Journal of Systems Engineering and Electronics, Vol.12, No.3, 2001, pp.45~50

Dynamic Coordination of Uncalibrated Hand/Eye Robotic System Based on Neural Network*

Su Jianbo, Pan Qielu & Xi Yugen
Institute of Automation, Shanghai Jiaotong University, 200030, P. R. China

(Received September 7, 2000)

Abstract: A nonlinear visual mapping model is presented to replace the image Jacobian relation for uncalibrated hand/eye coordination. A new visual tracking controller based on artificial neural network is designed. Simulation results show that this method can drive the static tracking error to zero quickly and keep good robustness and adaptability at the same time. In addition, the algorithm is very easy to be implemented with low computational complexity.

Keywords: Nonlinear visual mapping model, Robotic system, Dynamic coordination, Neural network.

1. INTRODUCTION

Over the past several years, vision servoing, especially hand-eye coordination, has been a major research topic in intelligent robotic area [1]. The traditional approaches solve the 3-D relationship between the robot and the environment based on 2-D vision measurements [2]. However, these measurements require accurate calibration of the internal and external parameters of camera, which could be tedious and error prone and in some situations, may be even infeasible to be obtained. Recently, uncalibrated visual servoing, which means no knowledge of hand/eye relationship is required before control, has attracted more and more attention. The basic idea is to employ the concept of image Jacobian matrix, which aims to relate the robot moverment to errors in image space directly. Yoshimi and Allen [3] proposed a scheme for performing a 2-D alignment task using rotational invariance. Hager [5] studied projective invariance for uncalibrated visual control. Based on their successful applications, Hespanha et al [4] discussed what can be done with an uncalibrated stereo system for overview.

The key problem of image Jacobian based method is to estimate the relation effectively on line. Sutanto [6] proposed the use of additional "exploratory motion" to improve the estimation of the image Jacobian. Some researches have used a neural network to approximate the image Jacobian. Miller [9] used a neural network to demonstrate the feasibility of the approach. Hashimoto et al [10] used the method to position and orientate the end-effector. However, deep investigation shows that the image Jacobian is very difficult to be obtained in the case of that both of the camera and the object are moving [8], such as an eye-in-hand system to tack a moving object.

In this paper, we present a non-linear visual-mapping model to replace the image Jacobian, and explore the uncalibrated dynamic coordination of eye-in-hand system. The new visual mapping model holds in the whole robotic workspace and need not to be iterated on-line. Computer simulations verify the performance of the new control scheme.

2. PROBLEM STATEMENTS

Fig.1 shows the visual control system discussed in the paper. For avoiding the occlusion of the image features, the camera is mounted on the end-effector of the manipulator arbitrarily (eye-in-hand configuration). Consider an object moving in 3-D space in pure translational way, then the problem of 3-D visual tracking can be defined as "find the camera translation $[T_{hX}, T_{hY}, T_{hZ}]$ with respect to the base frame that keeps the projection of the moving object always at the desired location in the image".

*This project was supported by the National Natural Science Foundation (No. 69875010).

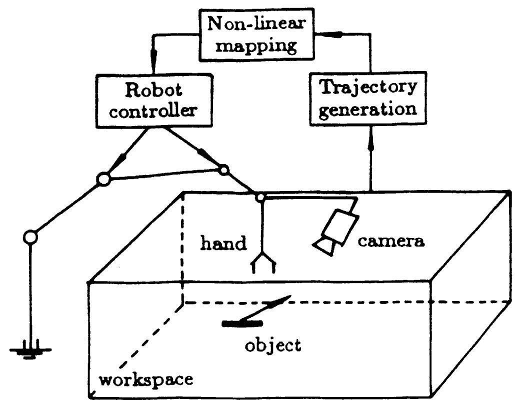

Fig.1 System configuration

Note that our work differs significantly from previous approaches in the sense that it requires no calibration about the camera parameters, including the relative pose of the camera with respect to the end-effector and the perspective model of the camera. In addition, there is not any type of assumption about the pose relationship among the camera, end-effector and object. We propose algorithms that address real-time robotic visual tracing of moving object and can be used in the whole robotic workspace successfully.

