Proceedings of the 1999 IEEE
International Conference on Robotics & Automation
Detroit, Michigan • May 1999

Neural Network-Based Vision Guided Robotics

Kevin Stanley, Q.M. Jonathan Wu, Ali Jerbi
National Research Council of Canada-
Integrated Manufacturing Technology Institute

William A. Gruver
Simon Fraser University
School of Engineering Science

ABSTRACT

An essential problem of image-based visual servoing is evaluating the inverse Jacobian which, relates changes in image features to the change in robot position. Neural networks can learn to approximate the inverse feature Jacobian. In addition, neural networks have been used in dimensionality reduction of image input. In this paper, we show that it is possible to use neural networks for both feature extraction using compression and for feature Jacobian approximation in the visual servoing problem. In our system, we consider the following feature extraction methods: geometric features, averaging compression, vector quantization, and principal component extraction.

I. INTRODUCTION

In recent years, there has been a great interest in the area of vision-guided robotics as a major research topic crossing many disciplines. Applications such as manufacturing processes, autonomous robotics and space technology can benefit significantly from using vision-guided robotics. Visual servoing is a method, which uses the inverse feature Jacobian to map input geometric features to an output robot velocity. This approach requires a human to derive the Jacobian and select the features. Feature selection and inverse Jacobian derivation can be tedious, and are unique to every robot-target pair. For every new manipulator and target a new set of features must be chosen and the Jacobian derived and inverted.

An excellent overview of visual servoing is given by Hutchison *et al.* [1] who describes the research and fundamentals of geometric feature based visual servoing. 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 perform visual servoing.

To automate the derivation of the Jacobian, many researchers have used neural networks to approximate the Jacobian. 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 course and fine motion. Wu and Stanley [6] used a fuzzy decision network to manage a hierarchy of backpropagation networks to approximate the Jacobian over the entire workspace. Van Der Smagt [7] theoretically demonstrated that a neural network can achieve positioning of a vision-guided robot. All of the above researchers used geometric features for their input. A new target object required that a new feature set be chosen.

Other researchers have attempted to automate feature selection. Feddema [8] has shown that geometric feature selection can be aided by measuring the properties of the Jacobian. Hashimoto and Noritsugu [9] have derived a similar measure for feature sensitivity. On the other hand, some researchers have used appearance or pixel-based methods to create a feature vector. Instead of automating the feature extraction, Nayar [10] used principal component extraction (PCX) algorithm to find the eigenvectors of the image and servoed the robot based on these eigenvectors. Anguita *et al.* [11] used an averaging compression to create a feature vector which was used to train a neural network which approximated the inverse Jacobian.

In this paper, we use neural networks for both feature extraction and inverse Jacobian approximation. We implement a completely neural network based system that uses a backpropagation network to approximate the image Jacobian, and other networks to extract the feature vector. We use two forms of neural network based image compression to extract the features to be fed to the inverse Jacobian. A Hebbian network is used to perform PCX eigenvector extraction and eigenspace compression. A Self-Organizing Feature Map (SOFM) is used to perform compression using the Vector Quantization (VQ) technique. We experimentally demonstrate that it is possible to position accurately a robot using only neural

networks for both image dimensionality reduction and inverse Jacobian approximation. We compare the behavior of our system with the geometric feature based and averaging compression approaches to show that the appearance-based methods (PCX and VQ) are comparable to the more traditional techniques. We show that it is feasible but not entirely practical to use neural networks for all aspects of vision guided robotics.

The paper is organized as follows: Section II describes the each feature extraction algorithm, Section III presents our experimental setup, Section IV contains the experimental results and Section V contains our conclusions.

II. FEATURE EXTRACTION ALGORITHMS

A neural network trained directly from the image would not converge in a reasonable amount of time [12]. Therefore, an intermediate step is required which maps the input image into a smaller dimensional space: mapping the input image **M** onto a smaller vector v of dimension m, via a transform **T**.

$$\mathbf{T}(\mathbf{M}) = \mathbf{v} \quad \{\mathbf{M} \in R^i, \mathbf{v} \in R^m; i > m\} \quad (1)$$

The transform **T** can be implemented in several different ways. Geometric-based feature extraction, averaging compression and neural compression can be used to reduce the input dimensionality.

