A Hybrid Neural Network Based Vision-Guided Robotic System

Kevin Stanley Jonathan Wu
Innovation Centre
National Research Council of Canada
3250 East Mall Vancouver, BC Canada
Kevin.Stanley@nrc.ca, Jonathan.Wu@nrc.ca

William A. Gruver
School of Engineering Science
Simon Fraser University
Burnaby, BC Canada
gruver@cs.sfu.ca

Abstract

There are two primary methods for mapping an input image to robot motion: computed kinematics and visual servoing. Computed kinematics uses a kinematic transform between the image plane and the world frame. Computed kinematics algorithms require only a single iteration, but are sensitive to calibration errors. Visual servoing uses a control law to regulate the image to a desired state. Visual servoing is more robust, but requires more computation to reach a solution. To balance these opposing factors, we proposed a hybrid system that uses an initial computed kinematics move followed by a visual servoing correction, thereby providing a compromise between speed and accuracy. A linear approximation model and a neural network were used to approximate the kinematic transform between the image and world frames. A PD control system is used to regulate the image to its final state.

1 Introduction

Many aspects of vision-guided robotics have been thoroughly researched, but few vision-guided robotic systems have been utilized in industry. Most vision-guided robotic systems were too slow, and too sensitive to the environment to be useful in an industrial setting. With the rapid rise in computing power and the drop in price of high quality robotic and vision systems, the application of vision guided robotic systems in industry is feasible. Advances in visual servoing, kinematics, computer vision, and neural networks provide the algorithmic basis to move robotic vision into industry.

An overview of visual servoing was given by Hutchinson, *et al.* [1] who described the research and fundamentals of geometric feature-based methods. Corke, [2] has shown that the performance of visual servoing algorithms can be enhanced by incorporating the dynamics of the system in the model. Panapikopolous, *et al.* [3] have used adaptive control techniques to servo a robot.

To automate the derivation of the Jacobian, many researchers have used neural network approximators in vision-guided robotics. Miller, *et al.* [4] used a neural network to demonstrate the feasibility of the approach. Hashimoto, *et al.* [5] used a two level self-organizing network to approximate the Jacobian for coarse and fine motion. Wu and Stanley [6][7] used a fuzzy decision network to manage a hierarchy of backpropagation networks to approximate the Jacobian over the entire workspace. Van der Smagt, *et al.* [8] theoretically demonstrated that a neural network may be used to position a vision-guided robot.

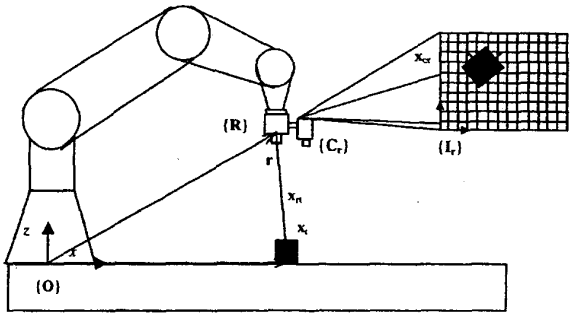
Calibration of the kinematic transforms between the image and the world coordinate system has been examined in detail. All research on visual servoing has been based on determining the coefficients of the transform using measurements captured from the robot camera pair. Wang [9] described the relationships between the different frames and applied three different methods to approximate the transform, ranging from the case of known target and position to unknown target and position. Horaud, *et al.* [10] described the effects of a perspective model on the accuracy of approximation. Wei, *et al.* [11] outlined an approach for computing the transform based on active vision principles. Zhuang, *et al.* [12] described a system where both the robot and camera were calibrated simultaneously. Remy, *et al.* [13] simplified the estimation by employing Euler representations in the transform.

We use a hybrid system, employing a coarse computed kinematics move followed by one or more visual servoing corrections, to balance the speed and accuracy of the system. For discrete part manipulation, we assume the workspace and targets are locally planar.

