Published in: ICANN'96, Proc. of the 1996 Conference, Bochum, pp. 215-220

Adaptive Saccade Control of a Binocular Head with Dynamic Cell Structures

Jörg Bruske, Michael Hansen, Lars Riehn, Gerald Sommer

Lehrstuhl für Kognitive Systeme, Christian Albrechts Universität zu Kiel,
Preußerstr. 1-9, 24105 Kiel

Abstract. In this article we report how Dynamic Cell Structures (DCS) [1] can be utilized to learn fast and accurate saccade control of a four-degrees-of-freedom Binocular Head. We solve the order selection problem by incremental growing of a DCS network until the controller meets a pre-specified precision. Calculation of the controller output is very fast and suitable for realtime control since the resulting network is as small as possible and only the best matching unit and its topological neighbors are activated on presentation of an input stimulus. Training of the DCS is based on error feedback learning and proceeds in two phases. In the first phase we use a crude model of the cameras and the kinematics of the head to learn the topology of the input submanifold and a rough approximation off-line. In a second phase, the operating phase, we employ error feedback learning for on-line adaptation of the linear output units. Besides our TRC binocular head we use a Datacube image processing system and a Stäubli R90 robot arm for automated training in the second phase. The controller is demonstrated to successfully correct errors in the model and to rapidly adapt to changing parameters.

1 Introduction

Saccades are fast eye movements used by animates or robots to change fixation from one point in the visual scene to another. Their speed must be very high (actually up to 1000 deg/s for human eye movements) and it has long been recognized that this speed is much to high to be influenced by visual feedback alone. Hence the saccade control systems must solve the inverse kinematics problem. According to an idea put forward by Kawato et al. [4] the inverse kinematics can be learned by error feedback learning, i.e. by using an error signal proportional to the retinal error as training feed back for a neural network. In his plenary lecture on ICANN'95 Kawato not only gave further neurobiological evidence that this is indeed the way the monkey visual system learns saccade control but that the principle of error feedback learning can be successfully applied to a variety of robotic tasks as well.

In a number of simulations Dean and Mayhew [2] have systematically studied different conventional neural net architectures (Linear nets, CMACs, Multilayer Backpropagation nets) for learning saccade control from error feedback. In [5] Mayhew et al. report a successful implementation of a layered control system for a four-degree-of-freedom stereo camera head utilizing their CMAC based PILUT architecture for learning the inverse kinematics from error feedback.

We have selected DCS [1] for saccade control because it ideally meets the demands of this control task: First, the calculation of the controller has to be as fast as possible to allow control at video rate. The incremental growing DCS network meets this require-

ment by growing the network only as large as to meet a pre-specified precision and, furthermore, by utilizing only a small subset of its neural units for output calculation. Second, it is well known that the angular rotations required to fixate are a linear function of the retinal coordinates of the target only in case of zero tilt and small angles. With increasing tilt the non linearity of the control law increases. Hence if we use an approximation scheme based on locally linear approximation, the density of neural units should be high in regions of the input space with a high tilt component. Contrary to Kohonen type networks or PILUT, the growing DCS is able to achieve this by allocating new neural units in regions of the input space where the approximation error is high. This is exactly the case for regions of the input space deviating from the linear control law.

Finally, the phase space trajectories of multidimensional systems like the head-eye system usually lie on a submanifold which locally may be of very much lower dimensionality than the system. The DCS just attempts a similar reduction in dimensionality in that it places its units in the input submanifold and adapts its lateral connection structure towards a perfectly topology preserving map which is utilized for improved adaptation and approximation.

2 Dynamic Cell Structures

Dynamic Cell Structures (DCS) as introduced in [1] denote a class of RBF-based approximation schemes attempting to concurrently learn and utilize perfectly topology preserving feature maps (PTFMs). DCS are a subclass of Martinetz's Topology Representing Networks (TRN) [6] defined to contain any network using competitive Hebbian learning for building PTFMs.

The architectural characteristics of a DCS network are a) one hidden layer of radial basis functions (possibly growing/shrinking) b) a dynamic lateral connection structure between these units and c) a layer of (usually linear) output units. Training algorithms for DCS adapt the lateral connection structure towards a PTFM by employing a competitive Hebbian learning rule and activate and adapt rbf units in the neighborhood of the current stimulus, where "neighborhood" relates to the simultaneously learned topology.