A. Geometric Feature Extraction

This is the traditional method of dimensionality reduction for neural visual servoing [1][2][3][4][5][6][7][12]. Features provide a compact representation of images. The 640 by 480 images in our experiments were reduced to a 6-vector when the appropriate features were selected resulting in a compression ratio of 50000 to 1.

B. Averaging Compression

The averaging compression technique in our system was inspired by the system impof Anguita *et al.* [11]. The technique divides the picture into windows, then the grayscale values of all pixels in each window are averaged to create a feature vector. The compression ratio of this technique is the total number of pixels in the image divided by the number of windows. In our case, the images were 640 by 480 and we used 192 windows, so the compression ratio was 1600 : 1.

C. SOFM Compression

Kohonen's self-organizing feature maps is one of the most common self-organizing paradigms. A self-organizing feature map is a two layer network with a two dimensional input and a single dimensional output, as shown in Figure 1.

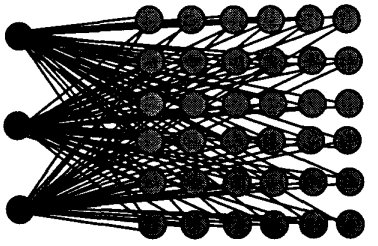

Fig. 1: SOFM Network

Every node in the output layer (black) is completely connected to every node in the input layer (gray). The output of the network is determined on a winner-take-all (WTA) basis. The WTA evaluation criterion can be expressed as

$$y_i = \begin{cases} 1 & \text{if } \min_i \|\mathbf{x} - \mathbf{w}_i\| \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

where x is the input vector and w_i is the weight vector for the ith output node.

The SOFM algorithm compresses the image by dividing the image into windows. However, instead of averaging the pixels in each window like the previous algorithm, the SOFM attempts to find the n most common patterns in the image using a technique similar to VQ. After training, the SOFM should have n patterns stored as weights in the network. In the forward iteration, these weights are compared to the current contents of the image window using Eq.. (2) [12].

In our implementation the size of each window is 40 by 40, so the message size is the same as the averaging compression network, resulting in a compression ratio of 1600 : 1. Using this configuration, the SOFM network must have 1600 nodes in the input layer. In order to represent enough common features, we used nine outputs or VQ codes. The SOFM network therefore has 14,400 connections. The control network must have 192 inputs. There is a significant difference in execution speed between the SOFM and the geometric feature based representation. The number of computations required to compute the inverse Jacobian for a 192 input backpropagation network is much more complex than the computational effort required to compute the backpropagation

network for a feature based representation. This is due to the exponential computational increase in complexity for completely connected backpropagation networks.

D. Hebbian Compression

Hebb's learning rule is based on observations of the functioning of the brain [13]. Hebb observed that stimulation tended to strengthen connections and lack of stimulation tended to atrophy connections. Equations governing the operation of Hebbian networks are shown in Eqs. (3) and (4) [12].

$$y(n) = \sum_{i=0}^{p-1} w_{ji}(n)\, x_i(n) \tag{3}$$

$$\Delta w_{ji}(n) = \eta\left[y_j(n)\, x_i - y_j(n) \sum_{k=0}^{j} w_k(n)\, y_k(n) \right] \tag{4}$$

where p is the number of nodes in the Hebbian (output) layer, and j is the number of nodes in the input layer, $y(n)$ is the output of the nth node, wij is it weight vector and $x(n)$ is its input. Eq. 4 is the learning equation where Δwij is the change in weight, $y(n)xi$ is the weighted output of the current node and the final term is the weighted response of preceding terms and η is some small number.

Hebbian networks have a two-layer topology similar to the Kohonen network. Links from the input layer to the Hebbian layer are feed forward and fully connected. Hebbian networks can perform principal component extraction. In principal component extraction, the network learns to extract the p-1 most influential eigenvalues, that is, the eigenvalues that contain the most information about the image. The Hebbian network compresses the image, while still preserving the statistically most important information.

