Self Calibration of the Fixation Movement of a Stereo Camera Head *

MIKE PAGEL

ERIC MAËL

CHRISTOPH VON DER MALSBURG

Institut für Neuroinformatik, Ruhr-Universität Bochum, Germany

Editors: Maja Matarić and Henry Hexmoor

Abstract. In this article we show how an active stereo camera head can be made to autonomously learn to fixate objects in space. During fixation, the system performs an initial and a correction saccade. In the learning phase the correction saccade is controlled by a crude pre-wired algorithm, in analogy to a mechanism surmised to exist in the brainstem. A vector-based neural network serves as the adaptive component in our system. A self-organizing fovea improves dramatically the convergence of the learning algorithm and the accuracy of the fixation. As a possible application we describe the visuo-motor coordination of the camera head with an anthropomorphic robot arm.

Keywords: self calibration, fixation, gaze control, visuo-motor coordination, growing neural gas

1. Introduction

Humans and higher animals have the ability to fixate points of interest by coordinated movement of the two eyes. This is important to let the image of the object of interest come to lie on the fovea, the part of highest resolution within the retina. As the angular speed attained during saccades can be as high as 1000 deg/sec, visual feedback is too slow to control saccadic movement (Carpenter, 1988). A sequence of several saccades with intervening visual sampling of progress is not feasible since initiation of a saccade takes 200 msec for a human (Carpenter, 1988). However, a direct computation of the full motion required to attain foveation with the necessary precision seems not to be possible in the visual system, which therefore routinely performs an initial relatively imprecise saccade and a second small correction saccade (Robinson, 1968).

In the generation of saccades the superior colliculus seems to play a prominent role (Robinson, 1968). When the visuo-motor correspondence of the visual system is disturbed, e.g., by prisms or lenses, the system is able to re-establish a correct mapping (Henson, 1978, Dean et al., 1991). This adaptive capability is necessary to cope with natural changes, e.g., during growth and aging. There is the general belief that during adaptation the system is guided by a simple rigid brainstem mechanism capable to provide a crude adaptive motion if an incorrect saccade has been performed (Optican, 1985, Carpenter, 1988, Keller, 1989, Dean et al., 1994).

Although in the technical domain it is possible in principle to design and operate active camera systems without recourse to adaptive capabilities, just relying on analytical descrip-

* Supported by grants from the DFG (Graduiertenkolleg KOGNET) and the German Federal Ministry for Research and Education (01 IN 504 E9).

tions of kinematics and precise knowledge of system parameters, it is highly advantageous to invoke adaptivity there too, for essentially the same reasons important in biology. Without it, unavoidable wear and decalibration would require frequent tedious and costly manual readjustments. Moreover, adaptivity in a robot system could greatly relax awkward and expensive design constraints and would permit such features as pliability or hysteresis, which are difficult to model analytically. Even more important are learning capabilities in future robots designed to act in natural environments, where even the intended movements cannot be designed ahead of time but have to be flexibly structured in response to unforeseen stimuli. Our ultimate goal with the project described here is to gain experience relevant for the development of a flexible robot able to act under perceptual control.

Our contribution is an incremental improvement on previous models for adaptive oculomotor maps, e.g., (Grossberg & Kuperstein, 1986, Rao & Ballard, 1995, Ritter et al., 1992) or for other types of sensorimotor calibration, e.g., (Anastasio & Robinson, 1989, Suzuki et al., 1987). The regular way for adaptive systems to deal with curved motor coordinate systems is to set up a more or less regular array of sample points ("units") each of which carries an affine local motor map. Another general feature is a simple (linear) teaching network that suffices to diagnose or iteratively reduce errors. Our particular approach is distinguished by techniques to judiciously choose sample points where they are needed most, thus economizing units, or improving the accuracy of the system. We achieve this by making use of a growing neural gas network (Fritzke, 1995a) and by letting the system self-organize a fovea, that is, an increased local density of units in the region which is reached by the initial saccade. The growing neural gas is able to adapt to the topology of the space that is actually used by the system, in the present case a two-dimensional subspace of the originally three-dimensional kinematic space. In distinction to Ritter's system (Ritter et al., 1992), in which a kinematic fovea is imposed on the system by unevenly distributed training examples, our system develops the kinematic fovea on its own.