The particular DCS network used in this article is similar to the one introduced in [1] in that incremental growing of the network is performed similar to B. Fritzke's Growing Cell Structures [3] using a local error variable attached to each neural unit called the resource of this unit.

3 Training scheme

Training proceeds in two stages. First, we use a crude model of the cameras and the kinematics of the head to learn saccade control up to a pre-defined precision off-line. In the second phase, the operating phase, we continue to adapt the output layer of the DCS network on-line to cope with deviations of the real head-eye-system from the model and possibly changing parameters. The first phase could be on-line as well but using a model has the advantage that training takes less time (no real head movement involved) and that, in the second phase, the real system is under reasonable control right from the start.

3.1 Off-line training phase

In this article we restrict ourselves to the so called head movement problem, i.e. for fixating an object the cameras must verge symmetrically and the head must directly point at the target. We use a 5 dimensional input vector $\bar{u} = (x_l, y_l, x_r, \theta_l, \phi)$ which we associate with a 3×3 (Jacobian) matrix A, the output of the DCS network. Here, x_l, y_l, x_r denote retinal coordinates of the target on the left and right camera, θ_l is the vergence angle of the left camera and ϕ the tilt angle of the head, see Fig. 1 (d). The output of the controller $\bar{v}$ is calculated as

$$\bar{v} = (\Delta\chi, \Delta\phi, \Delta\theta_l) = A\bar{p}\,, \tag{1}$$

where $\Delta\chi, \Delta\phi, \Delta\theta_l$ denote the changes (rotations) in pan, tilt and vergence necessary to fixate the target, and $\bar{p} = (x_l, y_l, x_r)$ is the retinal coordinate vector.[1]

The matrix A is computed as a normalized weighted sum of the matrices A_i attached to the rbf units of the DCS network

$$A = \sum_{i \in Nh(bmu)} A_i h_i \text{ with } h_i = \frac{rbf(\|\bar{u} - \bar{\mu}_i\|)}{\sum_{j \in Nh(bmu)} rbf(\|\bar{u} - \bar{\mu}_j\|)}\,, \tag{2}$$

where Nh(bmu) denotes the best matching unit and its direct topological neighbors.

After fixation we use the output of a simple proportional feedback controller $\bar{v}^p = \left(\Delta\chi^p, \Delta\phi^p, \Delta\theta_l^p\right)$ to adapt the matrices A_i. The components $\Delta\chi^p, \Delta\phi^p, \Delta\theta_l^p$ are proportional to the mean target's retinal x-coordinates, the left cameras y-coordinate and the difference in the x-coordinates after fixation. If fixation was perfect, $\bar{v}^p$ would be the null vector.

Applying an α-LMS rule we obtain

$$\Delta A_i = \alpha\left(\bar{v}^p \frac{\bar{p}^T}{\|\bar{p}\|^2}\right) h_i \quad , \quad i \in Nh(bmu) \quad . \tag{3}$$

The centers $\bar{\mu}_i$ of the rbf units are adapted according to a Kohonen type rule,

$$\Delta\bar{\mu}_i = \varepsilon\, \lambda_{(bmu,i)}\, (\bar{u} - \bar{\mu}_i)\,,\ i \in Nh(bmu)\,, \tag{4}$$

and the fixation error in retinal coordinates is used as a resource value. A new unit is inserted whenever the resource value of the bmu *and* the current fixation error exceed the required precision. Between successive insertions we require at least $n \ln(n)$ training steps without insertion, n the current number of neural units, to allow the lateral connection structure of the DCS to build a PTFM. Due to this strategy we need a relatively large number of trials for meeting high precision demands. However, in the off-line phase emphasis is on as small a number of neural units and as good a PTFM as possible since these will no longer be adapted in the on-line phase. Training in the off-line phase stops when the averaged fixation error falls below the pre-specified precision.

