This paper will appear in:
Proc. of the 8th International Conference on the Simulation of Adaptive Behaviors (SAB'04), 13-17 July 2004, Los Angeles, CA, USA,

# Adaptive Object Tracking with an Anthropomorphic Robot Head

**Hyundo Kim*, Boris Lau**, Jochen Triesch***

*Complex Systems & Cognition Lab
University of California, San Diego
hyundo@ucsd.edu, triesch@cogsci.ucsd.edu
http://csclab.ucsd.edu

**Department of Neuroinformatics
Ilmenau Technical University, Germany
boris.lau@stud.tu-ilmenau.de

## Abstract

We present a system for model-free, adaptive object tracking with a redundant anthropomorphic robot head. The visual tracking system is based on the idea of "Democratic Integration" to fuse information from several cues in a biologically plausible, self-organized fashion. The motor control system utilizes the idea of "controller partitioning" to exploit complementary capabilities of redundant degrees of freedom in the neck and eyes. A biologically inspired vergence control mechanism ensures coordination of the robot's two eyes. We demonstrate the performance of the overall system in different active tracking scenarios.

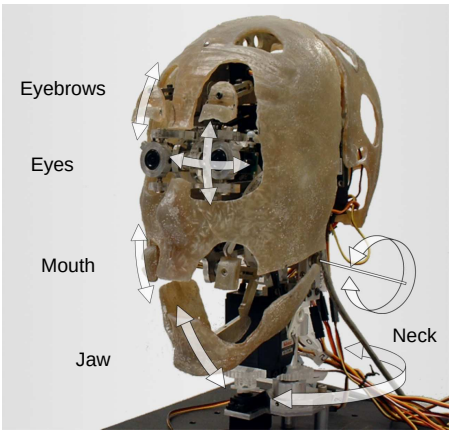


Figure 1: Anthropomorphic robot head with nine DoFs.

## 1. Introduction

The long term goal of our research is to understand how biological systems can autonomously learn to see. How does the developing primate brain form representations of the objects in its environment? Since the developing visual system actively explores its visual environment, it shapes the statistics of what it is seeing rather than being passively exposed to it. Furthermore, it shapes these statistics in a purposeful manner — even infants tested minutes after birth show preferences for some visual stimuli over others. Also, eye movements are likely to play an important role in the development of invariant representations. Thus we believe that in order to fully understand the post-natal development of the primate visual system it is crucial to take its active and goal-driven nature into account. Our approach to the problem of autonomous visual learning is to build embodied models of these active learning processes. To this end we have developed an anthropomorphic robot head platform, whose kinematic structure closely resembles that of the human neck/eye system (Fig. 1). In this paper, we present a system for active object tracking implemented on the robot head. This system will form the "front end" for a more sophisticated module for learning higher level object representations in future explorations into autonomous learning in actively developing visual systems. The system presented here allows our robot to actively track "interesting" objects in its environment, creating the opportunity to learn models of their appearance and behavior.

Our specific goals for this system were the following. First, the tracking scheme needs to be model-free, i.e. it must not require a detailed prior model of the object to be tracked, because we are interested in the *completely autonomous* learning of representations of unfamiliar objects. Second, despite the lack of a model of either the object or its typical patterns of motion, the system should function robustly and be able to cope with fast object movements over a wide range. Third, the system should operate in real-time, generating motor responses at the rate of the visual input of 30Hz. Finally, the system should be roughly consistent with biological constraints, if it is to be part of a larger model for the