2. Analytical consideration of the fixation movement

2.1. Description of the used camera head model

Our robot system uses a stereo camera head¹ with three degrees of freedom. The first axis allows a horizontal pan rotation, the second performs a vertical tilt movement. The third possible movement is done by altering the vergence angle between the two cameras. As usual the geometrical model is mathematically formulated by using homogeneous transformations to define a kinematic chain (see (Craig, 1989) for further explanation). In our case the chain consists of four transformations (see fig. 1): the first three are given by the three angles already mentioned, the fourth one rotates the current camera frame so that the picture is spanned by the x, y -plane and the view looks along the negative z -direction. This last transformation only simplifies the computation of picture- from world-coordinates.

An important fact is that the movements of both cameras are coupled. That leads to problems if one wants to find a simple mapping from initial pixel positions of an object in the left and right image to the desired head turns, such that the resulting pixel positions are identical to the image centers. But let us first consider the fixation of *one* camera to explain this.

Figure 1. The camera head (**A**: photo, **B**: model) with its three degrees of freedom: pan, tilt and vergence. The numbers are indicating the end position of the corresponding coordinate transformation in the kinematic chain: (0) base system, (1) head offset and pan rotation, (2) camera offset and tilt rotation, (3) vergence rotation, (4) (not shown) constant transformation of the camera image into the x,y -plane.

2.2. Fixation with one camera

As a first approximation the two coupled cameras might be considered independent from each other. One camera should have only two degrees of freedom: pan and tilt. Because the x -position of an object in the image can be changed by performing a pure pan rotation and the y -position of the same object by making a pure tilt rotation (see fig. 2A), it is obvious that the pan/tilt fixation movement $(\Delta p, \Delta t)$ is of the form:

$$\begin{aligned}\Delta p &= \Delta p(x) \\ \Delta t &= \Delta t(y)\end{aligned}\tag{1}$$

We now compute the exact value for the movement for the x -axis, without loss of generality. If x is the distance in pixels from the object to the image center, r_x the x -resolution of our image and $\gamma_x^{\max}$ the maximal (half) angle seen by a camera (see fig. 2B), then the following relations hold:

$$\tan \gamma = \frac{x}{f_x}\tag{2}$$

$$\tan \gamma_x^{\max} = \frac{r_x}{2 \cdot f_x}\tag{3}$$

Thus the resulting angles to perform the fixation step are:

$$\begin{aligned}\Delta p &\propto \arctan \left[\frac{2x}{r_x} \tan \gamma_x^{\max} \right] \\ \Delta t &\propto \arctan \left[\frac{2y}{r_y} \tan \gamma_y^{\max} \right]\end{aligned}\tag{4}$$

where the last equation follows from similar considerations for the y -axis.

2.3. Generalization for two coupled cameras and resulting problems

If two cameras are coupled in a way shown in section 2.1, they share the same pan and tilt angles p and t , and the vergence angle v labels the angle between the view directions of both cameras. A first simple attempt to compute a correct fixation movement for this model might be a generalization of the formulae shown above:

$$\begin{aligned}\Delta p &\propto \arctan \left[\frac{2\bar{x}}{r_x} \tan \gamma_x^{\max} \right] \\ \Delta t &\propto \arctan \left[\frac{2\bar{y}}{r_y} \tan \gamma_y^{\max} \right] \\ \Delta v &\propto 2 \cdot \arctan \left[\frac{\Delta x}{r_x} \tan \gamma_x^{\max} \right]\end{aligned}\tag{5}$$

with

$$\begin{aligned}\bar{x} &:= \frac{1}{2}(x_r + x_l) \\ \bar{y} &:= \frac{1}{2}(y_r + y_l) \\ \text{and } \Delta x &:= x_r - x_l\end{aligned}\tag{6}$$

The last expression Δx is also known as the disparity of the object point. The indices r and l indicate that their corresponding values are taken from the right or left camera, respectively.

A closer look at these equations reveals several problems:

1. The pan axis no longer lies in the camera chip plane. Hence, the pan movement computed in (5) is not correct, although the error is negligible if the half width of the head's stereo-basis is small compared to the distance to the object of interest.

