Online Learning of a Motor Map for Humanoid Robot Reaching

Chris Gaskett and Gordon Cheng

Department of Humanoid Robotics and Computational Neuroscience,
ATR Computational Neuroscience Laboratories, Kyoto, Japan
{cgaskett,gordon}@atr.co.jp, http://www.cns.atr.co.jp/hrcn/

Abstract

We propose a control system for humanoid robot reaching using a motor-motor mapping that is learnt online. The system combines endpoint closed-loop and open-loop visual servo control. The closed-loop component moves the eyes, head, arm, and torso, based on the position of the target and the robot's hand, as seen by the robot's head mounted cameras. The learned, open-loop component brings the hand into view.

1 Introduction

Reaching is a hand-eye coordination task involving vision, tracking of moving objects, depth perception; and dynamic, high-speed movement of the whole body, including the eyes. Our experimental platform is a hydraulically driven full-body humanoid robot with 30 degrees of freedom, anchored at the hips to a support. The robot is shown in Figure 1. For reaching, we control 14 joints: 4 eye joints, 3 neck joints, 3 torso joints, and 4 arm joints; based on the view from both cameras. Our vision system tracks colour blobs [1], and provides the location of the target object (coloured pink) and the hand (coloured yellow), in pixels at 60Hz (see Figure 2).

Our control system for reaching does not rely on camera calibration or accurate knowledge of the robot's kinematics. When the hand is visible the system performs accurate endpoint closed-loop control. It uses uncalibrated stereo vision to compare depths, but still functions when the target object or hand are only seen by one eye.

When the hand is completely occluded, the controller can still reach for the target using endpoint open-loop control. The open-loop system is based on learned motor-motor relationships [2] between the eye, head, and arm joints, represented by a Kohonen Self Organizing Map (SOM) [3].

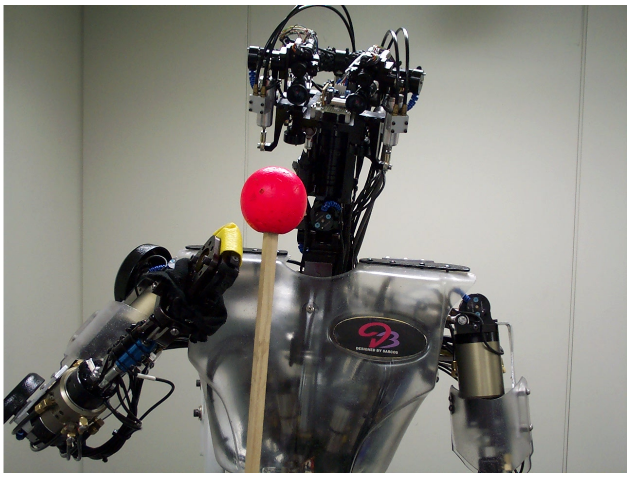
Figure 1: The robot reaching for the target

2 Hand-Eye Coordination

Continuously updating the position of part of a robot based on visual information is known as *visual servo control* [4]. The mapping relating the position of the target object, in the pixels coordinate frame of the cameras, to robot joint positions can be found by using a camera model and calibration to convert the position of the target into a fixed coordinate frame, then using knowledge of the robot's kinematics to convert the position into robot joint angles; or through learning. A system suitable for humanoid robot must account for the movements of the robot's eyes and head.

Marjanović et al.'s [5] reaching system for an upper body humanoid controlled 2 arm position parameters based on 2 eye/head parameters (pan, tilt). It used 3 learned mappings: pixels to eye/head motors (sensor-motor mapping); eye/head motors to arm motors (motor-motor mapping); and arm motors to eye/head motors. Each map had 2 input and 2 output variables. Corrections to the eye/head to arm motor-motor map were performed without needing to gaze at