The Hebbian network outputs a set of values that correspond to recreating the image from the n eigenvectors obtained during training. These values can be used to recreate the image according to the following equation:

$$\mathbf{I} = \sum_{i=0}^{n} y_i \mathbf{W}_i \tag{5}$$

where $\mathbf{I}$ is the uncompressed image in the jth window of the image, y_i is the ith output of the Hebbian network and w_i is the weight vector associated with y_i. Because each output is stored as a weight vector, the message is n times longer than an averaging compression or SOFM compression. This increase in size corresponds to a compression/quality tradeoff. The Hebbian network gives noticeably better results than the SOFM network, but the compression for the same number of windows changes from 1600 : 1 to 1600 : n. The Hebbian network has difficulty extracting eigenvalues for eigenvectors (windows) larger than 40 by 40. Because of this constraint, the input of the control network must be n by 192, resulting in networks too large to be effectively implemented, even for $n = 4$. In order for the compression to work correctly, the image must be sampled before feeding it into the network. If 192 is the maximum input vector length, then the image must be subsampled to $\mathbf{M}/n$. If n is large, it may be desirable to use an n by n mean filter on the image to ensure that important pixels are not ignored.

III. EXPERIMENTAL SETUP

The experiment utilized of a 6 DOF CRS A465 robot moving an end effector mounted camera over a simple object. A backpropagation network (called the control network in this paper) was used to approximate the Jacobian for each type of vision input. The robot was servoed in 4 DOF using a single camera. Two objects were chosen, a wood chip and a C-ring. The chip and C-ring are shown in Figure 2 and Figure 3.

Figure 2: Wood Chip

Figure 3: C-Ring

Before the control networks could be trained the vision compression networks for both the VQ and PCX algorithms were trained. These networks trained on a single image offline.

The SOFM network for the VQ algorithm had a 40 by 40 input window, and nine outputs. The SOFM network required approximately 10 iterations over the image to converge. The SOFM network required no sampling or output weight scaling.

The PCX network had a 40 by 40 input window, and four outputs. The Hebbian network requires approximately 50 iterations across the image to converge. The Hebbian network required that data be sampled at a 4:1 ratio to

remain below the maximum number for inputs for the control network.

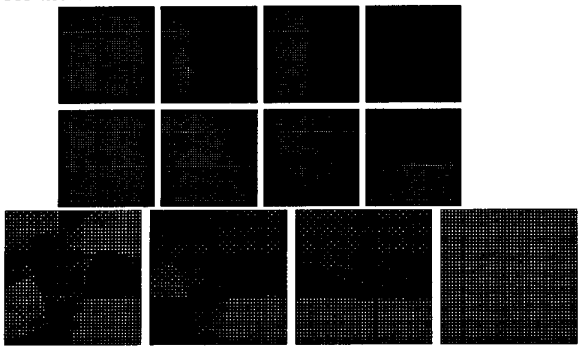

Figure 4: VQ Codes (top) and PCX Eigenvectors (bottom)

The 4 by 2 image on the top of Fig. 4 contains 8 of the codes generated by the SOFM VQ algorithm for the woodchip in Fig. 4. The 4 images on the bottom show the 4 most influential eigenvectors generated by the Hebbian PCX algorithm. It is apparent that the VQ algorithm codes parts of the object, while the PCX algorithm obtains features, such as edges and textures.

In the experiment, the geometric feature method and averaging compression algorithms serve a control group with which to compare the performance of the neural network compression algorithms.

IV. RESULTS

A. Convergence Test

The purpose of the convergence test was to evaluate how well and how quickly the control network could learn to approximate the feature Jacobian for each of the different kinds of input. The feature based control network had 16 input nodes (10 features and 6 DOF current position). The other control networks had 198 input nodes (192 member feature vector and 6 DOF current position). All control networks had 20 hidden nodes and a learning rate of 0.25. The results are shown in Figure 6, which shows the root mean square (RMS) error versus the number of iterations. The RMS error is computed in joint space in order to eliminate scaling requirements for comparing angular motions to cartesian space motions.

In order to reduce the total timeto convergance, a 100 point training file was recorded for each input type. The backpropagation network was trained from the data file.

The VQ based input exhibited the worst performance, as expected. The VQ algorithm performs poorly when the object is rotated. However, it still is sufficient to position the robot on the order of millimeters with respect to the target, which is sufficient for many pick and place applications.

The PCX algorithm approaches the performance of the geometric-based feature extraction method and performed better than the VQ algorithm because the eigenvector approach was more robust with respect to orientation. The PCX algorithm ended with an error of approximately 4%.

The averaging compression approach attained a very low error. The final error value was 2%. This difference can be explained in part by the size of the control network associated with the compression-input algorithm. The control network has 198 inputs and 20 hidden nodes. This means the network has 3960 connections to modify between the input and hidden layers. The geometric feature based network only had 320 connections to modify between the two layers. It is logical to surmise that the smaller network would converge faster.

