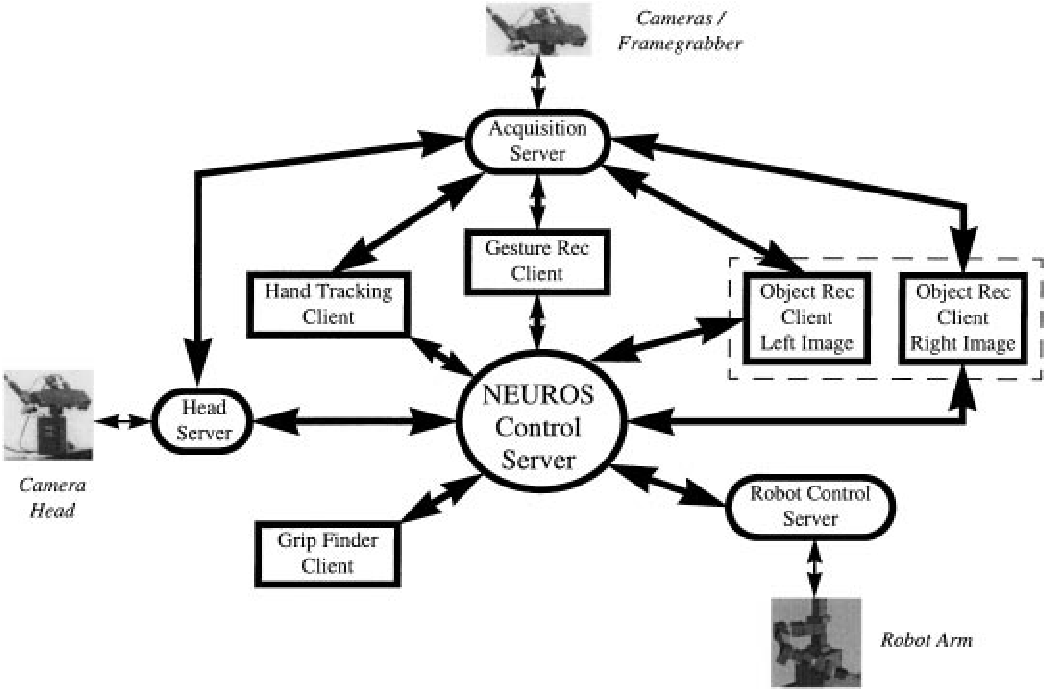


Figure 2. Modular structure of server and client processes, each of which implements an autonomous agent.

processing or robotics task (see Section 3) is implemented as a separate program. A main control program, the *NEUROS Control Server*, coordinates the separate processes distributed over the network. It implements the actual application (see Section 4) by defining the timeline of actions to be performed; it also provides the user interface to control the application and to display results.

3. Skills Currently Implemented

In this section, we describe the individual modules that currently comprise our system. In Section 4, we will show in several examples how those can be combined to yield useful behaviors. Results for the single skills and for the overall behavior will be presented in Section 5.

3.1. Human-Robot Interaction

A gesture interface for a service robot must meet at least two requirements. First, it must be person-independent, i.e., the robot must understand commands given by different persons. Second, it must be robust with respect to all the variation and background noise present in natural environments. The interface we have developed (Triesch and von der Malsburg, 1996, 1998) allows the operator to transmit commands to **GripSee** by performing hand gestures, e.g., by pointing at an object in a specific manner in order to have the robot pick up the object in a specific way. It consists of two agents. The first one tracks the operator's hand, the second one performs a refined analysis of the hand posture.

3.1.1. Hand Tracking. The hand tracking agent (see Fig. 3) integrates *motion detection*, *color analysis* and *stereo information*. By combining these cues the pitfalls inherent in each single one can be avoided to a large degree. The motion cue is based on difference images of subsequent frames. Smoothing and thresholding yield a binary image with pixels belonging to moving objects switched on. The color cue detects pixels of approximate skin color by comparison with a skin color histogram in HSI (hue, saturation, intensity) color space. The histogram can, in principle, be calibrated in advance for a particular person, but it usually suffices to use a non-restrictive default, which however must be adapted to the lighting conditions.

For each camera, we obtain an *attention map* by computing a weighted sum of the results of the motion and color cues and smoothing with a Gaussian. It highlights moving regions similar to skin color. For the

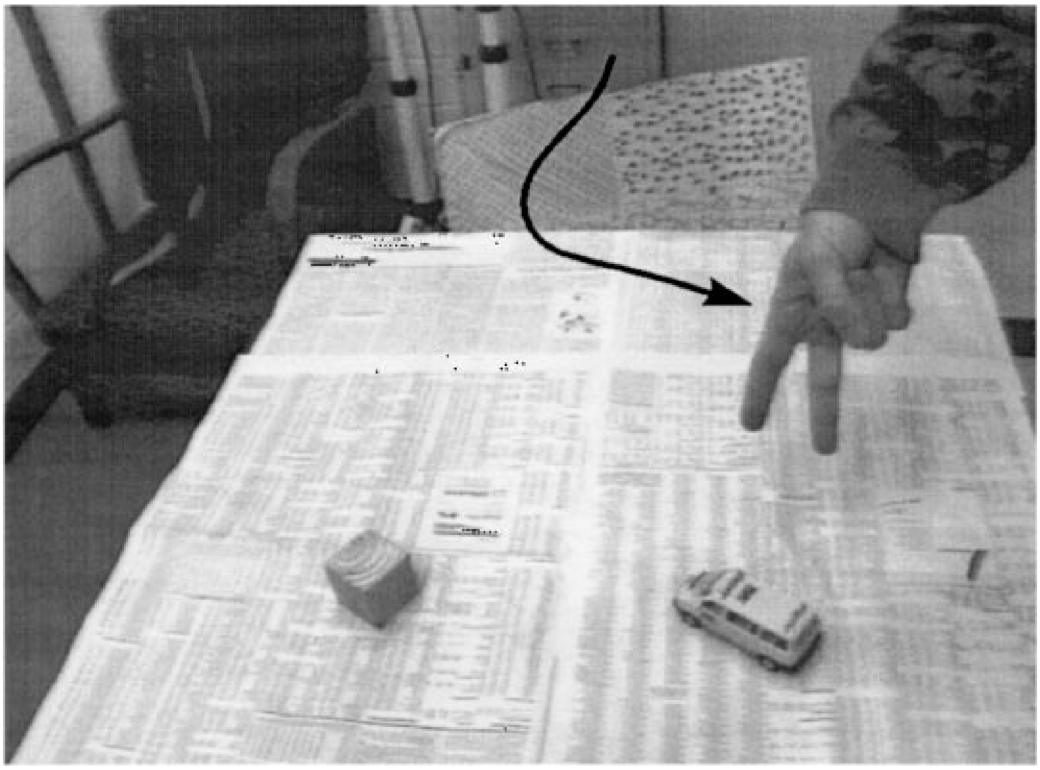


Figure 3. Tracking of the operator's hand relies on motion detection, skin color analysis and a stereo cue.

stereo cue, the attention maps of the left and right camera are added. In the resulting map objects in the plane of fixation are emphasized, since only these overlap. Their depth is computed by comparing the positions of local maxima in the left and right attention map. For the purpose of the example procedure described in Section 4, only the position of the object with *maximal* response is calculated. We are aware of the fact that much more sophisticated attention control methods are available, but at this point real time processing is crucial.

3.1.2. Hand Posture Analysis. When the moving hand stops its posture is analyzed by *elastic graph matching* (Lades et al., 1993; Wiskott, 1996; Wiskott et al., 1997). Different hand postures are represented as attributed graphs, whose nodes carry local image information in the form of responses to Gabor-based wavelets, while the edges (in the graph-theoretical sense) contain geometrical information (Fig. 4).

In previous work we have demonstrated that elastic graph matching can be successfully applied to the person-independent recognition of hand postures in front of complex backgrounds (Triesch and von der Malsburg, 1996). The system presented there was optimized for the robot application by using graphs with fewer and sparser nodes, as well as by reducing the number of allowed hand postures from ten to six (see Table 1 for results for both systems). Higher performance, which is necessary for truly reliable control, can be achieved by investing in longer processing times or more powerful hardware, especially for the wavelet convolution (Triesch and von der Malsburg, 1996).