2 Kinematic Approaches

Figure 1 shows the experimental configuration of the system, and defines the frames and variables. The purpose of the system is to grasp a target object at an unknown position $\mathbf{x}_t$ by positioning the end-effector of the robot to a position $\mathbf{r}$ with respect to the target. The position of the robot $\mathbf{r}$ is known from the joint encoder positions and the inverse kinematics. The target position, $\mathbf{x}_t$, must be estimated from the images. The x, y, and θ components of the target position $\mathbf{x}_t$ are

estimated by measuring the position of the target in $\mathbf{x}_{cr}$ in the end-effector mounted camera frame $\{\mathbf{I}_r\}$. Because targets may have varying heights, we must also



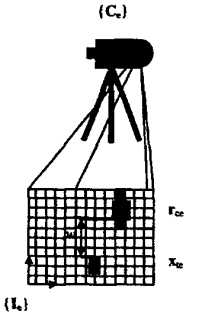


Figure 1: Camera and Robot Configuration for Test Cell

estimate the distance along z in the world coordinate frame $\{\mathbf{O}\}$. This measurement is made by examining the image frame of the external camera $\{\mathbf{I}_e\}$. The difference in y in $\{\mathbf{I}_e\}$ between the observed robot position $\mathbf{r}_{ce}$ and the measured target position $\mathbf{x}_{te}$ generates w, the observed difference in world z. Using notation from Hutchinson [1], let $\mathbf{x}_{cr}$ be represented by the 3-vector

$$\mathbf{x}_{cr} = (u\ v\ \theta)^T \qquad (1)$$

Combining $\mathbf{x}_{cr}$ with w as measured in $\{\mathbf{I}_e\}$ we generate the image space position of the target, denoted $\mathbf{x}_{ti}$, as

$$\mathbf{x}_{ti} = (u\ v\ w\ \theta)^T \qquad (2)$$

Consequently, the problem now may be described as using $\mathbf{x}_{ti}$ to position the robot with respect to the target, such that grasp planning and grasping occur. This motion can be generated using either a computed kinematics (also called "look-and-move") or a visual servoing. In computed kinematics, the position of the target in the world frame $\mathbf{x}_t$ is estimated using a transform $\mathbf{T}$ such that

$$\mathbf{x}_t = \mathbf{T}\mathbf{x}_{ti} \qquad (3)$$

Determining the transform T is the key focus of references [9]-[13]. Because look and move techniques are so sensitive to the calibration of $\mathbf{T}$, we also introduce visual servoing, which regulates the image position of the target $\mathbf{x}_{ti}$, to a desired position $\mathbf{x}_{tid}$ such that the 3-D position of the target relative to the robot in the base frame $\mathbf{x}_{rt}$ is known:

$$\mathbf{x}_{rt} = \mathbf{r} - \mathbf{x}_t \qquad (4)$$

Thus, if

$$\mathbf{x}_{ti} = \mathbf{x}_{tid} \qquad (5)$$

then (4) is satisfied.

2.1 Linear Calibration

For a distant viewpoint, the relationship between the translations of the camera in x and y and the translation in the image is approximately linear for planar motion. Correcting for the rotation θ, we arrived at the x and y calibration values. The value for θ was determined by measuring the drift in one variable while holding the other constant. The angle θ was determined as the arctangent between the drift and the magnitude of the move. This calculation assumes that the errors in pitch and yaw are small when compared to θ.

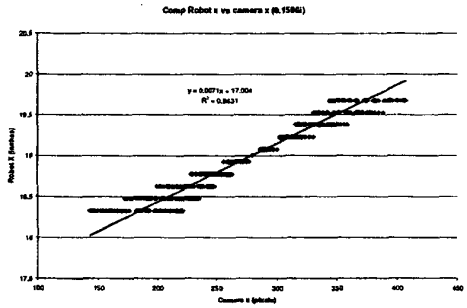


Figure 2: Compensated x Calibration Data with z Motion

Figure 2 shows that linear calibration is not accurate enough to generate a reliable position with respect to the target when motion in z is considered. The only method for eliminating this inaccuracy within a look and move framework is to perform more accurate calibration. Because all 6 variables of the calibration matrix are nonlinearly coupled, the solution involves a non-linear programming problem [9].