Figure 2. **A:** Image of one mono camera, containing a visible object point that may be fixated on by a combination of pure pan and tilt movements. The dashed lines mark the center of the image which is usually the end point of a perfect fixation. **B:** Geometrical relations for one image dimension (scaled and simplified). The visible object point is mapped onto the pixel plane. The angle γ between an object and the view direction is equivalent to the desired pan rotation Δp . The camera parameters are the length f_x , the resolution r_x and the maximal visible angle $\gamma_x^{\max}$.

2. If both tilt and vergence are non-zero, the camera image is rotated around the optical axis (see fig. 3). Therefore, the simple mappings of $\bar{x} \rightarrow \Delta p$ and $\bar{y} \rightarrow \Delta t$ in equation (1) become wrong.

This rotation of the camera images – if the left image is rotated clockwise the right one rotates counterclockwise and vice versa – causes a non-linear mapping from the pixel positions to the resulting fixation movement angles. Moreover, this rotation angle depends on the current tilt and vergence angles. For that reason the correct mapping is of the form:

$$\begin{pmatrix} \Delta p \\ \Delta t \\ \Delta v \end{pmatrix} = f(x_l, y_l, x_r, y_r, t, v) \quad (7)$$

where f is the non-linear function described above. This function also depends on the head geometry which may vary due to the external influences mentioned above. Thus, we have attempted to have the robot learn it autonomously, instead of computing it analytically.

Figure 3. Left and right camera images taken with $(p, t, v) = (0^\circ, -36^\circ, 38^\circ)$. Positive pan means that the head turns to the left, positive tilt means that the heads looks down. The scan-lines of the object seen in both images are no longer parallel, they are rotated against each other.

3. The adaptive approach

3.1. Selection of a network algorithm

Each neural algorithm that has input and output facilities may be used to perform the task of learning a mapping from input to output values (see (Haykin, 1994) for further details). This can be done very easily with a vector based neural network, because the nodes of such a network consist of input/output pairs, which are simply the base points of the approximated function.

In this work we use the *growing neural gas* network as it has been introduced in (Fritzke, 1995a) and its extension to the learning of *local linear mappings* (Fritzke, 1995b). The basic characteristics of this network algorithm are:

- The network consists of nodes and edges, therefore it has a graph structure.
 - Nodes** contain two reference vectors, one for the input and one for the output space. Additionally, for each node a local linear mapping is used to linearly extrapolate the output value in a region around the node position.
 - Edges** are used to connect two nodes. These connections create a topological neighborhood in the graph, which is needed for the moving node to shift its neighbors by an amount proportional to its own displacement.
- The output of the network corresponding to a presented input vector ξ is computed as follows. At first the nearest node (in input space) has to be found. Let w be the input reference vector of this unit, θ the output reference vector and A the corresponding local linear mapping. Then,

$$g(\xi) = \theta + A \cdot (\xi - w) \quad (8)$$

computes the output answer of our network.

- During the training, pairs of input/output patterns are presented to the network. Proper adaptation takes place by moving the reference vectors of the nearest nodes towards the training pattern and by adapting the local mapping with respect to the output of the net for each presented input vector (see Eq. (8)).

Edges can be created and removed depending on how frequently the corresponding nodes are chosen as one of the nearest units with respect to the training input vector. Nodes that are not connected by any edge are removed and new nodes are inserted into the input region after a fixed number of training steps with the maximal approximation error.
- All the training parameters are fixed ². Decrease of the adaptation step width is caused automatically when the number of nodes increases, because it is proportional to the distance from the nearest node.

Now we can formulate the learning task in terms of the growing neural gas network. The six-dimensional input space is given by the values of x_l, y_l, x_r, y_r, t and v . The output space of the network is three dimensional and represents the fixation movement in the form of corresponding angle offsets $\Delta p, \Delta t$ and Δv .

The next subsection describes how the necessary information is presented to the network in order to improve the currently contained knowledge about the approximated function.

3.2. The training scheme for self calibration

3.2.1. Learning with a crude a priori algorithm. Due to the fact that the camera head should learn the fixation task autonomously one might think that supervised training, as it

is needed for the type of network described above, is not the correct approach. Who should provide the information, who is the supervisor, the teacher? This is a justified question, especially if the development of the corresponding oculomotor functionality in humans is considered. There is no teacher who tells a baby how to move the eyes in order to look into a specific direction.