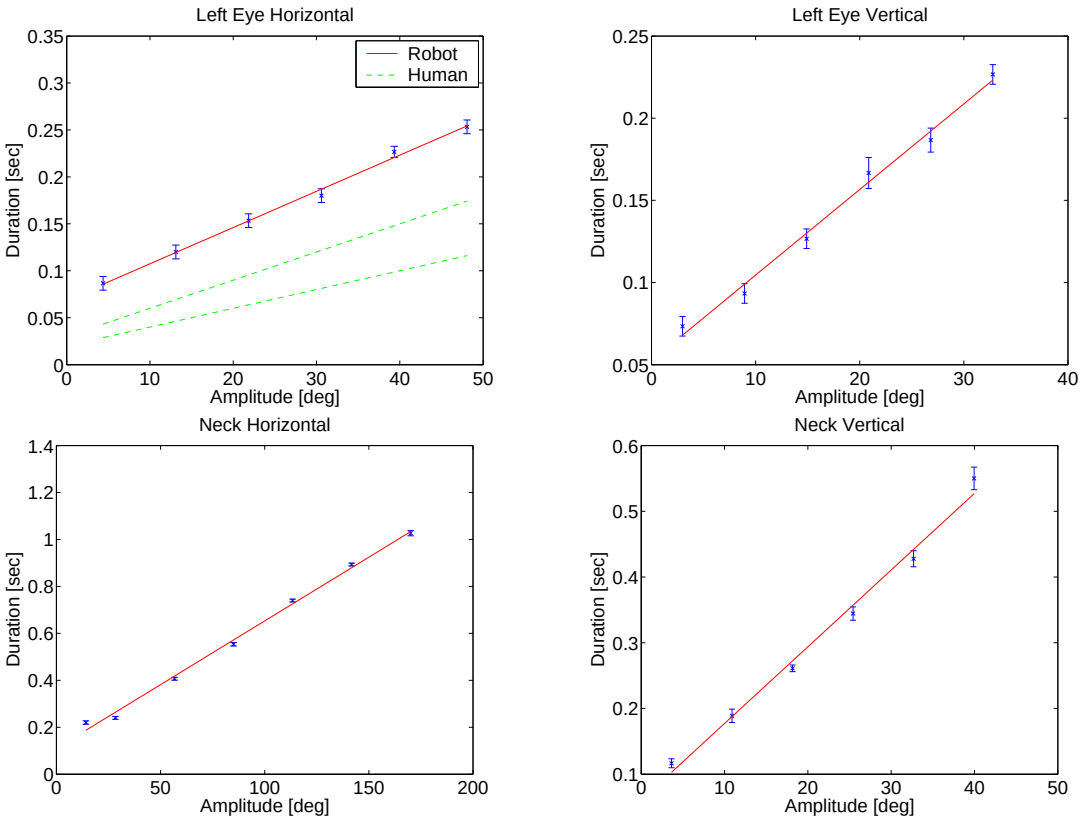


Figure 2: Average time elapsed vs. size of motion for horizontal (top left) and vertical (top right) movements of the left eye and horizontal (bottom left) and vertical (bottom right) movements of the neck. For horizontal eye movements (top left), the dashed lines indicate the range of saccade durations typically observed in humans.

learning of object representations in the primate brain.

The remainder of the paper is organized as follows. Section 2 briefly describes our redundant anthropomorphic robot head. Section 3 explains the adaptive visual tracking system. In Sec. 4 we describe how our partitioned motor control system of the robot exploits the complementary nature of the robot's degrees of freedom (DoFs). Section 5 presents experiments and Sec. 6 discusses our system from a broader perspective.

## 2. Anthropomorphic Robot Head

We have developed an anthropomorphic robot head that closely mimics the neck-eye system of the human head (Fig. 1). The robot head is intended as a flexible platform for studying autonomous visual learning and human robot interaction. A detailed description will be published elsewhere (Kim et al., 2004). Here we only give a brief overview. The robot head has a total of 9 degrees of freedom (DoFs). Six of them are for the neck and the two eyes: both eyes can independently pan and tilt. The neck can pan and tilt, too. The remaining 3 DoFs are for facial expressions (jaw opening, eyebrow lifting, smiling/frowning) but will not be used in this paper. The robot head is controlled from a host computer (Linux PC, Pentium-4 3.06GHz processor, 2GB RAM).

The robot's eyes are two miniature CCD cameras (Point Grey Research, model FireFly). The cameras are connected to the host computer via two IEEE 1394 interfaces with 400 Mbit/s. Each camera gives 24 bit color images with up to 640x480 resolution at 30 fps. We use a focal length of 4mm, providing a diagonal field of view of 100 degrees.

All DoFs are actuated via servo motors. They provide fast, easy, and fairly accurate control, while being robust and inexpensive. Most DoFs are connected via four bar linkages, introducing a slightly non-linear relation between the rotation of a servo shaft and the rotation of the actuated DoF. Table 1 gives the range of motion and the average angular resolution for the robot's major DoFs. The servos are controlled through two serial control boards (Mini SSC II). The control boards receive commands from the host computer through a 9600bps serial interface allowing us to issue up to 400 motor commands per second. Since we are using 9 DoFs, we can send a command to each servo every 22.5ms — faster than the frame rate of the cameras. Due to the use of very small, light-weight cameras and fast servo motors, our robot can make saccade-like gaze shifts that come

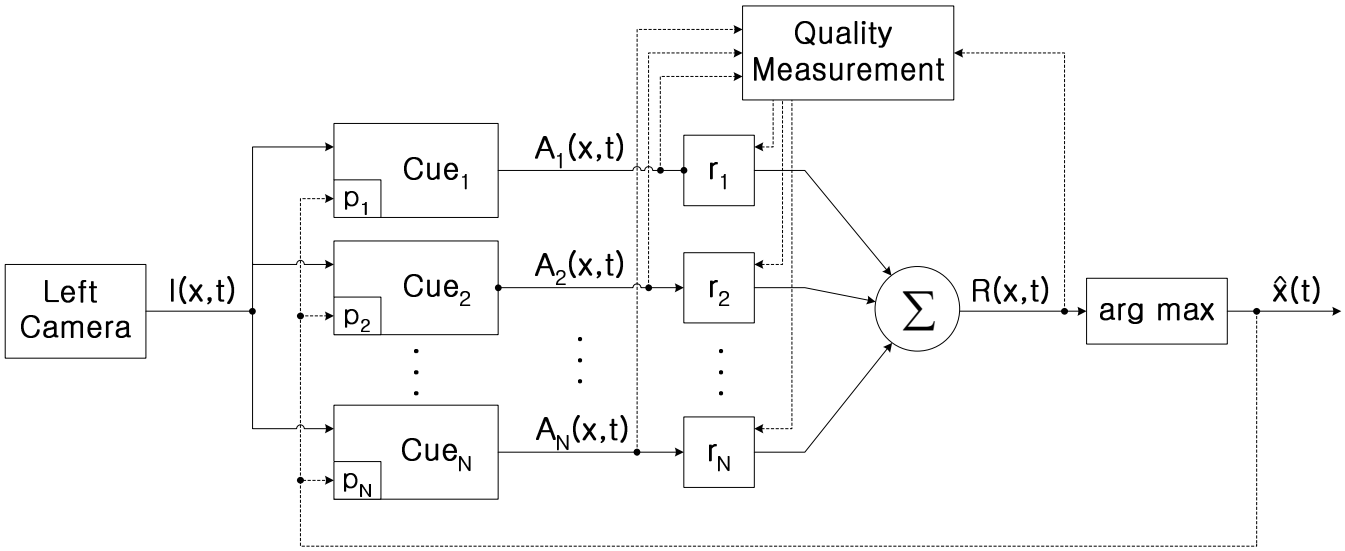


Figure 3: Block diagram of the democratic integration system. Solid lines represent feed-forward information flow, while dashed lines represent adaptation signals.

| | Range | | Resolution | |
|---|---|---|---|---|
| | Hor. | Ver. | Hor. | Ver. |
| Left Eye | 42.8° | 35.3° | 0.37° | 0.28° |
| Right Eye | 37.9° | 33.4° | 0.33° | 0.27° |
| Neck | 180° | 40.2° | 0.71° | 0.17° |

Table 1: Range of motion and average angular resolution of the neck and eye DoFs.

close to the speeds of human saccades. Figure 2 compares the durations of the robot's saccades with those of humans (Becker, 1991). The focus of this paper is smooth pursuit-like tracking, however.