Most computed kinematic systems rely on some form of non-linear optimization to approximate the kinematic mapping. We employ a neural network to

approximate T locally and use a visual servoing component to provide a correction for local errors.

2.2 Neural Network Calibration

Neural networks are a class of non-linear approximators loosely based on the function of the human brain. They use multiple non-linear elements to approximate a global function. There are many networks and learning or optimization algorithms, but the most commonly used approach is the backpropagation network. We use a neural network in place of traditional non-linear optimization techniques [9] and [11].

Neural networks consist of nodes and weighted connections organized into layers. For a backpropagation network the input is carried through the network by a series of multiplications over the connections and summations at the nodes. The connection weights are held between -1 and 1 and the activation function is usually sigmoidal. The multi-layer perceptron node sums the outputs of all connected nodes and also a local bias that it feeds into the activation function and then passes to the next layer. Therefore, the output can be written as

$$\zeta_i = \sum_i \Theta(x w_{ij} + bias) \qquad (6)$$

$$y_k = \sum_j \zeta_j w_{jk} \qquad (7)$$

where

$$\Theta(\pi) = \frac{1}{1+e^{-\pi}} \qquad (8)$$

A sigmoidal non-linearity was chosen because it approximates the non-linearity in human neurons. In addition, the use of a sigmoidal non-linearity in (6) gives a three-layer perceptron the capability for arbitrarily accurate non-linear estimation [14].

Backpropagation propagates the error at the output nodes through the network, performing a gradient descent over the error surface. Because gradient descent provides local optimization, we must ensure that the network converges rapidly, and does not become stuck at local minima. Momentum was used to aid the gradient descent algorithm in overcoming small local minima. The result is the following equation relating the change in weight to the error.

$$\Delta w_{ij} = \alpha w_{ij}(n-1) + \eta \delta_i \Theta'_{(net)} x_j \qquad (9)$$

where Θ' is the derivative of the activation function, α is the momentum coefficient, and δ_j is determined as

$$\delta_j = \begin{cases} y_d - y_{net} & \text{if output layer} \\ \sum_j w_{ij}\delta_j & \text{otherwise} \end{cases} \qquad (10)$$

By iteratively calculating the delta term (node error), the error at the outputs of the network is carried backwards through the network.

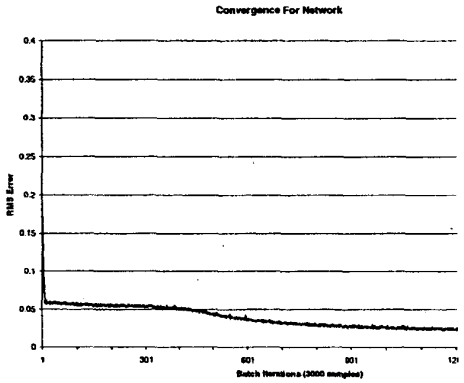


Figure 3: Network Convergence Profile

Figure 3 is the convergence profile for this neural network showing that it quickly converged to an error of 0.05 , then required a large number of iterations to converge to its final error of 0.02, approximately 0.01 inch along each axis.