The answer to this question may be derived by considering the different brain regions involved in learning motor tasks. In case of the oculomotor system it is assumed that a crude hard-wired, pre-defined functionality already exists at birth, partly localized in the brain stem (Dean et al., 1991). This coarse knowledge provides slow and inaccurate movements that may be improved by adaptive components which are characteristic of an individual or, in our case, of a special machine. The cerebellum is often thought of as such an adaptive component (Keller, 1989, Optican, 1985). Here we propose a module based on the growing neural gas network.

The simple expressions in (5) which are not completely correct as it already has been mentioned, are used as an *a priori* mechanism of the fixation task, and therefore present input/output training patterns to the network.

What is learned by the network algorithm? A natural candidate would be the correct non-linear mapping in equation (7). Since the crude algorithm provides no perfect training patterns but local improvements from the last position – a step done by this algorithm brings an object closer to the center of the image center regardless of the starting position – each training step will lead to a better approximation of the desired function (see the learning scheme below). The second but at least as important feature is the difference between model and real world kinematics. If, for example the head's stereo basis is not exactly the one assumed, additional errors will occur. The adaptive component will also be able to learn the influence of such effects.

Now we would like to illustrate how the training takes place. One learning step can be described as follows:

1. move the camera head into a random position (p_0, t_0, v_0)
2. find a random object point in space that is visible in both camera images
3. get the pixel positions $(x_{l0}, y_{l0}, x_{r0}, y_{r0})$ of this object point
4. get the network output $(\Delta p_0, \Delta t_0, \Delta v_0)$ for the input $(x_{l0}, y_{l0}, x_{r0}, y_{r0}, t_0, v_0)$ and move the head into this direction
5. get the pixel coordinates $(x_{l1}, y_{l1}, x_{r1}, y_{r1})$ of the object point for this new position
6. compute the crude fixation step $(\Delta p_1, \Delta t_1, \Delta v_1)$ and make that move
7. train the network with the original pixel and head position and the complete resulting movement, i.e., train the mapping

$$\begin{pmatrix} x_{l0} \\ y_{l0} \\ x_{r0} \\ y_{r0} \\ t_0 \\ v_0 \end{pmatrix} \longrightarrow \begin{pmatrix} \Delta p_0 + \Delta p_1 \\ \Delta t_0 + \Delta t_1 \\ \Delta v_0 + \Delta v_1 \end{pmatrix} \quad (9)$$

Figure 4. Learning curve (smoothed) during the training of a simulated camera head. The constant dotted line indicates the mean error that is introduced by only using the crude *a priori* algorithm (see equ. (5)). The error value is taken from the variance of the pixel end positions of the object of interest.

Figure 4 shows the learning curve of the network compared to the (constant) error that is introduced by only using the crude algorithm. The error has been measured by performing about 100 test moves and computing the variance of the resulting end positions of a fixated object point in both camera images, which seems to be a natural measure of the fixation accuracy. We note that this way of error computation is independent of the mean end positions in both images. This important feature will be used for the recalibration facilities of our system in section 3.4.

3.2.2. Improvement of the convergence by using a self-organized fovea. In the simulation mentioned above we have reached a mean fixation accuracy of about 7 pixels. Further learning would improve the performance of the network very slowly as it can be seen in figure 4. To produce a higher accuracy we need to make more than one fixation step. This is also the case in human vision, where the first big saccade is often followed by a small correcting one (Grüsser & Grüsser-Cornehls, 1995). If we simply start a new fixation step

Figure 5. Influence of *one*, *two* and *three* training iterations on the fixation quality (smoothed). **A:** Fixation error after one fixation step. Because of the lower cell density in the peripheral visual field the error achieved with 2 or 3 successive training steps is slightly higher than with only one iteration. **B:** Fixation error after two fixation steps. Now the accuracy improves very strongly if the net has been trained with more than one iteration. Furthermore it may be noticed that there is no significant difference in the quality of fixation movement, if the network has been trained with two or three training iterations.

where the first one has ended, the mean error (in the same network as above) is reduced below 5 pixels, but that is still not sufficient.