Table 1. Results for gesture recognition. The first two rows show experiments with 10 gestures with different backgrounds. The bottom row shows parameter settings, which yield relatively fast matching and acceptable performance—those have been used for the overall behavior in Section 5.

No. of postures	Background	No. of tests	Percentage correct	Nodes per graph	Average matching time (s)
10	Complex	239	86.2	35	16
10	Simple	418	93.8	35	16
6	Table top	96	78.1	25	5

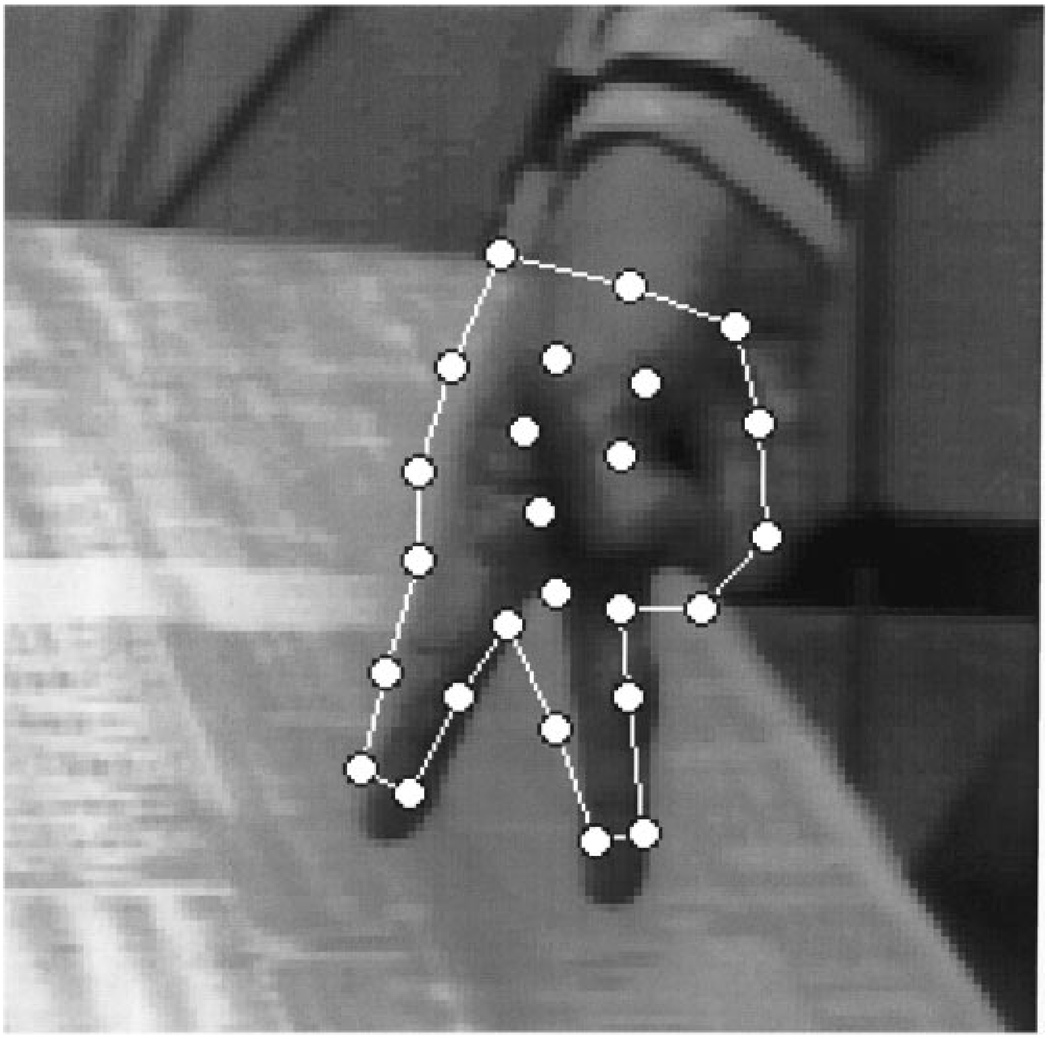


Figure 4. Successful matching of a model graph on the input image. Although not all nodes are positioned perfectly, the placing usually yields correct recognition.

3.2. Object Localization and Fixation

In Section 3.3 we will apply a modified version of elastic graph matching in the form of a high-level object recognition module based on object edges. That module works best if it is provided with a single object within a region of interest as small as possible. The localization module solves that preprocessing task by the heuristic that regions of high edge densities are candidates for objects. It will also become important for autonomous object learning because it only presupposes very general knowledge of the object, namely that it is rich in edges as compared to the background.

First, we obtain an edge description of the scene by employing the Mallat wavelet transform (Mallat and Zhong, 1992) as a multiresolution edge detector. Important features ("strong" edges) are enhanced using a dynamical thresholding operation and converted into binary format. Some results of the binarized edge description of a scene are shown in Fig. 5.

For a simple decision about the *focus of attention*, this edge map is low-pass filtered. The global maximum in each of the resulting images of a stereo pair becomes the first focus of attention. Its three-dimensional position is estimated by comparing the position of the global maxima in the left and right image. Both cameras then make a fixation movement to adjust their centers of view to that particular point (see Section 3.6.1 for details of the fixation process). The object is now centered in the images of both fovea cameras.

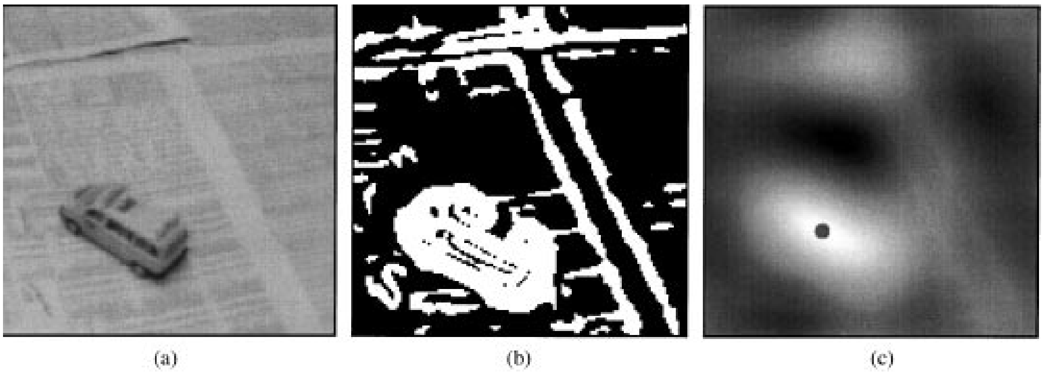


Figure 5. Preprocessing for the localization process. (a) Input picture from the left camera. (b) Feature extraction: thresholded and binarized edge description of the scene. (c) Extracted blobs of interest. The point indicates the pixel with maximal intensity, which becomes the point of fixation.

3.3. Object Recognition

After the fixation process, the object has to be classified in terms of its shape. Our approach to object classification relies upon object-adapted representations from different viewpoints. We deal with the depth rotation problem by using a multiview approach and call the resulting representation a *multigraph*. Each multigraph consists of a certain number of model graphs representing the same object from different viewpoints. We create a discretization of a unit sphere for an object as shown in Fig. 6 by placing the object on a rotating table and taking images after rotation by multiples of 9°. The nodes of the multigraphs are labeled with sets of local features called *jets*, which are vectors of responses of the Mallat wavelet transform (Mallot and Zhong, 1992). The graphs are constructed in the following way. First, a regular graph is positioned on the image, with nodes on a square lattice of pixels with a spacing of 4. Edges are introduced to each of the 8 neighbors of every node. This graph is then thinned by dropping nodes with transform values below threshold, which usually results in different connected components. From those components only the one with most nodes is kept. Finally, nodes with more than 6 neighbors are deleted, because they are likely to lie on a surface rather than a contour. The remaining edges are labeled with distance vectors between node position (see Kefalea et al., 1997).