3 Visual Servoing

Our visual servoing system is an image based, PD controller operating with the robot joint controllers in the loop. The control system is based on a PD controller structure shown in Figure 4,

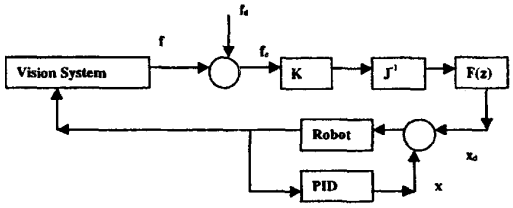


Figure 4: PD Control System

where x_d is the desired position x is the measured position, f is the current feature vector, f_d is the desired feature vector, f_e is the current feature error, **K** is the gain matrix, $\mathbf{J}^{-1}$ is the inverse feature Jacobian, and $F(z)$ is the discretization function. The plant model in this case is the inverse feature Jacobian. It is a mapping from the input feature space to the output Cartesian space. The Jacobian maps image errors to Cartesian velocities. The Jacobian is derived from the geometry of the system. Our derivation of the image

Jacobian is closely related to that described by Hutchinson [1].

We define the motion of the camera as

$$\dot{\mathbf{C}} = \Omega \times \mathbf{P} + \mathbf{T} \qquad (11)$$

where **C** is the speed of the camera, Ω is the rotational velocity of the end-effector, **P** is the position of the camera with respect to the robot end-effector and **T** is the translation velocity of the end-effector. Since servoing occurs only on 4 degrees of freedom, the angular velocities ω_x and ω_y are zero.

The inverse feature Jacobian is

$$\mathbf{J}^{-1} = \begin{bmatrix} \frac{w}{c_x c_z} & 0 & \frac{u}{c_x c_z} & \frac{wv}{c_x c_z} \\ 0 & \frac{w}{c_y c_z} & \frac{v}{c_y c_z} & -\frac{wu}{c_y c_z} \\ 0 & 0 & \frac{1}{c_z} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (12)$$

The measurement of z in the inverse Jacobian is stated in terms of the image measurement of z (w) because the distance from the manipulator to the top of the target is not known.

3.1 Gain Scheduling

Control strategies generally must balance the precision of convergence with the number of iterations required to converge. While higher gains tend to converge faster, they can lead to oscillation and even limit cycling. Because iterations are computationally expensive, we want to reduce the number needed to obtain a solution. To achieve a rapid response, and a stable solution, we have employed gain scheduling for the PD controller.

The gain scheduling system uses two gain levels, a high gain followed by a low gain positioning to achieve a fast and accurate final state.

The low gain portion removed limit cycling, which primarily occurred in x and y, and occasionally caused a coupled effect on θ. By reducing by 50% the proportional and derivative gains of the controller for the x and y coordinates, we eliminated limit cycling. The high gain is executed after the initial computed kinematics move and it continues until the target is within a 15-pixel square of the center of the image. The high gain matrix is then swapped with the low gain matrix.

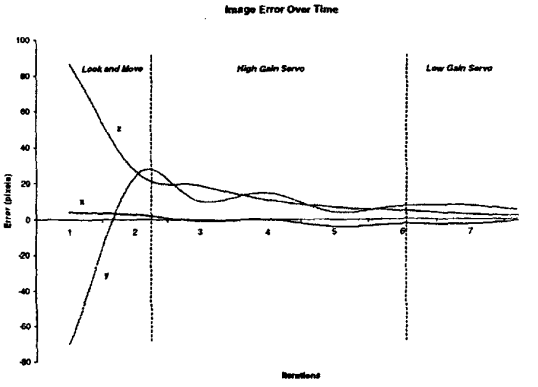


Figure 5: Sample Multi-Phase Response for the Hybrid System

Figure 5 indicates a marked difference in system response for the three stages of motion. The look and move portion is a linear point-to-point motion, whereas the high gain servo is characterized by a rapid response and oscillatory underdamped behavior. The low gain servo is more typically overdamped and moves the robot smoothly to its final position. The three stages of motion are apparent from studying the response in the y coordinate.