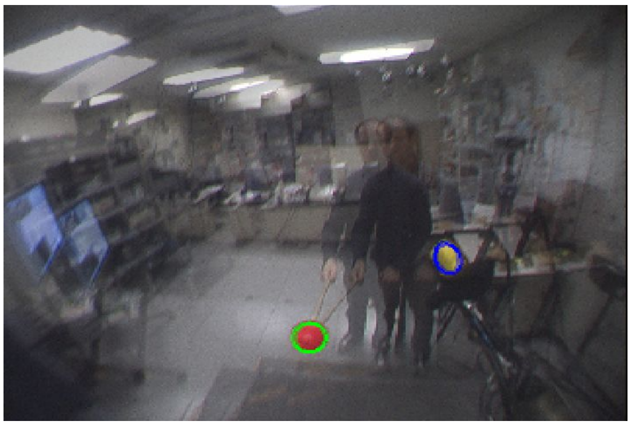
Figure 2: Superimposed left and right camera views

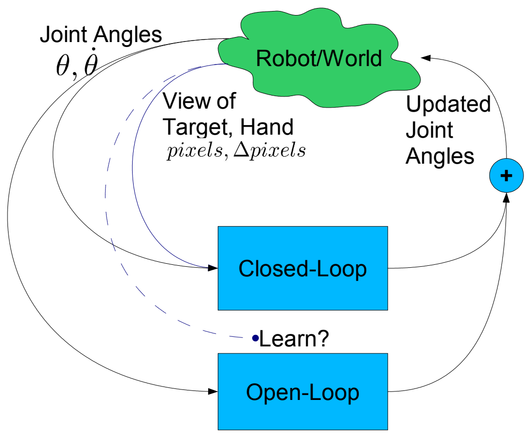

Figure 3: Top level control architecture

the hand, but each attempt by the robot to reach the target only produced one update to the maps. Learning was slow; however, the robot was able to reach towards targets in a region of its view.

Rougeaux and Kuniyoshi's [6] system was based on the assumption that the eyes will be looking at the target. It used a motor-motor mapping between 4 eye and head joint angles and 3 arm parameters. The mapping was learnt during a training phase in which the eyes watched the hand. After the training phase, there was no error correction mechanism.

Metta et al. [2] also developed a controller based on a single motor-motor mapping. The system mapped from 2 eye/head parameters to 2 arm control parameters; a later version added eye vergence allowing control of depth [7]. Instead of using additional mappings [5], correcting mapping errors required that the robot redirect its gaze to look at the hand after looking at the target object.

In the systems described, imperfections in the learned mapping cause errors in the hand positioning. Although some of the methods include online error correction, it is a slow process of learning, rather than a mechanism for reaching the target position quickly. The *endpoint closed-loop* strategy, as opposed to *endpoint open-loop* control, allows hand positioning accuracy independent of the errors in hand-eye calibration [4]. Endpoint closed-loop control compensates quickly for imperfections in the mapping by comparing the *view of the hand* with the view of the target—the view of the hand provides performance feedback.

Several systems using *fixed cameras* have used learning and endpoint closed-loop control [8–11]. Consequently, they can perform *two types of error correction*: slow correction of the learned mappings, and fast correction of errors in hand positioning through endpoint closed-loop control. Although endpoint closed-loop control seems superior to endpoint open-loop control, we cannot assume that the hand will be visible at all times, especially in moving camera systems. For example, a humanoid robot looking ahead can't see that its hands are by its sides.

3 A Hybrid Open and Closed-Loop Control System

Our system uses a combination of endpoint closed-loop and open-loop control (Figure 3). The endpoint closed-loop control system for the arm is unified with the controller for the eyes, head and torso. The open-loop control system is used when the eyes can see the target, but not the hand. A learned, motor-motor mapping is used to position the hand within view, so that the closed-loop controller can operate. The mapping represents joint configurations that place the hand in the center of the field of view, and is updated online when the hand passes through the field of view (indicated as Learn? in Figure 3).

3.1 Closed-Loop Control System

Our closed-loop control system uses only a qualitative idea of the relationships between joints and sensors. It is a non-learning network of proportional derivative (PD) controllers. Errors caused by poor modelling are reduced over time through feedback and redundancy.

The purpose of the control system is to look at the target with both eyes, move the hand toward the target, and move the head and torso to assist the eyes

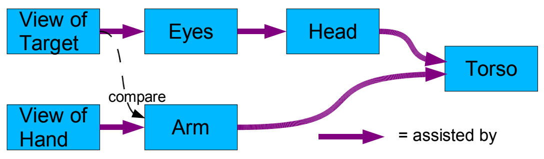

Figure 4: Assistive relationships in the closed-loop control system

and hand. The assistive relationships are shown in Figure 4. At a lower level, we define a network of assistive relationships for each joint. Each joint's purpose is to *relax* to its relaxation position, and also to assist other joints and *blobs* seen by the vision system toward their relaxation positions.