Recognition is done by elastic graph matching, which compares stored multigraphs to the image in terms of similarities between stored jets and jets extracted from the image, adapting location and size of the model graphs until an optimum is found. In order to

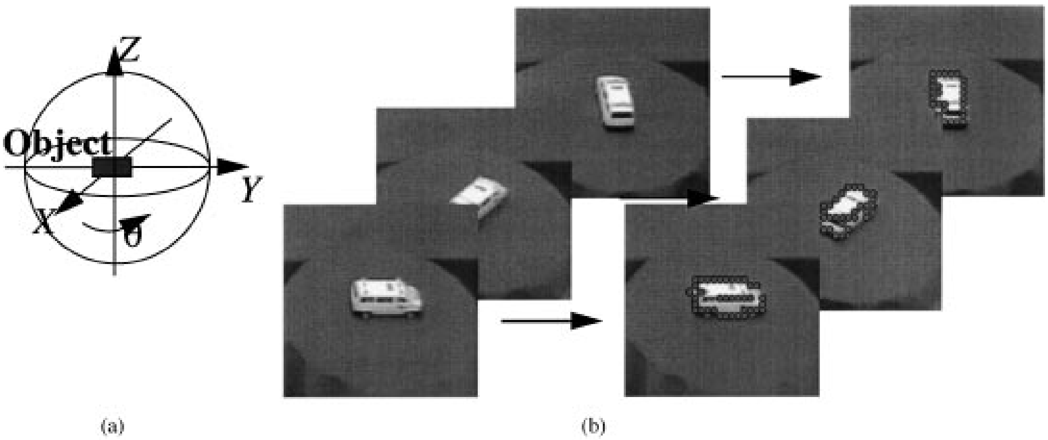


Figure 6. Gallery creation. (a) We sample a discrete unit sphere taking object images at steps of 9 degrees. (b) The corresponding model graphs constituting the multigraph of the object.

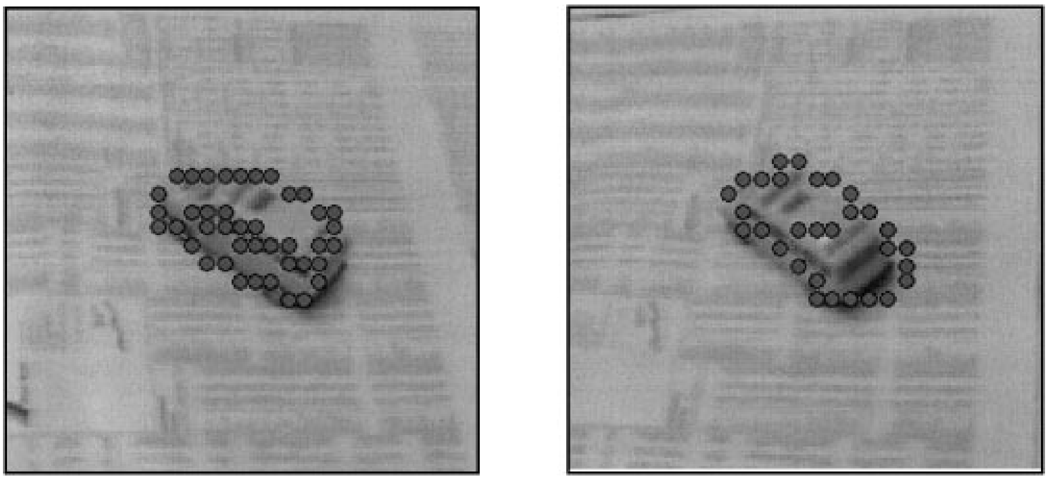


Figure 7. Results of a recognition process (for both left and right image). The superimposed graphs are the model graphs corresponding to the most similar object in the gallery.

speed up the process and to avoid local minima, matching proceeds in two steps. In the first step the graph remains undistorted. The object location corresponds to the position with maximal similarity between model graph and input image. In the second step the scale is adapted and the localization is improved. The graph from the first step is now allowed to vary in size by a common factor in the horizontal and vertical direction, and allowing a shift of the position of the resulting graph by a few pixels to maximize similarity. The scale factor is varied in the range from 0.8 to 2.0.

It is important to note that this process not only yields the identity of the object, but also its orientation (with an accuracy of 9°) and its position with an accuracy of about one pixel. Results of a typical recognition process are shown in Fig. 7. Full details about the algorithm can be found elsewhere (Kefalea, 1998, 1999; Kefalea et al., 1997).

3.4. Grip Planning

For grasping an object, its position, size, orientation, and type must be known. These are usually provided by the object recognition module. For each known object, several different grips are stored in a *grip library* (see Fig. 8 for an example), where they are represented in an object-centered frame.

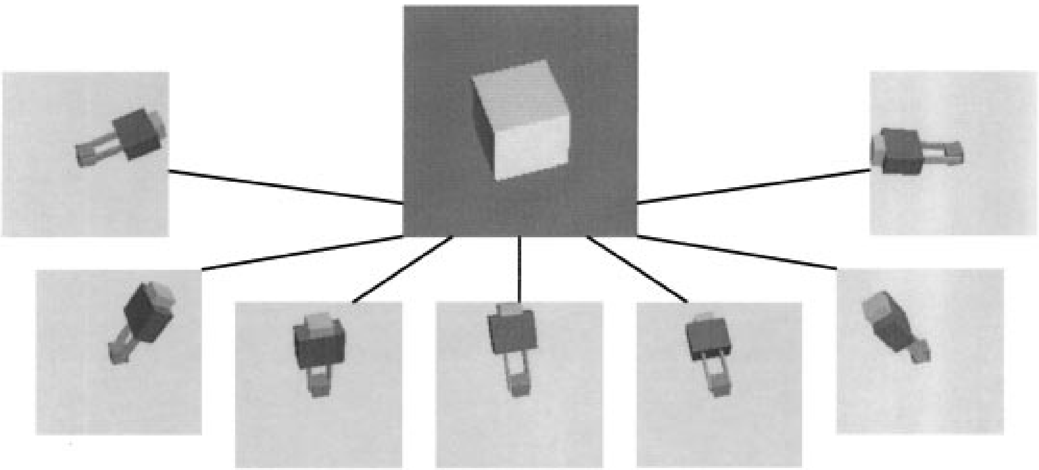


Figure 8. Example for the grips stored with an object in the grip library.

3.4.1. Building the Grip Library. The grip library contains three items per object: (i) the possible grips, (ii) the sizes of the object graphs if they are at a default distance, and (iii) the systematic difference between the real object center in three dimensions and the one estimated by object recognition.

(i) The grips are represented in an object-centered coordinate system. In the long run, good grips must be learned from tactile feedback about their stability, which has been postponed until the necessary sensors are implemented. In the meantime we use a reflex-controlled system, based on tactile feedback and inspired by the grip learning of infants (Zadel, 1999). This works in a simulation setup on CAD-models of the simple objects in our database (cube, cylinder, cuboid, . . .). For the more complicated objects such as the car in Fig. 10(e), the grips have been preprogrammed.

(ii) For scaling the grips to different object sizes, the sizes of the object graphs at the time of grip learning are needed. These sizes are dependent on the orientation of the object.

(iii) The position of an object, i.e., its center of gravity, is estimated from the centers of the object graphs. Due to this two-dimensional estimation of the object image centroids, there is a systematic difference to the real object's center of gravity. The difference ranges from zero to about 25 mm and depends on object type and orientation. This difference is learned in a simulation which thus helps to substantially improve the accuracy of grasping (Zadel, 1999).

The grip parameters (i)–(iii) are currently learned in a simulation of the complete setup including robot, table and object, the robot's kinematics, image acquisition, object localization, fixation, and object recognition. The suitable grips for each object type are defined and stored in the grip library. Then, the modules for object localization and fixation are used to fixate the simulated object, and object recognition determines the sizes of the object graphs and their positions. The sizes are normalized by the object's distance, and the difference between the visually estimated and the real object center is calculated. Finally, both are stored in the grip library. This procedure is repeated for all object shapes and orientations.