To achieve much higher accuracy especially in the second fixation step, the distribution of the network nodes has to be changed. Because of the uniformly presented input data (remember the random steps 1 and 2 in the training scheme shown above) the units of our growing neural gas network are distributed in the input space in the same uniform way. The reproduction of topologies is a typical feature of this kind of network algorithm (Fritzke, 1995a).

The first fixation step takes us into a region with a radius of about seven pixels around the mean end position. In this center region the density of network nodes should be higher than on the outside, which would cause a higher accuracy in the successive step. How can this kind of cell distribution (further referred to as *fovealized* distribution) be reached?

Very often the structure of the resulting network topology is determined directly by the probability density of the presented training patterns (see (Fritzke, 1995a, Martinetz, 1992) for examples). In contrast to this artificial way of producing the desired output, we used a more functional approach to create the required cell distribution.

After the first training step a successive, second training step is done, starting at the pixel and camera positions where the first step has ended. Speaking in terms of the training scheme of the previous subsection, a new step has to be introduced after the last one:

8. continue with step 3 until the number of fixation steps performed is *maxNumberOfFixations*

The old training scheme would be realized by setting *maxNumberOfFixations* to 1. We have tested the influence of this iterated training on the accuracy of successive fixation steps. See figure 5 for the results: with two training iterations we have reached a mean fixation error of about 10 pixels after one fixation step and a mean error of less than one pixel after two fixation steps (sub-pixel accuracy is possible because these experiments are still on a simulated basis).

3.3. Examination of the resulting network topologies

3.3.1. Cell distribution in the input space. If the ccd chip of one camera is taken as an analogue for a retina then we may compare the mean fixation end position to a biological fovea. To reinforce this impression we now take a look at the resulting node distributions in the input space. Figure 6 shows the node positions in the network input space and the cell densities, projected onto the camera images for training with one or more iterations. Notice that in the case of more than one training iteration the resulting distribution of the nodes is qualitatively comparable to the well known distribution of cones in a human retina (e.g., see (Zrenner et al., 1990)): a very high density in the fovea region and a decreasing density in the peripheral visual field.

3.3.2. Topology in the output space. To finish our analysis of the network topology we now try to understand the resulting output structure of the network. At first we have to say that this topology does not depend on the number of training iterations as it was the case in

Figure 6. Distribution of the nodes of our network in the input domain. Left column shows the results of one training iteration, right column of two iterations. In the upper two rows the node positions are projected onto the left and right images, where in case of two training iterations a self organized fovea arose. The lower row shows the node density of both distributions: without fovealizing the nodes are approximately uniformly distributed, but with iterated training the density is very high in the fovea region and decreases very fast towards the peripheral visual field. (To improve the effect of fovealizing the network consists of 80 units. To reach a fixation error of 1 pixel less than 40 units are needed.)

the former subsections. But the resulting appearance of the network is still interesting. The output space consists of three dimensions mentioned earlier: the fixation movement angles Δp , Δt and Δv . But the units in this space obviously build a two-dimensional subspace (see fig. 7).

This can be explained as follows: Let us first consider a *fixed* starting position of the camera head. Then the resulting fixation movement only depends on the pixel positions of the object in both images. But this kind of movement depends in a nearly linear way on

Figure 7. Resulting network topology in the output domain. The axes are drawn in the interval $[-30^\circ, 30^\circ]$. Obviously a two-dimensional subspace is built during the learning, which results from the function that has been approximated (see the text). The small expansion of the network along the vergence axis results from the relations in equation (5), that are not completely linear.

the pixels, because the arctan-functions in equation (5) may be considered linear in a first order approximation. Linear mappings can easily be learned by *one* network node, because each node contains its own local linear mapping. Therefore the three degrees of freedom when choosing a random object point in space lead to one local linear mapping of rank three that enables the camera head to fixate on the object from some fixed starting position.

Furthermore, if we allow to change the starting angles, the resulting movement depends on them in a non-linear way. The image rotation (see previous section) only affects the change of pan and tilt angles, because a pure vergence movement *always* affects *only* the x -pixel position of a visible object. As a result only the pan and tilt angles depend non-linearly on the starting position, which produces the distribution of nodes in the pan/tilt plane, because these effects cannot be realized by one linear mapping. The network has to provide many local linear mappings to approximate the non-linear functionality.