The continual process of relaxation helps to maintain a natural pose. When the target is not visible, relaxation slowly brings the robot back to an upright posture. When the target is visible, the influence of relaxation draws the hand to touch the underside of the target object. Without relaxation, the control system drives the hand into the target object and pushes it along, or occludes the target with the hand.

We define the *desired change* for self-relaxation, D, for each joint,

$$D_{joint} = \left(\theta^*_{joint} - \theta_{joint}\right) - K_d \dot{\theta}_{joint}, \qquad (1)$$

where K_d the derivative gain for joints; θ is the joint angle; and the asterisk indicates the relaxation position.

The desired change for a vision blob is:

$$D_{blob} = (x^*_{blob} - x_{blob}) - K_{dv} \dot{x}_{blob}, \qquad (2)$$

where K_{dv} is the derivative gain for vision blobs; and x is position in pixels.

The following describes the control systems for three example joints, left eye pan, head tilt, and elbow:

The purpose of the *left eye pan* ($\mathsf{L}EP$) joint is to move the target into the center of the left camera's field of view:

$$\dot{\theta}_{\mathsf{L}EP} = K_p \times \Big[K_{\text{relaxation}} D_{\mathsf{L}EP} - K_{\text{target}\rightarrow EP} K_v C_{\mathsf{LXtarget}} D_{\mathsf{LXtarget}}\Big], \qquad (3)$$

where L and R represent left and right; X represents the x pixels axis; K_p is the proportional gain; K_v is the proportional gain for vision blobs; C_{blob} is the tracking confidence for that blob.

The *head tilt joint* (HT), which tilts the head from side to side, moves to equalise the pan (EP) and tilt (ET) of the eyes:

$$\dot{\theta}_{HT} = K_p \times \Big[K_{\text{relaxation}} D_{HT} - K_{EP\rightarrow HP}\left(D_{\mathsf{L}EP} - D_{\mathsf{R}EP}\right) - K_{ET\rightarrow HT}\left(D_{\mathsf{L}ET} - D_{\mathsf{R}ET}\right)\Big]. \qquad (4)$$

The *elbow* ($\mathsf{L}EB$) serves to equalise the left-right disparity of the view of the hand with the disparity of the target:

$$\dot{\theta}_{\mathsf{L}EB} = K_p \times \Big[K_{\text{relaxation}} D_{\mathsf{L}EB} - K_{EB} K_v C_{\mathsf{LXtarget}} \cdot C_{\mathsf{LXhand}} \cdot (D_{\mathsf{LXtarget}} - D_{\mathsf{LXhand}}) - K_{EB} K_v C_{\mathsf{RXtarget}} \cdot C_{\mathsf{RXhand}} \cdot (D_{\mathsf{RXtarget}} - D_{\mathsf{RXhand}})\Big], \qquad (5)$$

where RX is in the opposite direction to LX for symmetry.

The control architecture (Figure 3) does not rely on the use of this particular closed-loop control system; any control system that fixates on and reaches for the target is suitable.

3.2 Endpoint Open-Loop Control System

When the hand is not visible, the robot estimates the required position of the hand using knowledge of its kinematics. In human terms, it is relying on *proprioception.*

We assume that the closed-loop control system is fixating the eyes on the target, then calculate the required position of the hand through a motor-motor mapping, without considering the position of the target in pixels [2, 6]. The mapping represents a *constraint*: joint configurations that place the hand in the centre of the field of view of both eyes.

Our system learns the mapping while reaching for the target. During reaching, the target object should be close to the centre of the field of view, and the hand should be close to the target object. Consequently, the hand should be close to the centre of the field of view. When the hand is visible, the map is trained with a list of joint angles for the eyes, head, and arm. Torso angles are not required since the head and the arm are fixed to the torso. The learning rate is multiplied by a function of the distance (in pixels) between the hand and the centre of the field of view, so that learning emphasises training examples that most closely satisfy the constraint.