3.4.2. Selecting and Parameterizing a Grip. For selecting a grip, the results of object recognition in both images are averaged to reduce noise and quantization effects (for the orientation angle). Then, the object type is used to extract suitable raw grips from the grip library. These grips are scaled according to object size, and the difference between the real object center and the estimated one is compensated for as learned in the simulation. The grips are then transformed into the object's position and rotated according to its orientation. Finally, the grasp directions of each grip are matched against the desired direction, and the best-matching grip is passed on to be executed by the robot.

3.5. Robot Kinematics and Trajectory Generation

3.5.1. Universal Kinematics. In order to execute a grip, a smoothарm trajectory in three-dimensional space has to be planned and transformed into joint space, where control of the arm takes place. This transformation depends on the robot geometry. As our robot is modular and easily reconfigured, we have decided against the use of analytical kinematics, which would be specialized to a given configuration. Instead we use numerical direct kinematics, which are separate for each joint, as a local geometrical model of **GripSee**'s arm. For control we use the *resolved motion rate control method*, which transforms a Cartesian motion into a joint motion using the inverse Jacobian. The Jacobian is calculated numerically as a partial velocity matrix (Wampler, 1986) with the joint positions and axes of the local geometrical model. As the robot arm has a redundant DoF, the transformation of Cartesian movements into joint movements is not unique. In this case the pseudoinverse of the Jacobian (Klein and Huang, 1983), calculated by *singular value decomposition*, yields a minimum norm solution. The redundant DoF, which is given by the null space of the Jacobian, is handled with the gradient projection method (Klein and Huang, 1983). This allows formulation of a secondary task, e.g., dexterity optimization or obstacle avoidance. For stabilization of motion near singularities we use a *damped least squares method* (Wampler, 1986), which balances the cost of large joint velocities against large trajectory deviations in combination with the gradient projection method for optimization of joint range availability.

This local control scheme runs at 100 Hz on one of the Pentium processors and is independent of the number of joints and their geometrical configuration. The robot-specific parameters are currently measured by the user and supplied in the form of

Devanit-Hartenberg parameters (see Paul, 1981 for a definition) to the robot control module. There is also the possibility to learn a robust model of the robot with a hierarchical neural network (Maël, 1996) using visual information, which is a major goal of our future work. Currently, the camera coordinate frame is calibrated to the predetermined robot coordinate frame (see Section 3.6).

3.5.2. Flexible Trajectories. Planning and generating trajectories in a flexible and efficient manner is another major problem in robotics. Our solution uses a cubic spline interpolation to plan trajectories in joint or in three-dimensional space, which are smooth in velocity and acceleration. The first and final splines are of degree 4 and thus can handle boundary conditions for velocity and acceleration. This allows a trajectory to be interrupted at any time and a different one to be smoothly fitted to it. In this way, complex trajectories can easily be generated by combining simple ones. Our trajectory generator also has the option to control only a subset of the six end effector coordinates and leave the remaining degrees of freedom to optimize the movement according to various requirements. This is especially useful to exploit the redundant DoF in our robot arm.

3.6. Autonomous Calibration of the Camera Head

In this section we describe two successive tasks that **GripSee** learns autonomously. The first one is the fixation of a point in three-dimensional space, i.e., bringing its projected position onto the left and right image centers. This skill is used by the object localization agent (Section 3.2). The second task is the estimation of a point's spatial position, which enables the robot arm to move to it, and finally grasp an object located there.

3.6.1. The Fixation Task. One way of solving the fixation task is to compute a hard-wired solution of the inverse kinematics of the stereo camera head. To avoid problems resulting from damage and wear, we have chosen for an adaptive component to learn this task. If, during operation of the system, the fixation error exceeds a threshold, some learning steps will be executed in order to regain calibration. This currently involves an intervention by the operator but will be completely autonomous once the performance (i.e., the precision of a grip) can be assessed by visual and tactile feedback.

Because of the geometry of our camera head, the images of both cameras are rotated against each other by an angle depending on the current tilt and vergence angles (Pagel et al., 1998a, 1998b). Therefore, there is a nonlinear mapping from an object's initial pixel positions in each image and the current tilt and vergence angles onto the desired increments of pan, tilt, and vergence required for the fixation movement.

In order to represent that mapping, we first train a *growing neural gas* network (Fritzke, 1995) for the fixation task *offline* in a simulation using a perfect model of the hardware. This achieves an average fixation error of about 0.5 pixels after less than 20,000 learning steps and is meant to be a rough estimate of the true kinematics. The error is computed by measuring the variance of the object's final position in both images. This accuracy (see Fig. 9) is achieved with a higher density of neurons near the fixation point (the fovea) combined with an additional fixation step (a small correction saccade) (Pagel et al., 1998a, 1998b). When that mapping is implemented on the real hardware, the average error increases to about 5 pixels. The network is then trained *online* by using the real kinematics of **GripSee**'s hardware. To this purpose, a small light (which can be easily located in a darkened room) is held by the gripper into the visual fields of both cameras at different random locations in the robot's workspace. After about 300 further learning steps, the resulting average fixation error drops below 1.5 pixels (Fig. 9).

3.6.2. The Position Estimation Task. To grasp successfully, the head angles and arm positions have to be coordinated. The arm moves the light to random positions in its workspace. After the two fixation steps another growing neural gas network is trained with the current head angles as input and the end effector position as output. Because only the offsets to the hardwired camera head kinematics are learned, we achieve a minimum estimation error of about 5 mm after less than 5 training steps. The internal lower bound for this error is about 4 mm, computed for a fully extended robot arm of one meter in length. This shows that the second network is not needed for the current state of our hardware. However, should any perturbations affect the system, it would be very useful to restore sufficient accuracy.

4. Example Behaviors

The skills we have implemented are among those crucial for a perceptual robot. In this section, we will present examples of how they can be combined to yield

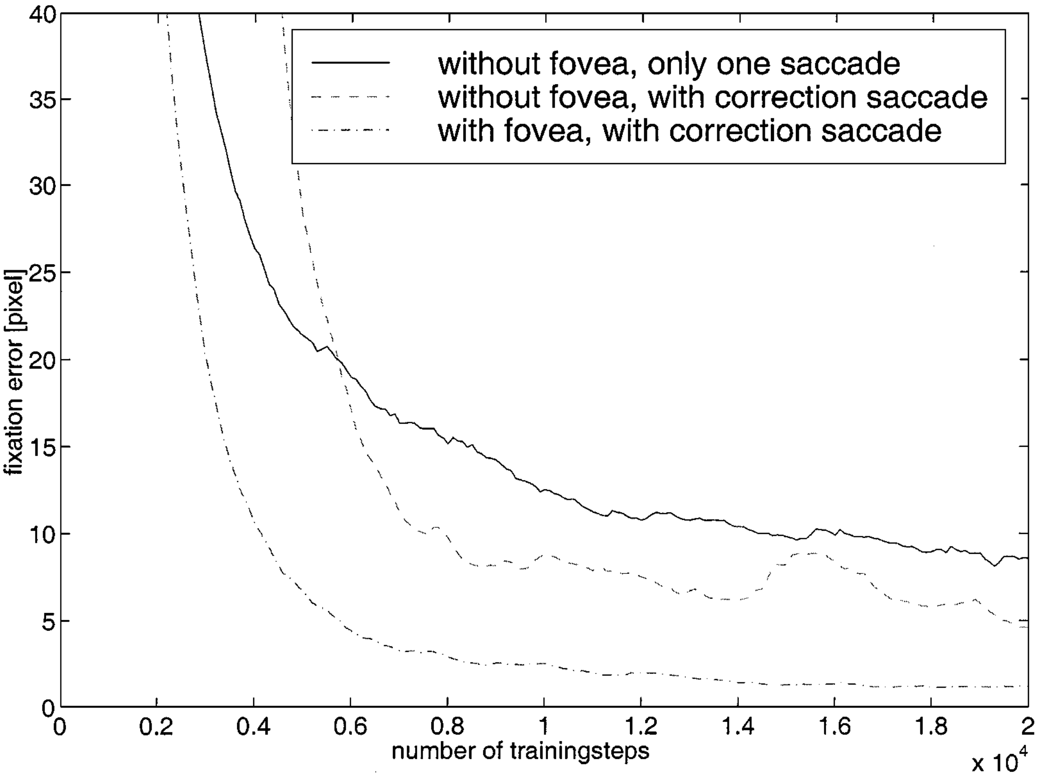


Figure 9. The fixation error of the network for the various parts of the self calibration process. See text for details.