4 Results

A test system was composed of a dual 200 MHz Pentium Pro computer with a PCI bus. It includes a Matrox Genesis frame-grabber and digital image processing board connected via a serial interface to the CRS C-500 robot controller running the RAPL-3 operating system. All image-processing operations were performed on the Genesis frame-grabber. The Genesis board allows multiple threads of processes to be queued. By performing all image processing on the Genesis board we eliminated the need for transmitting images across the system bus. The serial connection to the robot was maintained at a relatively slow 9600 BPS rate because the serial hardware on the controller side is unreliable at high speeds and may cause errors. The robot was driven over the serial port using a master-slave system.

In this experiment, the robot was presented with five targets as shown in Figure 6. The first object, a conical paper cup, appearing as a circle in the top projection, was run eight times for each type of robotic positioning system as a baseline for observing the other targets. Each of the other 4 objects was targeted once using each algorithm. All robot-positioning algorithms were employed for each given target and initial position. Initially, targets were

Table 1: Target Generalization Comparison

	Visual Servoing		Look and Move Hybrid		Neural Network Hybrid	
	Iterations	Error	Iterations	Error	Iterations	Error
backing	11	0.79	7	2.48	4	2.41
crushed	12	2.93	12	2.67	6	4.07
elbow	9	3.36	5	3.06	11	3.04
plug	10	2.95	5	3.79	6	3.80
cup	10	3	7	2.73	7	2.05
Average	**9.92**	**2.82**	**6.75**	**2.37**	**6.92**	**2.46**

isolated in three degrees of freedom (x, y, and z). Then, they were positioned in θ after the grasp point was calculated. Because the mapping between the measured θ and the robot θ is 1:1 with the camera placed directly over the target, there is no coupling between motion in θ and motion in x and y.

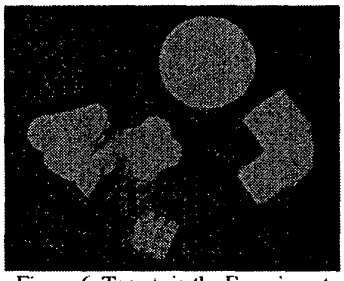


Figure 6: Targets in the Experiment

There are three distinct phases of motion for the computed kinematics systems, and only two phases of motion for the visual servoing system. The following diagrams compare the response of each system for the same target, and starting position.

Table 1 indicates that there is little difference in the behavior of the system to different targets.

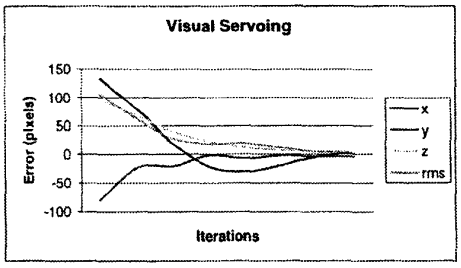


Figure 7: Visual Servoing Response

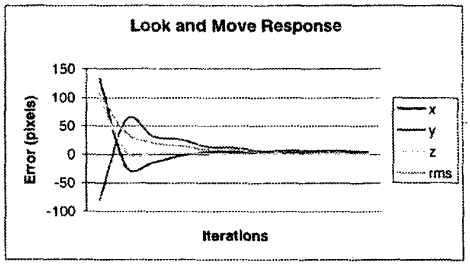


Figure 8: Linear Hybrid System Response

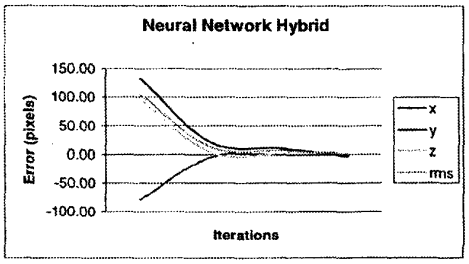


Figure 9: Neural Hybrid System Response

5 Conclusions

We have demonstrated a four-degree of freedom, generic, vision-guided robot. The system was designed to solve a general class of problems, targeting well-contrasted, visible, rigid, stationary objects. By ensuring that only assumptions about the size and color of the target were necessary for proper operation, we were able to create a very useful tool for implementing industrial vision-guided robotic systems. We chose our input features to be target independent for rigid objects. We used an orthogonal camera configuration to limit changes in target image projections. By ensuring that the input always had the same meaning, and satisfied well-established boundaries, it was possible to directly apply the same positioning algorithm to any target.