## 3. Adaptive Visual Tracking

Given the lack of a detailed model of the target object we need to have a tracking system that evaluates and adapts itself according to the tracking results. We use the Democratic Integration idea (DI), which is a framework for integration of different visual cues in a self-organized manner (Triesch and von der Malsburg, 2001). This system adaptively fuses several cues into one result. There are two ways in which it is adaptive: first, it allows for a re-weighting of cues based on their estimated reliabilities, second, cues adapt object models to better match the fused result. Such re-weighting can also be observed in human object tracking (Triesch et al., 2001).

The two eyes of our robot head play different roles in the tracking system. The dominant left eye uses an adaptive tracking scheme to estimate the object's position. A vergence controller steers the right eye to the location looked at by the left eye, as described in Section 4.

Figure 3 shows a block diagram of the DI system. We are using four cues (shape, color, motion, prediction). A detailed description of the cues is given in the appendix. Our current implementation of the active DI-style tracking system works in real time (30Hz) with an image resolution of 160x120 pixels.

### *3.1 Democratic Integration System*

In this system $N$ different cues search for features in the image that match the appearance of the object. Each cue $i$ computes a saliency map $A_i(x,t)$ that measures the similarity of the cue's object model $p_i(t)$ to features extracted from the image $I(x,t)$. High values indicate high similarities. Each cue has a reliability $r_i(t)$ that controls the cue's influence on the result saliency map $R(x,t)$. Concretely, $R(x,t)$ is a weighted sum of the cues' individual saliency maps:

$$R(x,t) = \sum_{i=1}^{N} r_i(t) A_i(x,t). \qquad (1)$$

The estimated target position $\hat{x}(t)$ is the point yielding the highest value in the result saliency map, as long as this value exceeds a threshold $T$ (here $T = 0.6$). Otherwise no target is detected:

$$\hat{x}(t) = \arg\max_x \{R(x,t)\}, \text{ if } R(\hat{x},t) > T. \qquad (2)$$

### *3.2 Adaptation of Reliabilities*

The adaptive parameters in the integration of the cues are the reliabilities $r_i(t)$. They reflect a cue's agreement with the result in the recent past. The agreement is measured with the quality $\tilde{q}_i(t)$. A detailed

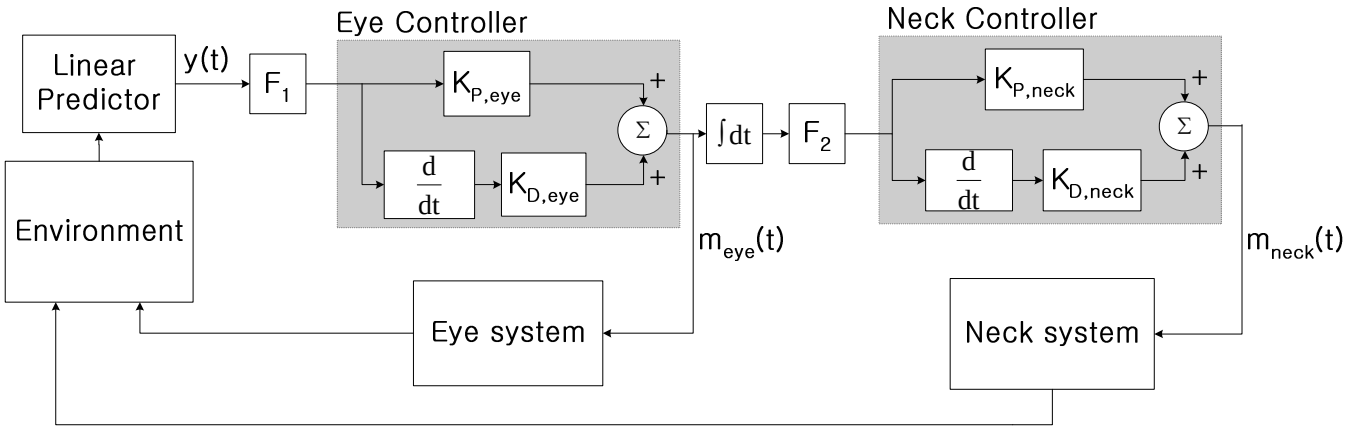


Figure 4: Block diagram of the partitioned controller.

discussion of different quality metrics can be found in (Triesch and von der Malsburg, 2001). Here we measure the quality by comparing the response in the cues' saliency maps at the estimated target position with the average saliency $\bar{A}_i$ of each cue:

$$\tilde{q}_i(t) = \max\left\{0, A_i(\hat{x}, t) - \bar{A}_i(t)\right\}. \qquad (3)$$

If a target has been detected, the reliabilities are updated to follow the normalized qualities $q_i(t)$:

$$\begin{aligned} q_i(t) &= \tilde{q}_i(t) / \sum_{k=1}^{N} \tilde{q}_k(t) \\ r_i(t) &= \alpha_{rel} \cdot q_i(t) + (1 - \alpha_{rel}) \cdot q_i(t-1). \end{aligned} \qquad (4)$$

If no target has been detected, the reliabilities are adapted towards their initial values. $\alpha_{rel} \in [0, 1]$ is a adaptation constant, controlling the speed of adaptation. A value close to 1 leads to a fast adaption, a small value close to 0 causes slow changes. Due to the normalization of the $q_i(t)$, the $r_i(t)$ are also normalized with their sum converging to one.