Two minutes of movement is enough to train a Kohonen Self Organizing Map (SOM) [3] to useful accuracy. We use a SOM with a 3D lattice ($10 \times 8 \times 5$), instead of the usual 2D, since the robot's hand moves freely in three-dimensional space [8].

The map is used when the hand is lost from view. Based on the current eye, head, and arm joint angles the *best matching unit* (BMU) in the SOM is selected, based on Euclidean distance:

$$BMU = \arg\min_i \left(\|x - m_i\| \right), \tag{6}$$

where m are the SOM units, and x is the vector of current joint angles.

The BMU gives a joint configuration in which the hand is near the centre of the field of view. The arm joint angles are updated to be closer to the angles specified by the BMU; the eye and head angles are not modified. New BMUs are selected continuously, and the hand moves towards the centre of the field of view.

When searching for the BMU, we include all 11 eye, head, and arm joint angles, rather than just the 7 eye and head angles. This addresses the robot's redundant joint configurations: the search for the BMU is likely to find a configuration close to the current arm configuration. However, if the selected unit represents a posture in which the joint angles of the arm are almost the same, but the eye and head angles are different, there is some risk that the arm could become stuck in that pose. The sticking cannot occur if the arm angles are not included in BMU selection, but that would not address redundant arm-joint configurations. To avoid sticking but address redundancy, we weight the arm joint components less than the head and eye joints during BMU lookup:

$$BMU' = \arg\min_i \left(\|x - m_i\|_w \right), \tag{7}$$

where w is a weighting vector, with 1 for head and eye joints, and < 1 for arm joints; and $\|x - m_i\|_w = \sqrt{(x-m_i)^{\mathrm{T}} w (x-m_i)}$.

Rather than representing an input to output mapping, like a perceptron, a SOM has no fixed division between inputs and outputs. Thus, the learned SOM can also be used to make the robot look at its hand by mapping arm angles to eye and head angles.

Is not practical to try to experience or exactly store all possible joint configurations that place the hand in front of the eyes. The SOM only approximates and generalises based on what has been observed. If whole areas of the joint space have not been covered then the SOM will not perform sophisticated extrapolation; it will return the closest configuration it has experienced. However, there is no need to be exact. The open-loop controller only needs to be accurate enough to ensure that it can position the hand within view—in practice, it can position the hand close to the centre of the field of view. Ensuring *accurate* reaching is the role of the closed-loop controller.

3.3 Combined Closed and Open-Loop Control

When the hand becomes visible or is lost from view there is a transition between the endpoint open-loop and closed-loop controllers. If the transition is performed instantaneously the arm movement is abrupt.

Instead of switching abruptly, we blend from one controller to another [12]. The proportions are based on the confidence we have in the sensed information and the degree of disturbance we expect from making transitions. When an object (either the target or the hand) comes into view the system increases its measure of confidence for that object over half a second. During that time, the influence of the open-loop controller decreases, while the influence of the closed-loop controller increases. When an object is lost from view the corresponding closed-loop controller's influence is reduced to zero, while the open-loop controller's increases. The confidence measure (C_{blob} in Equations (3) and (5)) is updated independently for each object, as seen by each camera.

The control system is implemented in Scheme, under the DrScheme environment [13], and generates actions at 210Hz.

4 Results

The learned motor-motor mapping had sufficient coverage to be useful after 2 minutes of online learning. For testing, we attached a sensor for measuring cartesian position to the hand. To assess coverage we moved the hand sequentially to each of the 400 positions represented by the SOM. Figure 5 shows that the vertical and horizontal coverage was good, but that there was little coverage of depth. We also confirmed that the head eyes look at the hand in each position. The total range of movement during reaching is higher than shown because the closed-loop control system moves the torso and has better depth control than the open-loop control system.