The system manipulates a wide range of discrete parts in four degrees of freedom without models of the targets. We have demonstrated that an efficient approach to vision guided robotics with limited bandwidth can be based on the use of a hybrid computed kinematics and visual servoing system. Limited bandwidth applications will be the only applications in industry for several years because robotic vendors are unwilling to provide complete access to the joint levels of their controllers, and low cost vision systems will not have the required ability to operate at speeds on the order of 1 kHz. The technology and cost now allow vision-guided robotics in practical applications. The proposed system has commercial potential for many applications including teleoperation and sorting.

6 References

1. Seth Hutchinson, Gregory Hager, and Peter I. Corke, "A Tutorial on Visual Servo Control," *IEEE Trans. on Robotics and Automation*, Oct 1996, pp. 651-669.
2. Peter I. Corke and Malcolm C. Good, "Dynamic Effects in Visual Closed Loop Systems," *IEEE Trans. on Robotics and Automation*, Oct 1996, pp. 671-683.
3. Nikolaos P. Papanikolopoulos, Pradeep K. Khosla, "Adaptive Robotic Visual Tracking: Theory and Experiments," *IEEE Trans on Automatic Control*, March 1993, pp. 429-445.
4. T.W. Miller III, "Neural Networks for Sensor Based Control of Robots with Vision," *IEEE Transactions on Systems Man and Cybernetics, Vol. 19, No.4*, 1989, pp. 826-831.
5. H. Hashimoto, T. Kubota, M. Kudon, and F. Harashimo, "Self-Organizing Visual Servo System Based on Neural Networks," *American Control Conference, Boston MA, 1991*, pp. 31-36.
6. J. Wu and K. Stanley, "Modular Neural-Visual Servoing using a Neural-Fuzzy Decision Network," *IEEE Conference on Robotics and Automation, Albuquerque, 1997*, pp. 3238-3243.
7. Q.M.J. Wu, C.W. de Silva and Kevin Stanley "Neural Control Systems and Applications," *Intelligent Adaptive Control: Industrial Applications*, CRC Press, 1998.
8. P. van Der Smagt, F. Groen, "Approximation with Neural Networks: Between Local and Global Approximation," *IEEE Int'l Conf. on Neural Networks, 1995*, pp. 1060-1064.
9. Ching-Cheng Wang, "Extrinsic Calibration of a Vision Sensor Mounted on a Robot," *IEEE Trans. on Robotics and Automation*, Vol. 8, No. 2, April 1992, pp. 161-175.
10. Radu Horaud, Fadi Dornaika, Bart Lamiroy, and Stephane Christy, "Object Pose: The Link between Weak Perspective Paraperspective and Full Perspective," *International Journal of Computer Vision* 22 (2), 1997, pp. 173-189.
11. Guo-Qing Wei, Klaus Arbter, and Gerd Hirzinger, "Active Self-Calibration of Robotic Eyes and Hand-Eye Relationships with Model Identification," *IEEE Trans. on Robotics and Automation*, Vol. 14, No. 1, February 1998, pp. 158-166.
12. Hanqi Zhuang, Kuanchih Wang, Zvi S. Roth, "Simultaneous Calibration of a Robot and a Hand-Mounted Camera," *IEEE Trans. on Robotics and Automation*, Vol. 11, No. 5, 1995, pp. 649-660.
13. S. Remy, M. Dhome, J. M. Lavest, N. Daucher, "Hand-Eye Calibration," *Proceedings of the 1997 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1997 pp. 1057-1065.
14. Simon Haykin, *Neural Networks*, Macmillan College Publishing, 1994.