The adjustment of the adaptation constant $\alpha_{rel}$ is important: If it is too small the system can't react fast enough to the sudden failure of one or more cues. On the other hand, if $\alpha_{rel}$ is too large, this typically also worsens performance.

### *3.3 Adaptation of Cues*

The adaptive cues try to maximize their agreement with the result saliency map by adapting their object models towards features $f_i(t)$ extracted at the estimated target position. This causes a better agreement with the result in the future, provided that the visual appearance of the tracked object is not changing much from one frame to the next. The shape cue for example uses a small grayscale template (e.g. 7x7 pixel) as object model $p_i(t)$ and searches for image locations that are similar to the template. The process of object model adaption uses similar dynamics as (4). If a target has been detected, the cues' object models are adapted according to:

$$p_i(t) = \alpha_{cue} \cdot f_i(t) + (1 - \alpha_{cue}) \cdot p_i(t-1). \qquad (5)$$

If no target has been detected, the cues' object models are adapted towards their initial values. Each cue can have its own adaptation constant $\alpha_i$, but in our experiments they are all identical: $\alpha_i = \alpha_{cue}\, \forall\, i$. The adjustment of the adaptation constant $\alpha_{cue}$ is also important: a constant object model is unlikely to match the object for more than a few frames, but a object model changing too quickly would not carry any information about the stable properties of the object.

### *3.4 Initialization*

As we have mentioned in the Introduction, infants tested shortly after birth already have preferences for some visual stimuli over others. Here, in the initialization of the cues, we incorporate this idea of having preferences for the visual system by setting initial models for the cues. One always needs a priori knowledge about the target to guide initial target selection. For example, a system configured to do face tracking could have its color object model set to skin color initially, or can use an additional face detection cue. Other cues like the motion cue can work without such a prior model and may have high reliabilities right from the start. If we don't have a priori knowledge for a cue we set its reliability to zero and its object model to an arbitrary value. As soon as the other cues agree on a target this cue will be bootstrapped and automatically integrated into the tracking process. It is important to find a set of cues that is sufficient for bootstrapping the system.

## 4. Motor Control System

The robot head has pan and tilt DoF for both eyes and the neck, such that multiple combinations of pan and tilt angles of the eyes and the neck can provide the same orientation of the camera in space. This redundancy needs to be resolved by the controller. Our solution is based on the idea of controller partitioning (Oh and Allen, 2001), which exploits complementary properties of redundant DoFs.

### *4.1 Eye-neck control mechanism*

Since the robot's cameras are very small, the eyes move faster than the neck, where the servo has to move the entire robot head. On the other hand, the neck has a much wider range of movement. The basic idea behind our partitioned controller is that once the tracked object is out of the center of the image, we use the DoF with the faster response (eye) first to reduce the visual error, and then the slower DoF (neck) to bring the eye back to its center position. In other words, we use the neck DoF to compensate for the small range of the eye DoF.

Inside the partitioned controller, we have 2 PD controllers in a cascade. The first controller receives the deviation $y(t)$ of the target object from the center in pixel coordinates. The second controller takes as input the estimated eye position derived from the output of the first controller. Here we explicitly assume that the horizontal and vertical mapping from image coordinates to motor commands are independent from each other and can be controlled separately. This may not be completely accurate due to a number of reasons but is a good approximation. The first PD controller computes the motor command $m_{eye}(t)$ given the visual error:

$$\begin{aligned} m_{eye}(t) &= K_{P,eye} \cdot y(t) + K_{D,eye} \cdot \dot{y}(t) \qquad (6) \\ \dot{y}(t) &= y(t) - y(t-1)\,. \end{aligned}$$

$K_{P,eye}$ is the proportional gain for the eye, and $K_{D,eye}$ the derivative gain.

This motor command is used to estimate the future eye angular deviation from its center position $\hat{x}_{eye}(t)$:

$$\hat{x}_{eye}(t) = \hat{x}_{eye}(t-1) + m_{eye}(t)\,. \qquad (7)$$

Here, we assume linearity and no delays to approximate the eye position $\hat{x}_{eye}(t)$ calculated from the previous motor commands. In other words, we make an assumption that the estimated eye angular deviation from its center position has a one-to-one relationship with the given motor command $m_{eye}(t)$. This approximation works reasonably well as demonstrated in Section 5. Then this estimated deviation is fed to the neck controller, where another PD controller is used to make a neck movement $m_{neck}(t)$, that brings the eye back to its center position:

$$m_{neck}(t) = K_{P,neck} \cdot \hat{x}_{eye}(t) + K_{D,neck} \cdot m_{eye}(t)\,. \quad (8)$$

Figure 5: Left and right image pair before (above) and after (below) vergence control.

Both controllers ignore inputs that are below a certain threshold. The vision system can detect errors that are smaller than the smallest possible corrective movement. The thresholds will suppress small oscillations around the target position. The first threshold ($\pm 2$ pixels for the full image resolution) corresponds to the visual error for the eye controller. The second one ($\pm 1$ ticks) corresponds to the angular error for the neck controller.

A block diagram for the complete controller is given in Figure 4. In this figure, $F_1$ corresponds to the threshold opearation in the eye controller and $F_2$ to the threshold operation in the neck controller. We also use a simple linear predictor in front of the partitioned controller which is basically the same as the prediction cue described in the appendix. One could use a more sophisticated predictor that estimates the target dynamics to improve the tracking performance, as demonstrated in (Shibata and Schaal, 2001b).