Hand positions during reaching for a static target are shown in Figure 6. Before reaching commenced, the eye(s) were fixated on the target. When both eyes are used, and the hand is visible, the hand traveled

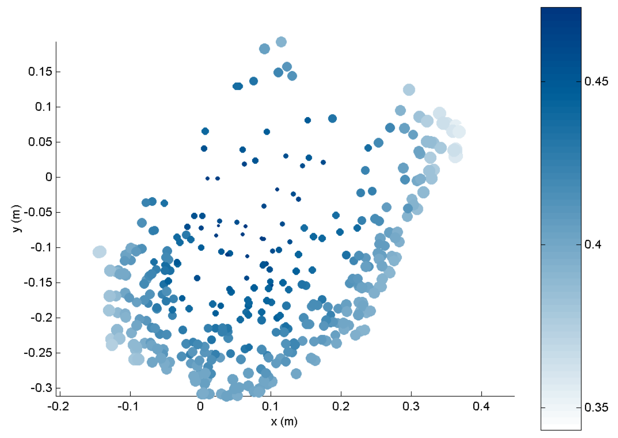

Figure 5: Hand position in joint configurations represented by the SOM. Positive y is upwards; positive x is to the right from the robot's point of view; points with greater depth (z) are darker and smaller. The robot's shoulder is at zero.

74cm to touch the target in 0.8 seconds. When the hand is covered, so that only the endpoint open-loop controller can be used, the final positioning error between the hand and the target was about 20cm.

With one eye covered, as well as the hand, depth information was lost and the final positioning error grew to 47cm. The starting position of the hand also changed because the posture for fixation changes when one eye is covered. We also evaluated performance during transitions between target and hand visibility, in one or both eyes. The system degrades gracefully in such cases and blends open-loop and closed-loop control depending on the information available.

5 Discussion

The closed-loop and open-loop controllers are complimentary: The open-loop controller can operate when the closed-loop control controller cannot; the closed-loop controller has higher accuracy. Additionally, closed-loop control generates training data for the open-loop controller without distorting the behaviour of the robot, e.g. making the robot look at its own hand instead of the target, or requiring specific behaviour from a trainer. Although design and tuning of the closed-loop controller was not difficult, it would be an improvement if the closed-loop controller could be refined online.

Feedback Error Learning also unifies a non-learning closed-loop controller and a learned open-loop controller [14]. It learns an inverse model that helps to compensate for errors during closed-loop control. The goal of our system is different; the learned mapping covers as much of the state space as possible so that the open-loop controller can operate by itself when closed-loop control is not available.

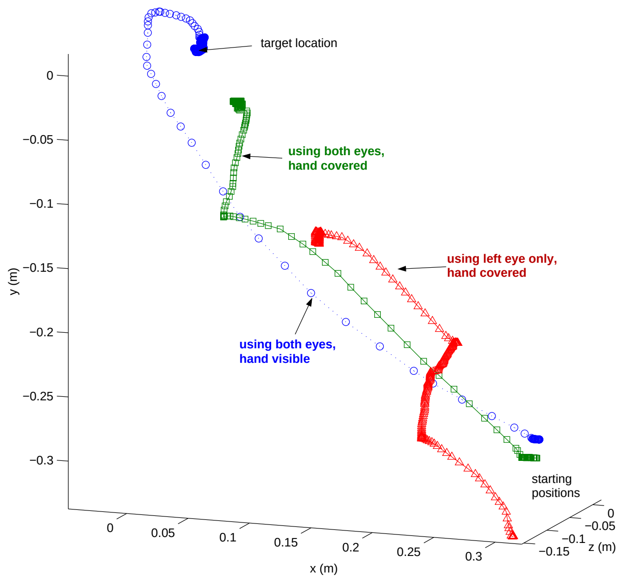

Figure 6: Hand positions during three attempts to reach a target at position zero. Markers are 20ms apart.

The Self Organizing Map that represents the open-loop controller generalises well after observing just a few minutes of closed-loop reaching behaviour, and new training data is generated constantly for further updates. Performance could be improved by reducing or redefining the inputs to the SOM, instead of using all of the joint angles. Other learned hand-eye coordination work has followed this approach, or used robots with fewer degrees of freedom. Nevertheless, including all the joint angles shows which relationships the learning system is finding between joints and allows redundant positioning solutions to be represented. The redundant solutions allow the robot to find a solution near to its current joint configuration, rather than always choosing the same pose for the same target object position.

The system's depth perception during closed-loop control could be improved through exploiting the robot's narrow-field of view cameras, rather than only the wide-angle cameras [15]. Also, we do not exploit the size of the target object as a cue. Comparing the size of the target object to the size of the hand could be especially useful.