3.4. Ability of recalibration

3.4.1. Effects of hardware calibration. What do we mean by the term calibrated hardware? The hardware of such a system is fine-tuned in a very accurate way. In case of our camera head this would include the following conditions: cameras' ccd chips lie exactly in the vergence turning axis, cameras are not turned around their optical axes, the optical axes of both cameras lie in a plane that is exactly horizontal, and so on.

What is the goal of an accurate fixation for a poorly calibrated hardware? If, e.g., one camera looks slightly more upward than the other one, it is obviously impossible to make a fixation move such that the resulting pixel position of an object point in space is centered in both images. Accurate fixation in the common sense cannot be achieved.

But it is important to understand how the crude algorithm (5) trains our network in such situation. The angle to consider for this problem is the tilt angle Δt . It becomes zero if $\bar{y}$ in equation (6) vanishes. A closer look at this equation shows that a *good* end position of a tilt movement is reached if y_r and y_l have the same value but opposite signs and their sum equals zero.

The result of the previous consideration is that in the case of a poorly calibrated hardware our algorithm provides a unique end position for each point in space. In terms of dynamic systems our algorithm has a unique attractor for (p, t, v) for each visible object.

3.4.2. Results of recalibrating different intrusions. In this subsection we show the ability of our system to recalibrate after different kinds of perturbations. To test this feature we have enabled the following ways of detuning the geometrical parameters of the camera head:

Figure 8. Recalibration facilities of our model. Projection of the nodes of both networks onto the image planes after recalibration has taken place. One should notice the new end position of a successful fixation step: both foveas are shifted, one up, one down. The decalibration parameters are for the left camera (roll, pitch, yaw) = $(-5.0^\circ, 5.2^\circ, 8.0^\circ)$, (xShift, yShift, zShift) = (1.0 cm, 0.3 cm, 0.4 cm) and for the right one (roll, pitch, yaw) = $(4.0^\circ, -4.2^\circ, 9.0^\circ)$, (xShift, yShift, zShift) = (0.2 cm, 0.4 cm, -0.3 cm).

- Disfiguring of each camera's orientation. The resulting detuned parameters are: $\text{roll}_l, \text{pitch}_l, \text{yaw}_l, \text{roll}_r, \text{pitch}_r, \text{yaw}_r$.
- Disfiguring of the position of each camera in space. This spatial offset of the chip plane is given by the parameters: $\text{xShift}_l, \text{yShift}_l, \text{zShift}_l, \text{xShift}_r, \text{yShift}_r, \text{zShift}_r$.
- Disfiguring of the base frame. It is possible to detune the position and the orientation, but either one has no immediate effect on the fixation quality. Nevertheless, these

parameters are important for visuo-motor coordination as shown in the last section of this paper.

- Disfiguring of the stereo basis width by changing the Δb parameter.

Parameters like the resolution of a camera or the maximal angles seen by the cameras and so on are not considered because they are easy to measure and will usually not change during operation.

Figure 8 shows a result of recalibration run in our simulation, whereby the number of neurons has been fixed to 40 to avoid the additional information being put into only new units. Because in this example we tested the case considered in subsection 3.4.1 the new *fovea* positions which no longer lie in the image centers.

4. Implementation of the hardware system

In the previous sections we explained our model and checked the features of self- and recalibration. Now we introduce an existing hardware realization of the fixation movement system.

Which modules must be replaced to make a hardware application possible? The first module that has been simulated so far is the detection of a specified object point in both camera images. Our approach needs these pixel positions first for learning and later for performance. In order to separate this work from the problems of object recognition we used a very simple way of finding an object in a stereo image pair: After darkening the room a small lamp is held into the visible region of the cameras. Then the only thing to do is to find this bright spot in the images, which is a simple task since the lamp is the only visible spot.

The next step is automation of the proper learning, especially presentation of an object at different positions in space. This has been solved by using the arm of the robot that our camera head is a part of. The arm moves the lamp to different positions.