As expected, the geometric-based method converged fastest. Control networks using other

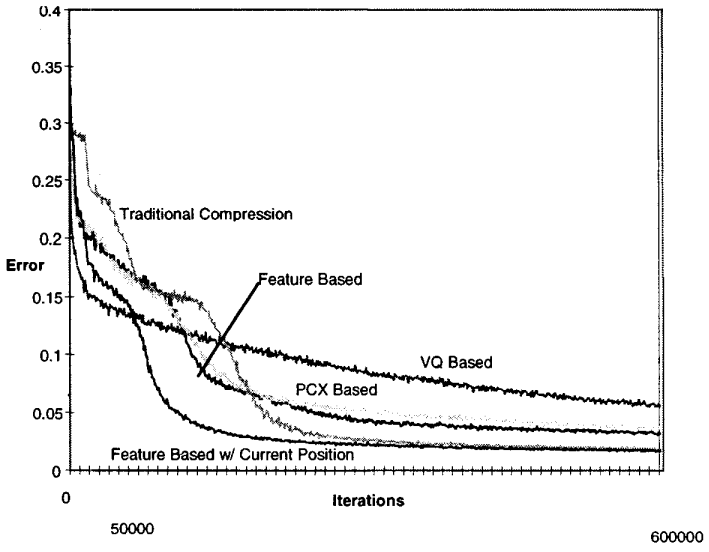

Figure 4: Convergence Profiles for Control Networks Under Different Inputs

image processing algorithms as input had to learn to ignore irrelevant feature vector elements such as background areas. In the geometric-based feature scenario, every item in the feature array is relevant, and contains independent information.

B. Generalization Test

The purpose of the generalization test was to measure the ability of a converged control network to approximate the Jacobian. The experiment tested the ability of the control networks to generalize by moving the robot to a random location within the workspace, then running the control network. The final desired position was the center of the workspace. Care was taken not to move the target object after training the network, so the center of the workspace remained the desired position. The network continued iterating until the feature error fell below a specified threshold, or the number of iterations exceeded a specified limit (20). The initial offset, final feature error, number of iterations, and final position error were recorded. One hundred iterations of the algorithm were tested.

The results of the experiment were expressed as the mean and standard deviation of the error. The objective was to achieve an error as close to the robot positioning error of 3 encoder pulses (a fraction of a degree) per joint as possible over a small area. The results for the best test are summarized below.

Table 1: Final Position Error for 100 Samples for Different Input Types

	Feature Based	Averaging Compression	PCX	VQ
Error Threshold	0.002	0.00035	0.005	0.1
Mean	9.78	9.18	16.78	52.69
Standard Deviation	4.14	2.84	7.42	26.40
Mean Iterations	2.21	1.93	2.3	1.7

The averaging compression system showed the best approximation properties. Its final error was slightly lower, and its standard deviation was smaller, resulting in a narrower error curve and a much better chance that the system would operate correctly. It also had the lowest number of iterations required to reach the desired position.

The PCX algorithm was not as successful. The mean error for the PCX was approximately twice the error of the other two algorithms. This performance gap was expected. While the PCX has better information conservation properties that the VQ algorithm, it still tends to lose rotation and scale information.

The VQ based input performed very poorly when compared to the other forms of input. This was primarily due to the loss of scale and rotation information at such high levels of compression. The resulting output did not capture the input well enough for the control to create a one to one mapping.

C. Compression Test

In order to compare the behavior of the geometric feature method and compression based methods, we tried to create a scenario that would be difficult for geometric-based feature extraction, but not compression based extraction.

We chose a C-ring to be the target. It is shaped much like a washer, but with a quarter of the circumference open. It is flanged to help it grip. The C-ring is shown in Figure 3. While the shape appears simple to the human eye, it highly concave shape can confuse simple segmentation algorithms.

The trial was conducted in the same manner as the previous trials. Other than the substitution of the C-ring for the wood chip, all the parameters for image, workspace, and neural network size were the same. After the failure of the SOFM-based visual compression in the previous experiment, it was not included in this experiment. During the course of the experiment, geometric feature based, averaging compression and PCX based networks were trained.

The first step of the generalization experiment was to characterize how well the geometric feature based neural control performed. However, this was not possible. The concave nature of the C-ring was outside the set of shapes the geometric feature based method could recognise. It could not even recognize that the object was present.