In off-line training we use a model of the head-eye system to calculate the new retinal coordinates of the target after applying the controller output $\bar{v}$. Input vectors are generated randomly. The only constraints are that the vergence and tilt angle, θ_l and ϕ, are

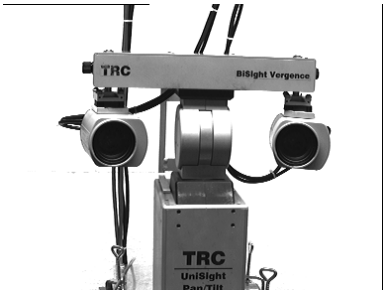
1. We assume $y_l = y_r$, and, because of symmetrical vergence, $\theta_l = \theta_r$. Note that $\bar{v}$ does not depend on χ.

restricted to an interval of interest and that the retinal coordinates x_l, y_l, x_r may not exceed the field of view of the cameras.

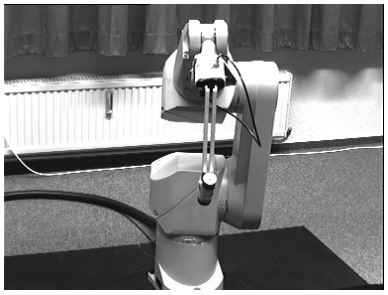
3.2 On-line training in the operating phase

In the operating phase we continue to adapt the matrices A_i in the output layer. This is done by error feedback, eq. (3), just as in the off-line phase. We do no longer grow the network nor do we further adapt the centers or the lateral connection structure.

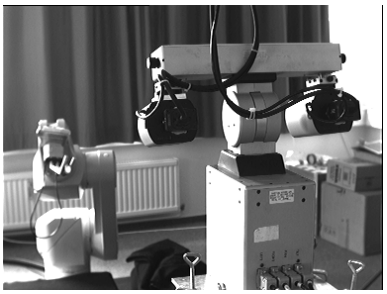
Targets for fixation are generated by randomly moving our Stäubli R90 robot arm, Fig. 1 (b), in its work space with a light source attached to its gripper, Fig. 1 (c). Relying on our Datacube image processing system we can calculate the retinal coordinates of the target w.r.t the two cameras of our TRC binocular head, Fig. 1 (a), at video rate. The controller output as calculated according to (1) and (2) is then applied to fixate the target, and the output of the proportional controller is used for error feedback learning. The latter is also used for a correctional saccade.



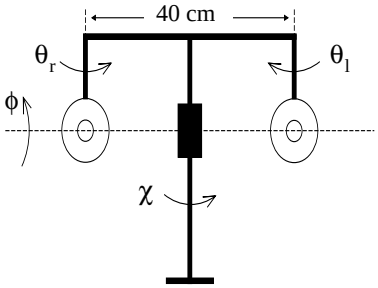
(a) The TRC binocular head



(b) The Stäubli R90 robot arm



(c) Setup for on-line training



(d) Diagram of the TRC head

Fig. 1: Binocular Head (a), (c), (d) and Robot Arm (b), (c)

4 Experimental Results

Fig. 2 shows the average pixel error and the number of neural units in the DCS network versus the number of trials in the off-line training phase for a pre-defined precision of 0.5% (2.5 pixel). On reaching this precision after 40000 simulated saccades only 120 neural units have been inserted into the network. The tilt angle ϕ was restricted to the interval $[-30°, 30°]$ and the vergence angle θ_l to $[0, 10°]$.

We could have reached this high precision with fewer training steps but since we want as few neural units and as good a PTFM as possible, we take our time. After all, the whole off-line training phase takes only 3 minutes for 40000 saccades on a Sparc 4 workstation. The very reasonable accuracy of 1% (5 pixel) is reached after 3500 steps with only 40 neural units.

Now we use the pre-trained controller for saccade control of the real TRC binocular head. As demonstrated in Fig. 3 the error at the start of the operating phase is about 12 pixel (2.5%) which is due to deviations of the model from the real system. However, due to on-line learning by error feedback the error drops down to 2.6 pixel within 1000 saccades. After 1000 saccades we changed the zoom of one of the cameras as reflected by a peak in the error plot. The controller is able to adapt to the new parameter setting of the cameras within the next 1000 saccades.