Now the training scheme looks as follows:

1. move the camera head into a random position
2. find a random point in space that is visible in both camera images as well as reachable by the robot arm; move the arm holding the lamp to that position
3. get the pixel positions by finding the light in the images
4. fixate on the object using the network
5. get the pixel positions by finding the light in the images
6. fixate using the crude algorithm
7. train the network
8. continue with step 3 until the number of fixation steps performed is equal to *maxNumberOfFixations*

Although the complete training phase could take place on the hardware, we have simulated training first *offline*, mainly to save time, but also to prevent the head from making a lot of unneeded movements. The following recalibration for the real parameters is done in a second step *online* using the hardware. After about 250 to 300 recalibration iterations the resulting error of the fixation end position is below 1.5 pixels.

It is clear that the fixation can only be as accurate as the used object detection is in finding the corresponding pixels for a visible object. Because the lamp is visible in more than one pixel a lower bound for the accuracy between one and two pixels is given implicitly.

5. Possible application of the fixation task: the visuo motor coordination of the camera head and a robot arm

5.1. Learning scheme

Now the camera head can perform an accurate fixation movement in two successive saccades. The end position of this movement is unique for each object point in space. An obvious application of this task is to learn the mapping from these unique head angles to the three dimensional position in space where the object point is located. This is also done with our robot, now using the robot hand's knowledge on the presented position to train a second neural network.

In detail one learning step for self calibration of the visuo-motor coordination of camera head and robot arm proceeds as follows:

1. Find a random position (x, y, z) in space that is reachable by the robot arm. Move the hand holding the light to this position.
2. Fixate on the light using the first network described in the former sections.
3. Train the network with the current camera head angles as input and the position of the light as output, i.e. train the mapping

$$\begin{pmatrix} p \\ t \\ v \end{pmatrix} \longrightarrow \begin{pmatrix} x \\ y \\ z \end{pmatrix} \quad (10)$$

5.2. Relation of the fixation and position estimation quality

To classify the results we now examine how the quality of the position estimation is biased by the accuracy of the fixation movement. Figure 9 shows the interdependence of these two variables in a simulated environment, where the uniformly distributed distance from the object point lies between 0.5 m and 5 m: for an average fixation error of about one pixel the resulting error of the position estimation is about 1 cm. The implicit bias of a fixation error of that dimension is given by the angular distance produced by one pixel. The resulting lower bound for the estimation error – computed for our fully stretched robot arm with the length of 1 m – is about 1.4 mm.

Figure 9. Relation of the position estimation quality and the fixation accuracy. Each data point is averaged over 100 trials, where the accuracy of both modalities (fixation and position estimation) has been measured. To force higher fixation errors, noise has been added to the visual data. The solid line shows the linear regression of the given data points.

Applying the entire system of fixation and position estimation to our hardware, we achieve an average estimation error of less than 1 cm, after an object point in space has been fixated on after two iterations. The achieved accuracy of fixating on an object and estimating the position with the camera head is sufficient for executing a coarse grasp without any feedback. See (Becker et al., 1998) for further documentation on how these skills may be used for visually-guided grasping.

To achieve higher accuracy in the visuo-motor coordination, closed loop approaches are required, where the quality of a grip is controlled visually and/or by tactile information.

6. Discussion

In this paper a method for an autonomous calibration of the fixation task of a camera head is presented. Although the system fully satisfies the requirements of the task, one possible drawback of it (or of the general adaptive approach) is the difficulty of extracting actual kinematic parameters. This is because their values are distributed over the nodes of the whole network and they are thus *hidden*.

The basic usefulness of a Kohonen-like feature map as an approach to the learning of sensory-motor coordination has been demonstrated previously, e.g., by Ritter et.al. (1992). We extend this approach by applying it to a coupled stereo camera head with a curved kinematic space. Furthermore, whereas in (Ritter et al., 1992) the spatial density of the network nodes is externally determined by the presentation of training patterns with a foveal probability distribution, in our system the kinematic fovea and the ensuing space-variant accuracy are self-organized, as a natural consequence of the usage of more than one saccade.

Notes

1. Modified TRC Zebra head with two Sony cameras XC 999P (6mm lens, color).
2. Throughout this work the parameters of the growing neural gas network are fixed to: $\lambda = 302$, $a_{\max} = 300$, $\alpha = 0.5$, $d = 0.005$, $\epsilon_b = 0.1$ and $\epsilon_n = 0.001$.