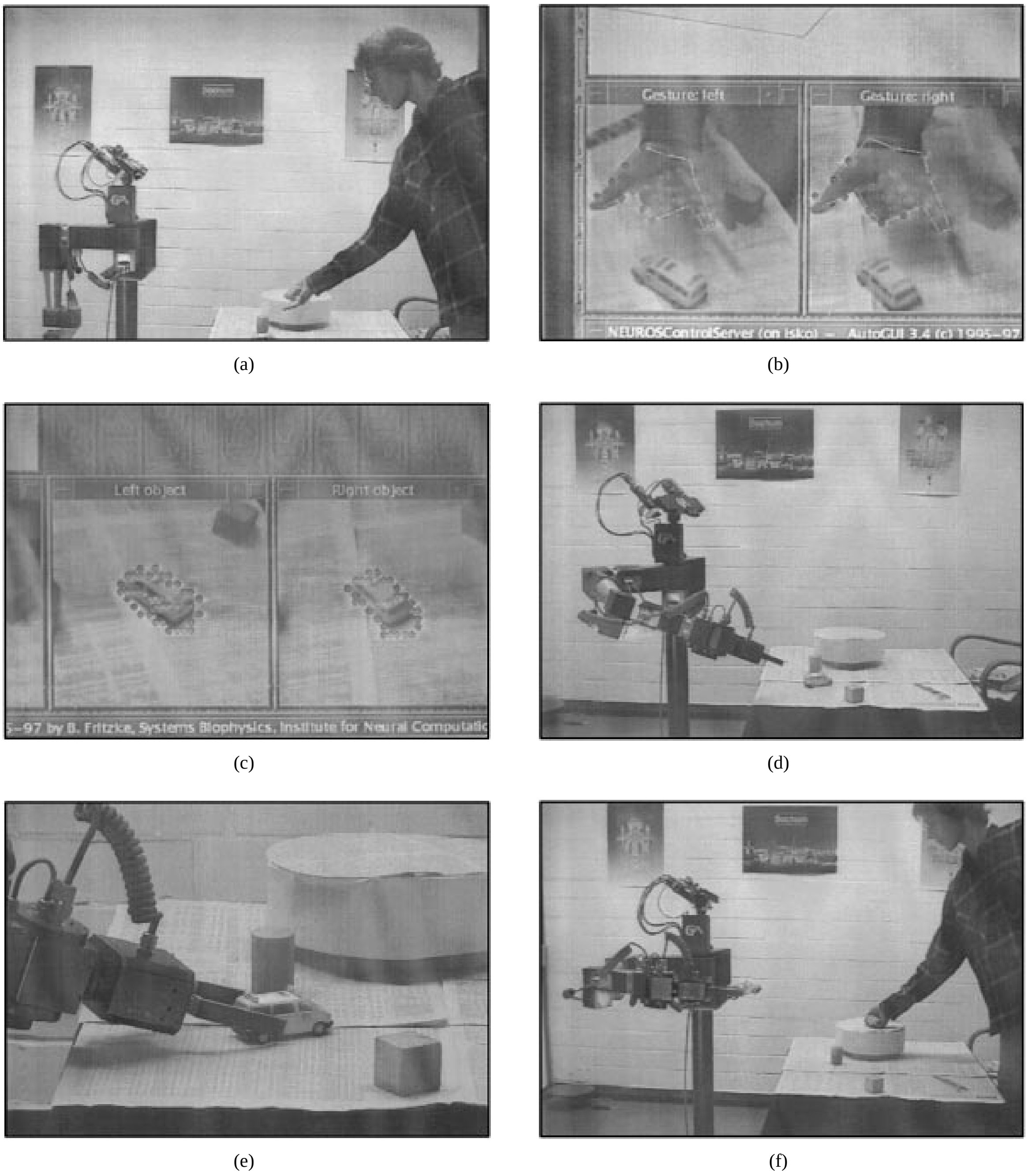
a useful behavior. We will first describe the system setup and then an example, in which the behavior consists of grasping one of the objects placed on a table. The operator selects the object by pointing to it with a gesture indicating a grasp direction and then the object is grasped. If only one object is present, grasping can proceed without operator intervention. A third behavior is to put down a grasped object at a point indicated by the operator. The first and third behavior can be combined for a complete pick-and-place task as illustrated in Fig. 10.

4.1. System Setup

Before the system can be used, several things have to be set up. Ideally, each of them should be constructed completely autonomously, which would qualify each setup as a behavior of its own. Fixation and triangulation have to be calibrated. This is performed autonomously using visual information and assuming correct calibration of the arm (precise proprioception), as described in Section 3.6. A set of hand postures is defined and the postures have to be learned and stored in a *hand posture gallery*. For that, the system goes into a learning mode, in which examples of the postures are presented, accompanied by keyboard commands that identify them. Manual interaction is required to construct the appropriate graphs. Then, a set of objects has to be learned and stored in an *object gallery*. It must contain views from many directions, which is currently done by placing the objects onto a rotating table and recording model views at 9° apart. Possible grips for each of the objects and other parameters have to be learned and stored in a grip library. As system for autonomous initialization of the basic grips exists in simulation (Zadel, 1999) and will be implemented as soon as the tactile sensors are working.

4.2. Recognizing and Grasping an Object of Choice

In the absence of input from the operator, **GripSee** looks down onto the table where some objects are placed. Then, the following sequence procedure is started:


Gesture: left
Gesture: right
NEUROSControlServer (on isko) — AutoGUI 3.4 (c) 1995–97
Left object
Right object
5–97 by B. Fritzke, Systems Biophysics, Institute for Neural Computatic

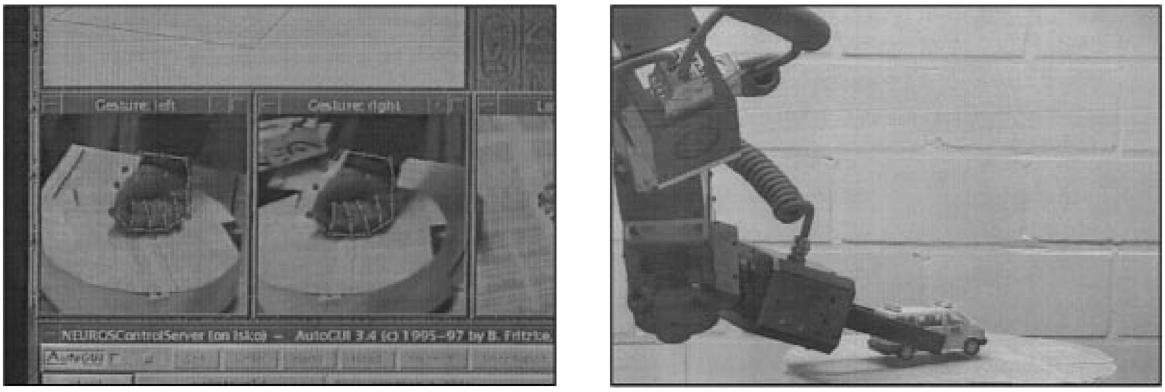
(a) (b)

(c) (d)

(e) (f)

Figure 10. Eight frames from a run of the whole behavior. (a) The user points to an object with a gesture ($t = 2.6$ s). (b) The gesture is recognized and localized by graph matching ($t = 82.2$ s). (c) The object is recognized and localized ($t = 142.9$ s). (d) The arm is moved towards the object ($t = 164.8$ s). (e) The object is grasped ($t = 189.6$ s). (f) The user depicts a point for putting down the object using another gesture ($t = 233.1$ s). (g) The gesture is localized ($t = 266.5$ s). (h) The object is released ($t = 308.5$ s). An MPEG-video is available from our web page.

(*Continued on next page*.)


Gesture: left
Gesture: right
NEUROSControlServer (on isko) – AutoGUI 3.4 (c) 1995–97 by B. Fritzke

(g) (h)

Figure 10. (*Continued*).

Hand tracking: The hand of the operator is tracked as soon as it enters the field of view.