Note that due to the feedback delays in the system, the active tracking will always lag behind the target, showing object following behaviors.

### *4.2 Vergence control mechanism*

Since tracking is done with only one "dominant" eye, we need a method for ensuring the coordination of both eyes, i.e. a way of controlling the non-dominant eye to look at the same object. We have adopted and extended previous work by Theimer and Mallot for doing vergence control (Theimer and Mallot, 1994). They extract Gabor jets from the center of left and right image and then compute a phase-based distance estimate in pixel coordinates. This estimate can be used to bring the center of the left and right image to the same target without specific knowledge about the object. This method mimics an array of disparity tuned simple cells as could be found in the primary visual cortex.

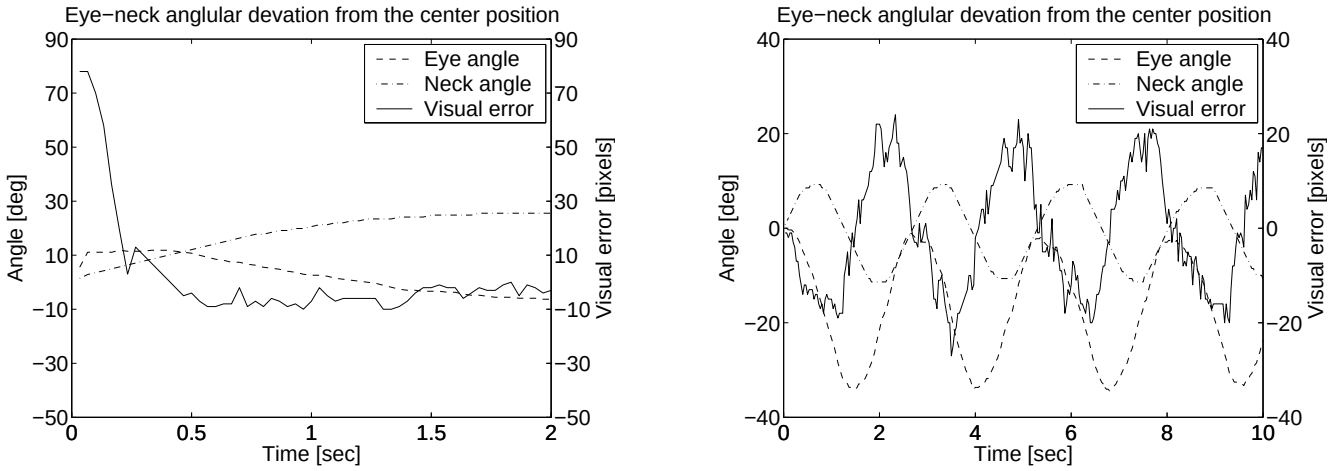


Figure 6: Eye and neck angle deviation from center position due to a step input(left) and a swinging pendulum(right).

Our version extends this idea and extracts a grid of Gabor jets from the right image and compares it to a Gabor jet from the left image. Due to space limitations, we will provide a detailed description in a separate paper. Figure 5 shows the left-right image pairs before and after the vergence control. A red cross is drawn at the center for comparison.

Alternatively, one could solve this vergence problem in the tracking application by having two separate tracking modules for the left and right eye and then have each module control one eye independently of the other. In this case the neck controller would be driven by the average deviation of the left and right eye from their central positions. As long as the two tracking modules share models, both eyes will track the same object. This tracking approach, although more time consuming, is more biologically plausible than the current scheme. Future work will try to combine two DI systems with the vergence control mechanism presented here.

## 5. Experiments

### *5.1 Experimental setups*

We have experimented with two different setups to evaluate the performance of our active tracking system. In the first setup, we let the robot head track a tennis ball in order to evaluate the controller's behavior in response to step inputs and oscillatory inputs. For initialization, we supply the system with color models for the color cue only. It is able to track robustly by bootstrapping the other cues. In the second setup, we track the face of a moving person.

For the DI tracking system, the initial reliabilities of the cues are 0.55 for the motion cue, 0.45 for the color cue and 0 for the shape and prediction cues. We have used a template size of 7x7 for both experiments. Blurring in the motion cue is done using a gaussian filter of size 5x5 for the tennis ball tracking experiment and 7x7 for the face tracking (see appendix). Blurring in the color cue is done using a gaussian filter of size 3x3 for tennis ball tracking and 5x5 for face tracking. The adaptation constant for the dynamics of cue reliability, $\alpha_{rel}$, is set to 0.1 and the adaptation constant for the dynamics of the cues' object models, $\alpha_{cue}$, is set to 0.05.

| | | | |
|---|---|---|---|
| $K_{P,eye,horizontal}$ | 0.189 | $K_{P,neck,horizontal}$ | 0.0189 |
| $K_{D,eye,horizontal}$ | 0.1 | $K_{D,neck,horizontal}$ | 0.1 |
| $K_{P,eye,vertical}$ | 0.284 | $K_{P,neck,vertical}$ | 0.0474 |
| $K_{D,eye,vertical}$ | 0.15 | $K_{D,neck,vertical}$ | 0.1 |
| $K_{P,vergence}$ | 0.075 | | |
| $K_{D,vergence}$ | 0.1 | | |

Table 2: Controller gains used in the experiments.

The gains for the controller were set manually after some experimentation (see Table 2) and are the same for all experiments. For tuning the gains, first, we set the gains for the neck controller equal to zero and moved the eyes only. We increased the gains of the eye controller until we have sufficiently fast response with the overshoot below a certain threshold, say $\pm 5$ pixels. Then we fixed the gains of the eye controller and increased the gains for the neck controller till we have sufficiently small overshoot with fast response time.