References

- Anastasio, T.J. & Robinson, D.A. (1989). The distributed representation of vestibulo-oculomotor signals by brain-stem neurons. *Biological Cybernetics*, 61:79–88.
- Becker, M., Kefalea, E., Maël, E., Pagel, M., Triesch, J., Vorbrueggen, J.C., v. d. Malsburg, C. & Zadel, S. (1998). Gripsee: a robot for visually-guided grasping. (submitted to) *IEEE International Conference on Robotics and Automation ICRA 98*.
- Carpenter, R.H.S. (1988). *Movements of the eyes*. Pion, London, 2 edition.
- Craig, J.J. (1989). *Introduction to robotics: mechanics and control*. Addison-Wesley, New York, 2 edition.
- Dean, P., Mayhew, J.E. & Langdon, P.M. (1994). Learning and maintaining saccadic accuracy: a model of brainstem-cerebella interactions. *Journal of Cognitive Neuroscience*, 6:117–138.
- Dean, P., Mayhew, J.E., Thacker, N. & Langdon, P.M. (1991). Saccade control in a simulated robot camera-head system: neural net architectures for efficient learning of inverse kinematics. *Biological Cybernetics*, 66:27–36.
- Fritzke, B. (1995a). A growing neural gas network learns topologies. In G. Tesauro, D. S. Touretzky, and T. K. Leen, editors, *Advances in Neural Information Processing Systems 7*, pages 625–632. MIT Press, Cambridge MA.
- Fritzke, B. (1995b). Incremental learning of local linear mappings. In F. Fogelman and P. Gallinari, editors, *ICANN'95: International Conference on Artificial Neural Networks*, pages 217–222, Paris. EC2 & Cie.
- Grossberg, S. & Kuperstein, M. (1986). *Neural Dynamics of Adaptive Sensory-Motor Control: ballistic eye movements*. North-Holland, Amsterdam, 2 edition.
- Grüsser, O.J. & Grüsser-Cornehls, U. (1995). Gesichtssinn und Okulomotorik. In Robert F. Schmidt and Gerhard Thews, editors, *Physiologie des Menschen*, chapter 16, pages 278–315. Springer, Berlin.
- Haykin, S. (1994). *Neural Networks: A Comprehensive Foundation*. Macmillan College Publishing Company.
- Henson, D.B. (1978). Corrective saccades: effects of altering visual feedback. *Vision Res*, 18:63–67.
- Kawato, M., Furuka, K. & Suzuki, R. (1987). A hierarchical neural-network model for control and learning of voluntary movement. *Biological Cybernetics*, 57:169–185.
- Keller, E.L. (1989). The cerebellum. In Robert H. Wurtz and Michael E. Goldberg, editors, *The Neurobiology of Saccadic Eye Movements*, chapter 11, pages 391–409. Elsevier, Amsterdam.
- Martinetz, T. (1992). *Selbstorganisierende neuronale Netzwerkmodelle zur Bewegungssteuerung*. PhD thesis, Technische Universität München.
- Optican, L.M. (1985). Adaptive properties of the saccadic system. In A. Berthoz and G. Melvill Jones, editors, *Adaptive Mechanisms in Gaze Control*, chapter 4, pages 71–79. Elsevier, Amsterdam.
- Rao, R.P.N. & Ballard, D.H. (1995). Learning saccadic eye movements using multiscale spatial filters. *Advances in Neural Information Processing Systems*, 7:893–900.
- Ritter, H., Martinetz, T. & Schulten, K. (1992). *Neural Computation and Self-Organizing Maps: An Introduction*. Addison Wesley, New York.
- Robinson, D.A. (1968). Eye movement control in primates. *Science*, 161:1219–1224.
- Zrenner, E., Abramov, I., Akita, M., Cowey, A., Livingstone, M. & Valberg, A. (1990). Color perception: Retina to cortex. In L. Spillmann and J. S. Werner, editors, *Visual Perception: The Neurophysiological Foundations*, chapter 8, pages 163–204. Academic Press, New York, 1990.

Received September 1, 1997

Accepted December 30, 1997

Final Manuscript February 1, 1998