Hand fixation: After stopping, the hand is fixated for recognition of the posture.

Hand posture analysis is executed on the distortion-reduced small central region of interest. This yields type and position of the posture.

Object fixation: Starting from a point about 10 cm below the center of the hand, the localization algorithm detects an object, which is then fixated. This is an iterative process which typically takes three or four cycles to determine a good fixation point. Reduction of the perspective distortion by fixation is crucial for a good estimate of the object's orientation in the following step.

Object recognition determines the type of object, its position, size, and orientation in both images.

Fixation: The recognition module yields a more precise object position than the previous fixation. This position is crucial for reliable grasping; therefore, the object is fixated once more in order to get a refined estimate of its spatial position by triangulation.

Grip planning. A grip suitable for the object and the type of grip indicated by the hand posture is selected and transformed into the object's position and orientation.

Trajectory planning and grip execution. A trajectory is planned, arm and gripper are moved to the object, and finally the object is grasped. Another trajectory transfers the object to a default position conveniently located in front of the cameras for further inspection.

4.3. Selecting an Arbitrary Object

In the previous example, hand tracking and posture analysis demonstrate our approach to human-robot interaction. They are used to select an object and a grasp direction, but they are not needed to find the object. Thus, an alternative behavior can be started with object localization and fixation. In that case, the robot selects the most significant one (in the sense of edge density) of the objects on the table and grasps it. The height of the table is unknown to the robot and need not be supplied, as the object's position is determined in space by fixation and triangulation. Thus, **GripSee** can deal with objects at different heights, such as on a table loaded with piles of paper. With simple modifications of the attention mechanism for blanking out objects already attended to, the procedure can be iterated to search for a an object with a specific shape, which can, in turn, be indicated by a gesture, because the meaning of the gestures can be assigned freely.

4.4. Placing a Picked Object

In a third example, demonstrating the flexibility and modular design of the system, we extended the procedure to a pick-and-place operation: After the robot grasps and picks up the object, the operator points out where it should be placed. The skills are reused as follows:

Hand tracking, hand fixation: GripSee again surveys the field of operation for the operator's hand to appear.

Hand posture analysis: Here, a reduced version with only one posture for pointing (a fist) is used, because now the requirement is to determine a precise three-dimensional position to place the object.

Fixation: The hand is fixated to determine its exact position.

Trajectory planning, execution, and release: A trajectory is planned, the arm moves the object to the desired location and the gripper releases it. In the absence of further cues, the 3D-precision of that release is an immediate measure for the quality of hand localization.

As for grasping, **GripSee** has no a priori information about the height above the table but finds it by triangulation. In fact, a robotically knowledgeable lab visitor was impressed when **GripSee** released an object smoothly onto his palm with his fist pointing at it, a behavior that had not been tested (and not even thought of) before. More example behaviors can be implemented with relatively little effort. The overall success of the system relies heavily on the quality of the perceptual components, which is relatively good and subject to continuous improvement.

5. Results

In this section, we present quantitative results for the single skills as well as for the overall behaviors.

5.1. Results for Gesture Control

Table 1 shows the results of gesture recognition, which becomes faster and more reliable when fewer gestures are used. A further speedup can be achieved by reducing the number of nodes in the graphs. The parameter set used on **GripSee** is shown in the third row of Table 1, the first two rows show results for a larger gallery. The six gestures used for control are shown in Fig. 11.

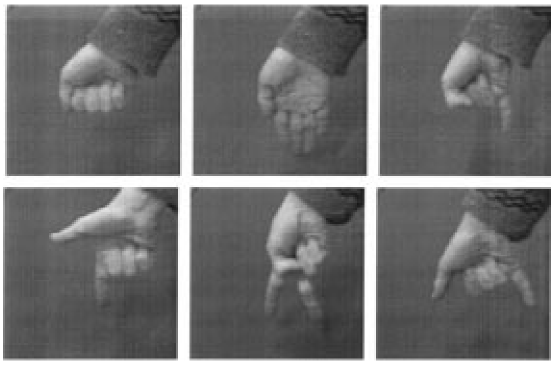


Figure 11. The six gestures used for control (upper row: A, B, C, bottom row: D, E, F).

For reliable gesture control and for the assignment of meaning to the individual postures not only the absolute recognition rates are important but also the confusion matrix which is shown in Table 2. The main confusions (F taken as A and B taken as C) are caused by the attempted background independence, because part of the hand is erroneously regarded as background, a problem which can be alleviated by a more careful selection of meaningful postures.

5.2. Results for Object Recognition

Object recognition performs well both in cases of uniform and lightly structured background. Since object features are extracted only at points on contours, recognition is independent of surface markings, which is an

Table 2. Confusion matrix for the experiments in the lower row of Table 1. The letters correspond to the gestures from Fig. 11. The total number of experiments is 96.

Posture	Taken as A	Taken as B	Taken as C	Taken as D	Taken as E	Taken as F	Errors
A	15	1					1
B		11	5				5
C	2		13		1		3
D	1			14	1		2
E			3		13		3
F	5		1	1		9	7
Errors	8	1	9	1	2	0	21

Table 3. Results for object recognition. The backgrounds ranged from a uniform color (unstructured) over moderately complex (lab scene) to difficult (newspapers under the object). The number of nodes per graph varied between 10 and 30, the average matching time was 30 s in each case.

No. of objects	Background	No. of tests	Percentage correct
12	Unstructured	72	95.8
12	Moderately complex	52	92.3
12	Difficult	56	82.1

advantage in our setting because for grasping the object's shape matters much more than the texture on the surfaces. In its current state the system has 12 stored objects, and recognition time is roughly proportional to that number. Typically, it takes the Sun workstation about 2–3 s to match with *one* multigraph, i.e., compare with one object.

The performance of object localization and recognition also depends on the parameters. The results of large comparison runs, which are currently under way, will be published elsewhere (Kefalea, 1999). Some figures that indicate the performance are shown in Table 3. In the case of object recognition, the number of nodes per graph varies considerably.

5.3. *Results for the System as a Whole*

Figure 10 shows an example run of the most complicated behavior implemented, a sequence of (a) a user pointing at an object, (b) the posture being recognized, (c) the object pointed to being recognized, (d–e) and picked up, (f) another gesture for determining where to put down the object, (g) its localization, and (h) the final release of the object. The whole sequence takes 5.9 minutes. The durations of posture and object recognition are longer than the ones in Tables 1 and 3, because the graphical display of the process is time consuming and irrelevant for the behavior itself. An MPEG-video of this run is available from our web page.

Results for the autonomous calibration of the camera head have been published (Pagel et al., 1998a, 1998b). Figure 9 shows the precision of fixation after calibration.

The performance of grasping and releasing is completely determined by the localization of the gesture and by the correctness of object recognition. Where those fail, manipulation fails. This shows the need to include grasping under visual feedback in later stages of the work. Given ideal results of object recognition, the uncertainty of the grasping position resulting from the finite image resolution is about 2×2 mm in the image plane and 1 cm in depth.

6. Conclusions and Future Work

We have designed and demonstrated a system which displays many of the basic skills required of a service robot: human-robot interaction (control by an operator through pointing and gestures), spatial vision (finding regions of interest and fixating them, recognizing objects and estimating their geometrical parameters), manipulation (grip and trajectory planning), and adaptive self-calibration. We have shown that specific complex behaviors can be easily set up from the building blocks that implement these individual skills. Concept and operation of the system can be compared to related work from a variety of viewpoints, from which we pick *man-machine interaction*, *perceptual capabilities*, and *organization of behavior*.

Simple man-machine interaction is one of the major prerequisites for useful service robots. It is generally felt that a user interface should contain language and visual interaction (Dario et al., 1996; Kawamura et al., 1996), which goes to such lengths as equipping a robot with a face capable of changing expression (Hara and Kobayashi, 1997). For special solutions for handicapped persons, a graphical user interface may be the method of choice (Tsotsos et al., 1998). We have decided to develop a gesture-based user interface, because of the conceptual relatedness to our object recognition approach. Examples for related systems are (Cipolla and Hollinghurst, 1997; Crowley, 1996), which need a uniform background. In our system, special care has been taken to achieve good recognition in the presence of difficult backgrounds. The opposite direction of interaction—passing messages from **GripSee** to the user—is currently only implemented via a terminal.