The averaging compression algorithm performed well, converging to the same average position error as obtained with the square woodchip. The PCX algorithm took longer to converge, and converged to a higher value, but could locate the position quickly. The results for the averaging compression and PCX-based networks are shown in Table 2. Geometric feature-based positioning failed to isolate the object, so no results were obtained.

Table 2: Generalization Test Results

	Averaging Compression	PCX
Error Threshold	0.0005	0.009
Mean	8.84	13.77
Standard Deviation	5.08	5.58
Mean Iterations	1.56	1.92

The results of the compression algorithm support the hypothesis that the compression algorithm would prove to be more robust for different kinds of objects. However, the PCX and averaging compression techniques only are useful for objects with a high degree of contrast. This constraint can be mitigated by using edge detection and thresholding techniques to increase the contrast before compressing the image. While the system is quite accurate, the size and complexity of the neural networks makes the solution quite computationally complex. In all cases on a Pentium processor cycle times were slower than 1 Hz. A parallel implementation of the neural networks would be required to make the system practical outside the laboratory.

V. CONCLUSION

We have developed a method for using neural networks to extract the feature information as well as to approximate the inverse feature Jacobian. We experimentally demonstrated that the behavior of our system is comparable to the behavior of traditional geometric and averaging feature extraction algorithms. While our method was computationally slower than the other algorithms, it was more flexible. The averaging compression algorithm had the best combination of performance and flexibility; however, we suspect that the averaging compression system would fair poorly on images that are more complex. Because the averaging compression system treats all windows the same, it may lose important detail information.

The control of the position of the robot was carried out by a backpropagation network approximating the image Jacobian. The feature based and averaging compression systems achieved the best convergence and generalization characteristics. The neural network performed to the same order of magnitude as the geometric feature based and averaging compression systems. The compression-based systems offer some advantages over the geometric system: they require no *a priori* knowledge of the target's appearance, and can be applied to any high contrast image.

The work described in this paper shows that the development of a completely neural network based visual servoing system is but not practical outside the labratory environment.

VI. REFERENCES

1. Hutchinson, Seth, Hager, Gregory, and Corke Peter I, "A Tutorial on Visual Servo Control", *IEEE Trans. on Robotics and Automation*, Oct 1996.
2. Corke, Peter I., Good, Malcolm C., "Dynamic Effects in Visual Closed Loop Systems", *IEEE Trans. on Robotics and Automation*, Oct 1996.
3. Papanikolopoulos, Nikolaos P., Khosla, Pradeep K., "Adaptive Robotic Visual Tracking: Theory and Experiments" *IEEE Trans on Automatic Control,* March 1993.
4. Miller, Thomas W III, "Neural Networks for Sensor Based Control of Robots with Vision", *IEEE Transactions on Systems Man and Cybernetics, Vol 19, No.4*, 1989.
5. Hashimoto, Hideki *et al.* "Self-Organizing Visual Servo System Based on Neural Networks", *American Control Conference, Boston MA, 1991*
6. Wu Jonathan and Stanley, Kevin, "Modular Neural-Visual Servoing using a Neural-Fuzzy Decision Network", *IEEE Conference on Robotics and Automation 1997.*
7. van Der Smagt, P. *et al.*, "A Monocular Robot Arm Can Be Neurally Positioned", *1995, International Conference on Intelligent Autonomous Systems.*
8. Feddema, John T., Lee, George C.S., Mitchell, Owen Robert, "Weighted Selection of Image Features for Resolved Rate Visual Feedback Control" *IEEE Transactions on Robotics and Automation, Vol 7, No 1*, 1991.
9. Hashimoto, Koichi and Noritsugu, Toshiro, "Performance and Sensitivity in Visual Servoing", *IEEE Conf. on Robotics and Automation, Leuven, Belgium*, May 1998.
10. Nayar, Shree K, Murase, Hiroshi, and Nene Sameer A., "A General Learning Algorithm for Robotic Vision" SPIE Vol. 2304, 1994.
11. Aguita *et al.* "Neural Structures for Visual Motion Tracking" *Machine Vision and Applications,* August 1995.
12. Haykin, Simon, *Neural Networks: A Comprehensive Foundation,* Macmillan College Publishing Company, New York, 1994.
13. Hebb, D.O. *The Organization of Behavior: A Neuropsychological Theory*, New York, Wiley, 1949.