### *5.2 Step input*

For the first experiment, we let the system track the tennis ball for a while without moving the robot head. Then we switched on the control system to measure the step response, and plotted the eye and neck angular deviation from its center position along with the visual error. We only show data for the horizontal movements of the

left (dominant) eye and the neck. The object was placed near the horizontal boundary of the image in order to generate a visual error, close to the maximum of ±80 pixels (±35 degrees).

In the left side of Figure 6, the step input is given at 0 s. In other words, we turned on the control system at 0 s and it clearly shows that the eye will first start to move in the direction of the object and the neck will follow the eye. Near the end of the movement, the eye will move in the opposite direction of the neck movement since the neck movements will cause the eye to make compensatory movements. There is an overshoot in the visual error at the end of the movement. An extra mechanism like the vestibulo ocular reflex can be used to alleviate the problem (Shibata and Schaal, 2001a).

### *5.3 Oscillatory input*

To create an oscillatory input, we used a tennis ball as a pendulum with an amplitude of approximately ±40 pixels. We let the system adapt its object models while actively tracking the object. The right side of Figure 6 shows that the eye immediately starts moving to reduce the visual error. The neck slowly follows the eye movement. Due to the feedback and visual processing delays we can see the visual error follows the oscillation of the swinging ball with the same frequency.

### *5.4 Face tracking*

For the face tracking experiment, we used skin color as the initial model for the color cue. Note that the color cue alone is not sufficient for robust face tracking, if other objects with a similar color are present in the scene.

Figure 8 shows every 20th frame of an image sequence captured from the left camera while tracking faces. A blue cross marks the center of the image and a red cross the object position estimated by the DI tracking system. Video clips are available on our website[1].

One of the nice things about the DI tracking system is that it self-evaluates its cues and finds out how useful each of the cues is. Figure 7 shows that the system is able to automatically suppress the simple motion cue based on difference images, as soon as it becomes useless when the robot head starts to move.

## 6. Discussion

We have presented an adaptive, model-free tracking system implemented on an anthropomorphic robot head. Our goals were fourfold: First, the tracking system should be model-free, because we want to use it in the context of a learning system that develops object representations completely autonomously. Second, despite the lack of a detailed object model, the system should

[1] http://csclab.ucsd.edu/publications/siab04_facetracking.mpg

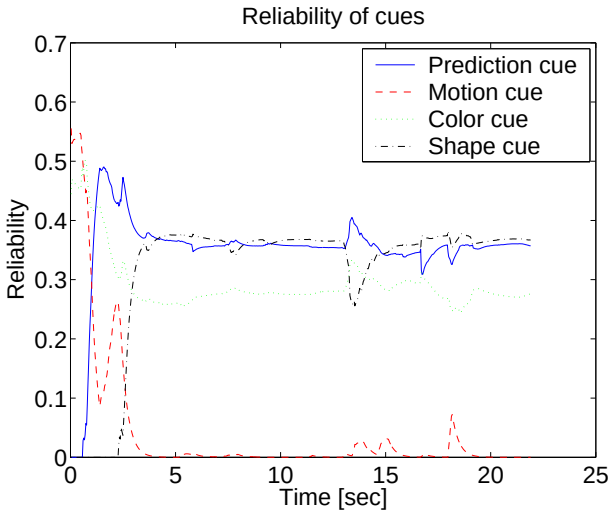


Figure 7: Reliability of cues as a function of time in face tracking experiment.

function robustly and handle fast movements over a wide range. Third, the system should operate in real-time, which in our case meant a frame rate of 30Hz. Fourth, the system should be biologically plausible because it will ultimately be a part of a larger embodied model for the development of object representations in the primate visual system. The system we have presented in here essentially meets all these requirements.

Our tracking system is based on the democratic integration idea (Triesch and von der Malsburg, 2001). The democratic cue integration used in the tracking system has the ability to bootstrap representations autonomously and does not require a detailed model of the object to be tracked (although some coarse prior knowledge is needed to specify which objects are "interesting"). It has been argued that this architecture mimics the robustness and flexibility known from biological vision systems. Furthermore, fast re-weighting effects as produced by the Democratic Integration mechanism can be readily observed in object tracking by humans (Triesch et al., 2001).

In the context of active tracking as demonstrated here, it is interesting to note that the system was fairly robust to the camera movements, despite its reliance on a simple motion detection cue based on difference images. This can be attributed to the democratic integration mechanism which automatically suppresses the motion detection cue during times of fast camera movements.

The idea of cue integration or sensor fusion has been very popular in diverse areas of computer vision. Helpful reviews, introductory papers and books published on the topic are available (Luo and Kay, 1989, Alomoinos and Shulman, 1989, Clark and Yuille, 1990, Abidi and Gonzalez, 1992). A number of different theoretical frameworks

Figure 8: Sequence of images (top left to bottom right) of tracking a face. The red crosses indicate the estimated target position. The dark blue crosses mark the center of the image.

are being used for cue integration, most notably probability and information theoretic approaches using Bayesian inference (Pearl, 1993, Bayes, 1783, Manyika and Durrant-Whyte, 1994), Dempster-Shafer evidential reasoning (Shafer, 1976, Hutchinson and Kak, 1992), and possibility theory (Zadeh, 1978, Dubois and Prade, 1992). In the areas of visual tracking, (Triesch and von der Malsburg, 2001) shows the robustness of the system with only weak visual cues. (Denzler et al., 2002) have combined DI-style cue integration with probabilistic fusion to work with multiple cameras for object tracking. While the Democratic Integration idea was originally introduced in the context of tracking, it is worth mentioning that it has also recently been successfully applied to the problem of model-free segmentation from video streams (Hayman and Eklundh, 2002). The potential benefits of such an approach for the purpose of autonomous learning of object representations is obvious. We will explore this issue in future work.