A huge amount of work has been done on visually guided grasping, e.g. (Hutchinson et al., 1996; Pauli, 1998; Yau and Wang, 1997). Our system currently only uses vision for the estimation of the position and orientation of an object and grasps it blindly. Although this is not satisfactory in the long run, it illustrates the quality of the object recognition method we are using. One of our most important constraints for system design is that internal representations must be learnable, i.e., designed such that they can be built up autonomously by the robot. This philosophy is somewhere in between the "classical" AI approach of highly sophisticated world models and the absence of any internal representation of the outside world (Brooks, 1991).

An important relative to **GripSee** is **Cog** (Brooks, 1997), a robot with a much more complete anthropomorphic body and with a more sophisticated behavioral organization. We have concentrated on the perceptual issues which we believe are indispensable for useful interaction with the real world. This aspect of our system is very similar to the one from (Tsotsos et al., 1998). The behavioral organization is relatively simple at the moment and will have to be improved once more skills are active concurrently, e.g., along the lines of (Crowley, 1995). Similar to **Cog**, **GripSee** still lacks many features but shows good promise for extension.

A major feature of the research done in our group is the focus on integration of various cues, behaviors or modalities. In the domain of computer vision the underlying concepts are Attributed Graph Matching (Lades et al., 1993; Lourens and Würtz, 1998; Wiskott et al., 1997), integration of multiple scales (Würtz, 1997; Würtz and Lourens, 1997), and integration of multiple segmentation cues in a system of interacting spins (Eckes and Vorbrüggen, 1996; Vorbrüggen, 1995). On the control side, the most important concept is the autonomous refinement of rough preprogrammed schemas (Corbacho and Arbib, 1995; Zadel, 1999) by proprioception and autonomously acquired visual information. Regarding software engineering, the consistently object-oriented design of the FLAVOR package (Rinne et al., 1998) developed at our institute has proved its power by allowing rapid integration of different perceptual modules into a coherent system. As a matter of fact, various modules had been developed long before the **GripSee** project was conceived, and were incorporated with amazingly few difficulties.

Clearly, both the list of skills and the performance of the perceptual modules will require considerable improvement before the goal of a perceptually guided robot can be reached. Nevertheless, we claim that we have taken a major step toward that end by supplying several central perceptual and manipulatory skills together with a convincing way of integration. A considerable degree of autonomy has been demonstrated, which will increase once visual and tactile feedback are implemented. In detail, the following steps will be:

- integration of tactile sensors onto the gripper;
- autonomous learning of grips using tactile feedback;
- visual learning of the arm kinematics to improve flexibility and robustness;
- more robust object localization and recognition by consistent exploitation of disparity cues;
- grasping with real-time visual feedback (visual servoing);
- imitation of trajectories or grips performed by the operator;
- autonomous learning of an object representation suitable for recognition and grasping;
- integration of readily available face recognition (Lades et al., 1993; Wiskott et al., 1997) for operator identification and authorization.

Beyond the implementation of the actual skills and behaviors, we have demonstrated that our integration concept constitutes a successful and promising strategy for learning in perceptual robots. This gives rise to the hope for **GripSee** to attain additional non-trivial perceptual capabilities in the near future.

Acknowledgments

We thank the designers and implementors of FLAVOR, especially Michael Pötzsch, Michael Rinne and Bernd Fritzke. Michael Neef has maintained a superb computing environment throughout the last years. Karsten Brauer took images of postures and objects and filmed the demonstration video, Gesa Lorenz helped with its digitization. **GripSee** was designed in cooperation with our colleagues Herbert Janßen, Percy Dahm, Rainer Menzner, Thomas Bergener, and Carsten Bruckhoff, and we thank them for their help in getting everything off the ground.

References

Bergener, T., Bruckhoff, C., Dahm, P., Janßen, H., Joublin, F., and Menzner, R. 1997. Arnold: An anthropomorphic autonomous robot for human environments. In *Workshop SOAVE '97*, Horst-Michael Groß (Ed.), Ilmenau, Germany, VDI Verlag, Düsseldorf, pp. 25–34.

Bergener, T. and Dahm, P. 1997. A framework for dynamic man-machine interaction implemented on an autonomous mobile robot. In *ISIE'97, IEEE International Symposium on Industrial Electronics*, IEEE Publications: Piscataway, NJ, pp. SS42–SS47.

Brooks, R.A. 1991. Intelligence without representation. *Artificial Intelligence Journal*, 47:139–160.

Brooks, R.A. 1997. From earwigs to humans. *Robotics and Autonomous Systems*, 20(2):291–304.

Cipolla, R. and Hollinghurst, N. 1997. Visually guided grasping in unstructured environments. *Robotics and Autonomous Systems*, 19(3/4):337–346.

Corbacho, F.J. and Arbib, M.A. 1995. Learning to detour. *Adaptive Behavior*, 3(4):419–468.

Crowley, J.L. 1995. Integration and control of reactive visual processes. *Robotics and Autonomous Systems*, 16(1):17–27.

Crowley, J.L. 1996. Vision for man-machine interaction. *Robotics and Autonomous Systems*, 19(2):347–358.

Dario, P., Guglielmelli, E., Genovese, V., and Toro, M. 1996. Robot assistants: Applications and evolution. *Robotics and Autonomous Systems*, 18:225–234.

Eckes, C. and Vorbrüggen, J.C. 1996. Combining data-driven and model-based cues for segmentation of video sequences. In *Proceedings WCNN96*, INNS Press & Lawrence Erlbaum Ass., pp. 868–875.

Fritzke, B. 1995. Incremental learning of local linear mappings. In *Proceedings ICANN95*. F. Fogelman-Soulié and P. Gallinari (Eds.), Paris, EC2 & Cie, pp. 217–222.

Hara, F. and Kobayashi, H. 1997. State-of-the-art in component technology for an animated face robot—its component technology development for interactive communication with humans. *Advanced Robotics*, 11(6):585–604.

Hutchinson, S., Hager, G.D., and Corke, P.I. 1996. A tutorial on visual servo control. *IEEE Transactions on Robotics and Automation*, 12(5):651–670.

Kawamura, K., Pack, R.T., Bishay, M., and Iskarous, M. 1996. Design philosophy for service robots. *Robotics and Autonomous Systems*, 18(2):109–116.

Kefalea, E. 1998. Object localization and recognition for a grasping robot. In *Proceedings of the 24th Annual Conference of the IEEE Industrial Electronics Society (IECON '98)*, IEEE, pp. 2057–2062.

Kefalea, E. 1999. Flexible object recognition for a grasping robot. Ph.D. Thesis, Ruhr-Universität Bochum, in preparation.

Kefalea, E., Rehse, O., and v.d. Malsburg, C. 1997. Object classification based on contours with elastic graph matching. In *Proc. 3rd Int. Workshop on Visual Form*, Capri, Italy, World Scientific, pp. 287–297.

Klein C.A. and Huang, C. 1983. Review of pseudoinverse control for use with kinematically redundant manipulators. *IEEE Transactions on Systems, Man, and Cybernetics*, 13(3):245–250.

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.

Lourens, T. and Würtz, R.P. 1998. Object recognition by matching symbolic edge graphs. In *Computer Vision—ACCV'98*, R. Chin and Ting-Chuen Pong (Eds.), volume 1352 of Lecture Notes in Computer Science. Springer Verlag, pp. II-193–II-200.

Maes, P. 1994. Situated agents can have goals. In *Designing Autonomous Agents*, P. Maes (Ed.), 3rd edition. MIT press, pp. 49–70. Reprinted from Robotics and Autonomous Systems, Vol. 6, Nos. 1 and 2 (June 1990).

Maël, E. 1996. A hierarchical network for learning robust models of kinematic chains. In *Artificial Neural Networks—ICANN 96*, C. von der Malsburg, J.C. Vorbrüggen, W. von Seelen, and B. Sendhoff (Eds.), volume 1112 of Lecture Notes in Computer Science, Springer Verlag, pp. 615–622.

Mallat, S. and Zhong, S. 1992. Characterization of signals from multiscale edges. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14:710–732.