Currently, our control system performs reliably as a component of an visual-tracking system [15]. We hope to include the reaching system as a component of other systems in the future.

6 Conclusion

We have implemented a practical control system for performing reaching using stereo vision on a humanoid robot. We avoided Cartesian coordinates and trigonometric functions; the only units are pixels, joint angles, and time. 14 degrees of freedom were used: 4 eye joints, 3 neck joints, 3 torso joints, 4 arm joints. The system is tolerant of occlusion of either the target object or its hand from the view of either eye. When the hand is visible, the system performs high accuracy endpoint closed-loop control. Otherwise, it positions the hand using a learned motor-motor mapping. The learned mapping can be acquired in a few minutes, without requiring a separate training phase. The combination of open and closed-loop, learned and non-learned control produces robust and accurate control without calibration or labour-intensive training procedures.

Acknowledgements

We thank Dr. Aleš Ude for providing us with the vision system; Mr. Keegan Wade, Mr. Mike Lin, and Mr. Jeff N. Carter for assisting with experiments; and the other members of HRCN for their support and suggestions. This research was supported in part by the Communications Research Laboratories.

References

[1] A. Ude and C. G. Atkeson, "Probabilistic detection and tracking at high frame rates using affine warping," in *Proc. of the 16th International Conference on Pattern Recognition, ICPR2002*, QC, Canada, 2002.

[2] G. Metta, G. Sandini, and J. Konczak, "A developmental approach to visually-guided reaching in artificial systems," *Neural Networks*, vol. 12:pp. 1413–1427, 1999.

[3] T. Kohonen, *Self-Organization and Associative Memory*, Springer, Berlin, third edn., 1989, [First edition, 1984].

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

[5] M. Marjanović, B. Scassellati, and M. Williamson, "Self-taught visually guided pointing for a humanoid robot," in *Proc. of the Fourth International Conference on Simulation of Adaptive Behavior*, MA, 1996.

[6] S. Rougeaux and Y. Kuniyoshi, "Robust tracking by a humanoid vision system," in *Proc. of the First International Workshop on Humanoid and Human Friendly Robotics*, Tsukuba, Japan, 1998.

[7] G. Metta, F. Panerai, R. Manzotti, and G. Sandini, "Babybot: an artificial developing robotic agent," in *Proc. of From Animals to Animats: Sixth International Conference on the Simulation of Adaptive Behavior (SAB 2000)*, Paris, 2000.

[8] H. Ritter, T. Martinetz, and K. Schulten, *Neural Computation and Self-Organizing Maps: An Introduction*, Addison Wesley, 1992.

[9] J. A. Walter and K. J. Schulten, "Implementation of self-organizing neural networks for visuo-motor control of an industrial robot," *IEEE Transactions on Neural Networks*, vol. 4(1):pp. 86–95, 1993.

[10] J. R. Cooperstock and E. E. Milios, "Self-supervised learning for docking and target reaching," *Robotics and Autonomous Systems*, vol. 11(3–4):pp. 243–260, 1993.

[11] M. Blackburn and H. Nguyen, "Learning in robot vision directed reaching: A comparison of methods," in *Proc. of the ARPA Image Understanding Workshop*, Moterey, CA, 1994.

[12] G. Cheng, A. Nagakubo, and Y. Kuniyoshi, "Continuous humanoid interaction: An integrated perspective— gaining adaptivity, redundance, flexibility—in one," *Robotics and Autonomous Systems*, vol. 37:pp. 161–183, 2001.

[13] R. B. Findler, J. Clements, C. Flanagan, M. Flatt, S. Krishnamurthi, P. Steckler, and M. Felleisen, "DrScheme: A programming environment for Scheme," *Journal of Functional Programming*, vol. 12(2):pp. 159–182, 2002.

[14] M. Kawato, K. Furawaka, and R. Suzuki, "A hierarchical neural network model for the control and learning of voluntary movements," *Biological Cybernetics*, 1987.

[15] A. Ude, C. G. Atkeson, and G. Cheng, "Combining peripheral and foveal humanoid vision to detect, pursue, recognize and act," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS2003*, Las Vegas, USA, 2003.