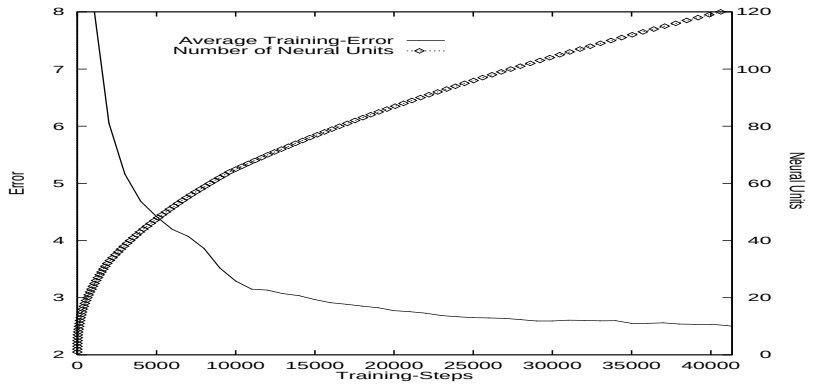


Fig. 2: Average fixation error in pixel and number of neural units in the DCS network versus number of training steps in off-line training phase

5 Discussion

Utilizing a DCS network for learning saccade control from error feedback we were able to substantially extend the work of Dean and Mayhew. By incrementally growing the network up to the size where it meets the pre-specified accuracy it utilizes as few neural units as possible. Since only a minor fraction of these units is involved in calculating the output of the DCS network, the generation of a controller output is very fast (even on conventional hardware) and suitable for real-time saccade control. By virtue of building perfectly topology preserving maps, the centers of the neural (rbf) units and

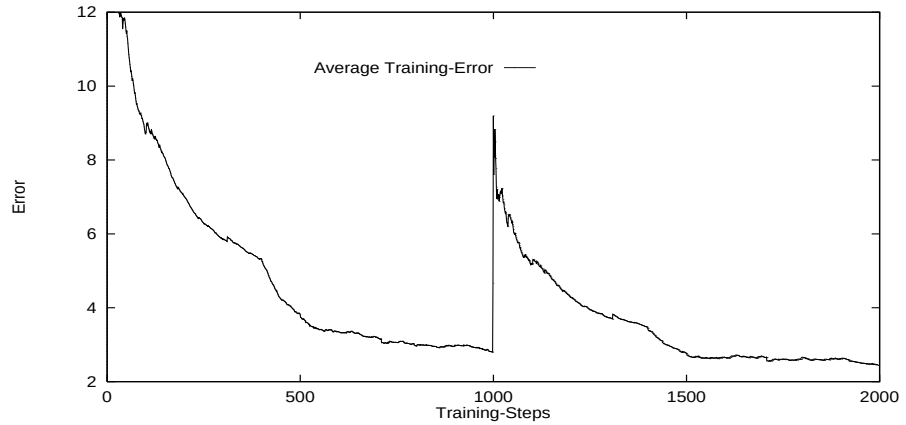


Fig. 3: Average fixation error in pixel versus number of training steps in operating phase. After 1000 saccades the zoom of the left camera changed.

the lateral connection structure are restricted to the relevant regions of the input space. Furthermore, the density of neural units is high in regions of the input space where the required sensory-motor mapping is difficult to approximate, i.e. is non-linear[2].

The controller has been successfully demonstrated to learn saccade control with high precision and to adapt to changing parameters on a real four-degree-of-freedom binocular head.

References

1. J. Bruske and G. Sommer: Dynamic Cell Structure learns Perfectly Topology Preserving Map, Neural Computation, Vol. 7, No. 4 (1995) 845-865
2. P. Dean, J.E. Mayhew, N. Thacker and P.M. Langdon: Saccade control in a simulated robot camera-head system: neural net architectures for efficient learning of inverse kinematics, Biol. Cybern., Vol. 66 (1991) 27-36
3. B. Fritzke: Growing cell structures - a self-organizing network for unsupervised and supervised learning, Neural Networks, Vol. 7, No. 9 (1995) 1441-1460
4. M. Kawato: Feedback-error-learning neural network for supervised motor learning, In: Advanced Neural Computers, Elsevier, Amsterdam (1990) 365-372
5. J.E. Mayhew, Y. Zheng and S. Cornell: The adaptive control of a four-degrees-of-freedom stereo camera head, Phil. Trans. R. Soc. Lond., Vol. 337 (1992) 315-326
6. Thomas Martinetz and Klaus Schulten: Topology Representing Networks, Neural Networks, Vol 7 (1994) 505-522

2. Due to lack of space this has not been demonstrated in this article. Together with additional experiments and further experimental details it will be the subject of a forthcoming publication of the authors.