In order to exploit the complementary nature of the robot's redundant degrees of freedom (fast eye movements with only small kinematic range and slower neck movements with wide kinematic range) we used the idea of controller partitioning (Oh and Allen, 2001). This simple method works very effectively and is likely to be sufficient for our purposes. However, a number of improvements could easily be made if higher accuracy was required (e.g. for a highly foveated vision system). Most interesting to us seems the option of adding a feedback error learning component to the partitioned controller (Kawato, 1990). From the standpoint of biological plausibility, the partitioned control scheme is likely to be far simpler than the control mechanisms actually used by the primate visual system. At present, there seems to be little data about the coordination of neck and eye movements during tracking tasks. However, data from natural tasks where eye, head, and hand movements are recorded simultaneously suggests rather complicated forms of coordination between eyes, neck, and hands (Hayhoe et al., 1999).

Since tracking is only done with one "dominant" eye, we implemented a vergence control method to ensure coordination of both eyes. Our method is an extension to the one proposed in (Theimer and Mallot, 1994). It is reasonably accurate but currently needs several frames to make the eyes converge on the same point.

While none of the individual components of the system is entirely new, we consider their integration into our anthropomorphic robot head as a significant achievement. Future work will extend this system by adding the ability to autonomously learn and maintain representations for the objects in the robot's environment — bringing us one step closer to an embodied account of the development of the primate visual system.

## Acknowledgments

This work was supported by the National Science Foundation under grants 0208451, 0233200, and 0329287, the UC Davis M.I.N.D. institute, and the National Alliance for Autism Research. We would like to thank the developers of the OpenCV libraries for making their software publicly available. We also thank Alan Robinson and Erik Murphy-Chutorian for comments on an earlier draft.

## Appendix: Description of Individual Cues

This section describes the detailed specification of the cues. Figure 9 shows the saliency maps of the individual cues for a typical frame during tennis ball tracking. Most cues operate on grayscale images, only the color cue uses

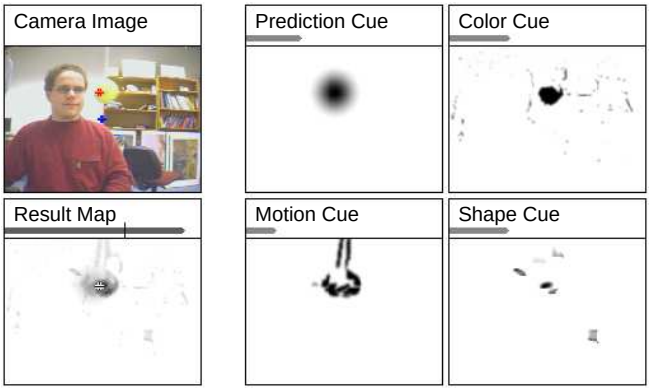


Figure 9: Saliency maps of the different cues during the tracking of swinging tennis ball. The length of the gray bars above the cue saliency maps represent the reliabilities of the cues. The bar above the result saliency map indicates the maximum saliency.

color images.

### *Motion Cue*

The motion cue does not have any adapting object models and can be used for bootstrapping the other cues. It takes the current and the last input image and computes their difference image:

$$D(x,t) = |I(x,t) - I(x,t-1)|\,. \tag{9}$$

All positions with differences over a threshold $T_{motion} = 15$ are set to one, all others to zero:

$$\tilde{A}(x,t) = 1 \text{ if } D(x,t) > T_{motion},\ 0 \text{ otherwise.} \tag{10}$$

This thresholded difference image is convolved with a gaussian filter to suppress camera noise and detect larger blobs of motion:

$$A(x,t) = \tilde{A}(x,t) * \mathcal{G}(x) \tag{11}$$

The specific kernel size of the smoothing filter depends on the approximate size of the target. This motion cue is useful to detect moving targets as long as the cameras are stationary.

### *Color Cue*

The color cue operates in YUV color space. It creates a two dimensional lookup table $L(u,v,t) \in [0,1]$ that contains a similarity value for every color in the discretized UV-color space.

This lookup table forms the object model of the color cue, along with a mean $\bar{Y}$ and deviation $\sigma_Y$ of the brightness of the target object.

Every pixel $x$ of the saliency map gets the similarity value that is assigned to the color found at the location in the source images, as long as its brightness is in the accepted range:

$$\tilde{A}(x,t) = \begin{cases} 0, \text{ if } \left|I(x,t) - \bar{Y}\right| > \sigma_Y \\ L\left(U(x,t), V(x,t), t\right) \text{ otherwise.} \end{cases} \tag{12}$$

This saliency map is convolved with a gaussian filter to detect larger blobs of homogeneous color (11). The size depends on the expected size of the target object.

The first step of adapting the color object model is extracting the average color from a 3x3 image patch around the target position $\hat{x}(t)$.

This color value is converted to polar coordinates, resembling hue and saturation values $\bar{h}$ and $\bar{s}$. If the standard deviation of $\bar{h}$ in this image patch is smaller than a predefined threshold $\sigma_h$, the new object model is computed according to the following equation:

$$f(h,s,t) = \begin{cases} 1, \text{ if } \left|h - \bar{h}\right| < \sigma_h \wedge |s - \bar{s}| < \sigma_s \\ 0, \text{ otherwise} \end{cases} \tag{13}$$

This range of similar colors has a rectangular shape in polar coordinates $h$ and $s$, which is approximated with a pie slice polygon in the cartesian UV color space. This is plotted into the new lookup table $f(x,t)$, which is blended with the lookup table $p(x,t)$ according to (5).