Pagel, M., Maël, E., and von der Malsburg, C. 1998a. Self calibration of the fixation movement of a stereo camera head. *Autonomous Robots*, 5:355–367.

Pagel, M., Maël, E., and von der Malsburg, C. 1998b. Self calibration of the fixation movement of a stereo camera head. *Machine Learning*, 31:169–186.

Paul, R.P. 1981. *Robot Manipulators: Mathematics, Programming and Control*, MIT Press.

Pauli, J. 1998. Learning to recognize and grasp objects. *Autonomous Robots*, 5(3/4):407–420.

Rinne, M., Pötzsch, M., and von der Malsburg, C. 1998. *Designing Objects for Computer Vision (FLAVOR)*, in preparation.

Rosenschein, S.J. 1985. Formal theories of knowledge in AI and robotics. *New Generation Computing*, 3(4):345–357.

Suchmann, L. (Ed.). 1987. *Plans and Situated Action*, Oxford University Press.

Triesch, J. and von der Malsburg, C. 1996. Robust classification of hand postures against complex backgrounds. In *Proceedings of the Second International Conference on Automatic Face and Gesture Recognition*, IEEE Computer Society Press, pp. 170–175.

Triesch, J. and von der Malsburg, C. 1998. A gesture interface for robotics. In *FG'98, the Third International Conference on Automatic Face and Gesture Recognition*, IEEE Computer Society Press, pp. 546–551.

Tsotsos, J.K., Verghese, G., Dickinson, S., Jenkin, M., Jepson, A., Milios, E., Nuflot, F., Stevenson, S., Black, M., Metaxas, D., Culhane, S., Ye, Y., and Mann, R. 1998. PLAYBOT: A visually-guided robot to assist physically disabled children in play. *Image and Vision Computing*, 16:275–292.

Vorbrüggen, J.C. 1995. *Zwei Modelle zur datengetriebenen Segmentierung visueller Daten*, Reihe Physik. Verlag Harri Deutsch, Thun, Frankfurt am Main.

Wampler, C.W. 1986. Manipulator inverse kinematic solutions based on vector formulations and damped least-squares methods. *IEEE Transactions on Systems, Man, and Cybernetics*, 16(1):93–101.

Wiskott, L. 1996. *Labeled Graphs and Dynamic Link Matching for Face Recognition and Scene Analysis*, Reihe Physik. Verlag Harri Deutsch, Thun, Frankfurt am Main.

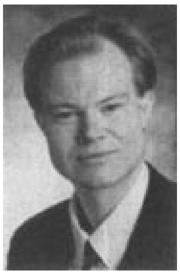
Wiskott, L., Fellous, J.M., Krüger, N., and von der Malsburg, C. 1997. Face recognition by elastic bunch graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):775–779.

Würtz, R.P. 1997. Object recognition robust under translations, deformations and changes in background. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):769–775.

Würtz, R.P. and Lourens, T. 1997. Corner detection in color images by multiscale combination of end-stopped cortical cells. In *Artificial Neural Networks—ICANN '97*, Wulfram Gerstner, A. Germond, M. Hasler, and J.-D. Nicoud (Eds.), volume 1327 of Lecture Notes in Computer Science, Springer Verlag, pp. 901–906.

Yau, W.Y. and Wang, H. 1997. Robust hand-eye coordination. *Advanced Robotics*, 11(1):57–73.

Zadel, S. 1999. Greifen bei Servicerobotern: Sehen und Tastsinn, Lernen und Selbstorganisation. Ph.D. Thesis, Ruhr-Universität Bochum, in preparation.

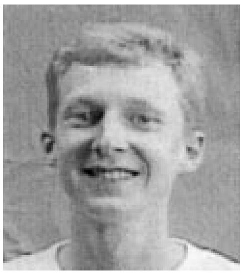


Mark Becker received his Diploma in Electrical Engineering from the University of Bochum, Germany in 1994 with a topic in the field of digital signal processing. Since June 1994 he has been with the group of Prof. v.d. Malsburg in the Institute for Neurocomputing at the University of Bochum, where he developed software applications for digital signal and parallel processing hardware. Within the GripSee project he is working on the hardware setup and software integration.

Efthimia Kefalea received her Diploma in Computer Engineering from the University of Patras, Greece, in 1992. She was with the “Forschungszentrum Informatik”, at the University of Karlsruhe, Germany from 1992 to 1993, where she worked on the representation of information. In 1993, she joined the Institute for Neurocomputing at the University of Bochum, Germany, where she is with the group of Prof. v.d. Malsburg. Her current research interests include robotic vision, object recognition/classification, learning of object representations, sensor fusion and control of visual attention. She is a member of IEEE and the Technical Chamber of Greece.

Eric Maël received his Diploma in Physics from the University of Dortmund, Germany, in 1993 with a topic in theoretical high energy physics. He then studied robotics for one year at the Institute for Robotics Research at Dortmund. Since 1994 he has been with the group of Prof. v.d. Malsburg in the Institute for Neurocomputing at the University of Bochum, Germany, where he is working on adaptive kinematics for redundant manipulators. His current research include robotics in combination with neural networks and computer vision.

Christoph von der Malsburg received both the Diploma and the Ph.D. degrees in Physics from the University of Heidelberg, Germany. He then spent 17 years as staff scientist in the Department for Neurobiology of the Max-Planck-Institute for Biophysical Chemistry in Göttingen, Germany. In 1988 he joined the Computer Science Department and the Section of Neurobiology of the Biology Department at the University of Southern California in Los Angeles. In 1990 he took on a position as director at the Institute for Neurocomputing at the University of Bochum, Germany. His interests are in brain organization, mainly at the level of ontogenesis and function of the visual system.



Mike Pagel received his Diploma in Physics in 1997 from the Ruhr-Universität Bochum, Germany. In his first studies he examined the structure and behavior of neural dynamic fields. Then, during his stay at the Institut für Neuroinformatik, Bochum, Germany, he developed and applied the algorithmic structures of the camera head system used above. In November 1997 he became a member of the Computational and Biological Vision Lab at the University of Southern California, Los Angeles. Since March 1998 he is working at a computer vision company in Santa Monica, CA.

Jochen Triesch studied physics at the University of Bochum, Germany and the University of Sussex, England. He received his

Diploma from University of Bochum in 1994 with a thesis on the metric of visual space. Since then he has been research assistant of Prof. v.d. Malsburg at the Institute for Neurocomputing in Bochum. His research interests include the binding problem in neural networks, gesture recognition, imitation learning and sensor fusion.

Jan C. Vorbrüggen received his Diploma in Physics from the University of Bonn in 1988. He then spent two years as research assistant at the Max-Planck-Institute for Brain Research in Frankfurt, Germany. In 1990, he joined the Institute for Neurocomputing at the University of Bochum. He received his Ph.D. in Physics in 1995 with a thesis on models for visual scene segmentation using cue integration. Since then, he has continued working on technically and biologically plausible segmentation models as a permanent staff member of the Institute. Additional research interests are computer architecture, parallel processing, VLSI implementation of machine vision algorithms, and biological models of vision and cognition.

Rolf P. Würtz obtained his Diploma in Mathematics from the University of Heidelberg, Germany in 1986. After that, he was research assistant at the Max-Planck-Institute for Brain Research in Frankfurt, Germany. In 1990, he joined the Institute for Neurocomputing at the University of Bochum, Germany, where he received his Ph.D. from the Physics department in 1994. Until 1997, he was a postdoctoral researcher at the department of Computing Science at the University of Groningen, The Netherlands. He is currently responsible for the GripSee project at the Institute for Neurocomputing. Further research interests include neuronal models and efficient algorithms for object recognition, hand-eye coordination, integration of visual and tactile information, and links to higher cognition.

Stefan Zadel received the Diploma degree in Electrical Engineering from the University of Stuttgart, Germany, in 1991. He then joined the group for system biophysics of the Institute for Neurocomputing at the University of Bochum, Germany, first as a Ph.D. student in a fellowship program, then as research assistant. He worked on the field of service robot grasping, especially the learning of grasps. In 1998 he joined the department of process development of Daimler-Chrysler at Stuttgart.