### *Prediction Cue*

The simple prediction cue tries to estimate the object location with a linear prediction based on the last two target positions:

$$\hat{X}(t) = \hat{x}(t-1) + (\hat{x}(t-1) - \hat{x}(t-2))\,. \tag{14}$$

The saliency map is defined as a gaussian blob with $\sigma = 10$ at the estimated position:

$$A(x,t) = \exp\left(-\left(x - \hat{X}(t)\right)^2 / (2\sigma^2)\right). \tag{15}$$

If the last target positions are unknown, or the predicted position is outside of the image boundaries, the whole saliency map is set to 0.1. This cue doesn't have a object model, the adaptation lies in the linear prediction.

### *Shape Cue*

The shape cue computes the similarity of its shape model, a gray level template $P(x,t) = p_{shape}(x,t)$, to image patches of the same size around every pixel using a normed squared difference metric:

$$\tilde{A}(x,t) = \frac{\sum_{x'} \left(P(x',t) - I(x+x',t)\right)^2}{\sum_{x'} \left(P(x',t)^2 \cdot I(x+x',t)^2\right)}. \tag{16}$$

This metric provided by the OpenCV template matching function has the lowest values for the highest similarity. In order to get a saliency map with values between 0 and 1 we compute:

$$A_{min} = \min_x \tilde{A}(x,t), \text{ and } A_{max} = \max_x \tilde{A}(x,t)$$

$$A(x,t) = \max\left\{0,\ 1 - \frac{(\tilde{A}(x,t) - A_{min})}{K \cdot (A_{max} - A_{min})}\right\}, \quad (17)$$

with $K = 0.002$. A template adapted to a non-homogeneous surface leads to a very peaked response and is useful to precisely track a spot on the target object. For the adaptation of the prototype template the cue extracts an image patch centered at the target position in the input image. This image patch is blended to the current object model template as in (5).

## References


Abidi, M. and Gonzalez, R., (Eds.) (1992). *Data Fusion in Robotics and Machine Intelligence*. Academic Press.

Alomoinos, Y. and Shulman, D., (Eds.) (1989). *Integration of Visual Modules*. Academic Press.

Bayes, T. (1783). An essay towards solving the problem in the doctrine of chances. *Philosophical Transactions of the Royal Society*, 53:370–418.

Becker, W. (1991). Saccades. *Vision & Visual Dysfunction Vol. 8:Eye movements*, pages 95–137.

Clark, J. J. and Yuille, A. L., (Eds.) (1990). *Data Fusion for Sensory Information Processing Systems*. Kluwer Academic Publishers.

Denzler, J., Zobel, M., and Triesch, J. (2002). Probabilistic integration of cues from multiple cameras. *Dynamic Perception, pp.309–314*.

Dubois, D. and Prade, H. (1992). *Data Fusion in Robotics and Machine Intelligence*, chapter Combination of fuzzy information in the framework of possibility theory, pages 481–505. Academic Press.

Hayhoe, M., Land, M., and Shrivastava, A. (1999). Coordination of eye and hand movements in a normal environment. *Invest. Opthalmol. & Vision Science*, 40.

Hayman, E. and Eklundh, J.-O. (2002). Probabilistic and voting approaches to cue integration for figure-ground segmentation. *ECCV (3)*, pages 469–486.

Hutchinson, S. and Kak, A. (1992). *Data Fusion in Robotics and Machine Intelligence*, chapter Multisensor strategies using dempster-shafer belief accumulation, pages 165–209. Academic Press.

Kawato, M. (1990). Feedback-error-learning neural network for supervised motor learning. *In Eckmiller R (Ed.) Advanced Neural Computers. Elsevier, North-Holland*, pages 365–372.

Kim, H., York, G., Burton, G., Murphy-Chutorian, E., and Triesch, J. (2004). Design of an anthropomorphic robot head for studying autonomous development and learning. *International Conference on Robotics and Automation (ICRA), accepted*.

Luo, R. C. and Kay, M. G. (1989). Multisensor integration and fusion in intelligent systems. *IEEE Trans. on Systems, Man, and Cybernetics*, 19(5):901–931.

Manyika, J. and Durrant-Whyte, H., (Eds.) (1994). *Data Fusion and Sensor Management: A Decentralized Information-Theoretic Approach*. Ellis Horwood, Chichester.

Oh, P. Y. and Allen, P. K. (2001). Visual servoing by partitioning degrees of freedom. *IEEE Trans. on Robotics and Automation*, 17(1).

Pearl, J., (Ed.) (1993). *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann Publishers, 2nd edition.

Shafer, G., (Ed.) (1976). *A Mathematical Theory of Evidence*. Princeton University Press.

Shibata, T. and Schaal, S. (2001a). Biomimetic gaze stabilization based on feedback-error learning with nonparametric regression networks. *Neural Networks*, 14(2):201–216.

Shibata, T. and Schaal, S. (2001b). Biomimetic smooth pursuit based on fast learning of the target dynamics. *IEEE Int. Conf. on Intelligent Robots and Systems*.

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

Triesch, J., Ballard, D., and Jacobs, R. (2001). Fast dynamics of visual cue integration. *Perception*, 31:421–434.

Triesch, J. and von der Malsburg, C. (2001). Democratic integration:self-organized integration of adaptive cues. *Neural Computation*, 13(9):2049–2074.

Zadeh, L. (1978). Fuzzy sets as a basis for a theory of possibility. *Fuzzy Sets and Systems*, 1:3–28.