3. Visual Mapping Model

In order to solve the 3-D relationship between the robot and environment, we define the image feature space $\Omega = \text{span}\{x, y, z\}$, where x, y are the coordinates of the projection of feature point p on the image plane while z is the characteristic size of the image. Here z is just the length of image since the object looks like a stick. At time kT, assume that the instantaneous velocity of the projection point in 3-D image feature space is $V(kT) = [u(kT), v(kT), w(kT)]^T$ and the instantaneous acceleration of it is $[\Delta u(kT), \Delta v(kT), \Delta w(kT)]^T$. To keep the notation simple, we use k instead of kT. It is obvious that the image feature acceleration. $[\Delta u(k), \Delta v(k), \Delta w(k)]^T$, is determined by the relative motion of the object with respect to the camera. In the base coordinate frame, the relative motion of the moment kT can always be described by the following three terms: 1) the instantaneous acceleration of object, $a_0(k)$; 2) the relative velocity between the camera and object. $V_{co}(k)$; 3) the acceleration of the camera, $a_c(k)$. Therefore, the instantaneous acceleration $[\Delta u, \Delta v, \Delta w]^T$ can also be decomposed into three corresponding terms as

$$\begin{bmatrix} \Delta u(k) \\ \Delta v(k) \\ \Delta w(k) \end{bmatrix} = \begin{bmatrix} \Delta u_0(k) \\ \Delta v_0(k) \\ \Delta w_0(k) \end{bmatrix} + \begin{bmatrix} \Delta u_{co}(k) \\ \Delta v_{co}(k) \\ \Delta w_{co}(k) \end{bmatrix} + \begin{bmatrix} \Delta u_c(k) \\ \Delta v_c(k) \\ \Delta w_c(k) \end{bmatrix} \tag{1}$$

where $[\Delta u_0, \Delta v_0, \Delta w_0]^T$ is the component of acceleration induced by $a_0(k)$. Assuming $a_0(k) \equiv 0$, then this term can be omitted (equals to zero). Notice that this assumption will be removed in Section 4. $[\Delta u_{c0}, \Delta v_{c0}, \Delta w_{c0}]^T$ is the component induced by the relative velocity $V_{co}(k)$. Even though $V_{co}(k)$ keeps the same along the time. the projection point in the 3-D feature space moves with a varying velocity, which leads to $[\Delta u_{co}, \Delta v_{co}, \Delta w_{co}]^T$. Additionally, because $V_{co}(k)$ is completely determined by the coordinate $[x(k), y(k), z(k)]^T$ and the instantaneous velocity $[u(k), v(k), w(k)]^T$, $[\Delta u_{co}, \Delta v_{co}, \Delta w_{co}]^T$ can be written as

$$\begin{bmatrix} \Delta u_{co}(k) \\ \Delta v_{co}(k) \\ \Delta w_{co}(k) \end{bmatrix} = f_{co}\left(\begin{bmatrix} x(k) \\ y(k) \\ z(k) \end{bmatrix}, \begin{bmatrix} u(k) \\ v(k) \\ w(k) \end{bmatrix} \right) \tag{2}$$

In equation (2), $[\Delta u_c, \Delta v_c, \Delta w_c]^T$ is the component induced by $a_c(k)$. Because the manipulator moves with translating motion, $a_c(k)$ is just the same as the acceleration of the end-effector $[\Delta T_{hX}, \Delta T_{hY}, \Delta T_{hZ}]^T$. Then, $[\Delta u_c, \Delta v_c, \Delta w_c]$ can be expressed as

$$\begin{bmatrix} \Delta u_c(k) \\ \Delta v_c(k) \\ \Delta w_c(k) \end{bmatrix} = f_c\left(\begin{bmatrix} x(k) \\ y(k) \\ z(k) \end{bmatrix}, \begin{bmatrix} \Delta T_{hX}(k) \\ \Delta T_{hY}(k) \\ \Delta T_{hZ}(k) \end{bmatrix} \right) \tag{3}$$

Moreover, both $f_{co}(*, *)$ and $f_c(*, *)$ are non-linear functions because the mapping between the robotic workspace and 3-D image feature space is non-linear. Substitute equations (2) and (3) into equation (1), we derive the visual-mapping model as

$$\begin{bmatrix} \Delta u(k) \\ \Delta v(k) \\ \Delta w(k) \end{bmatrix} = f'\left(\begin{bmatrix} x(k) \\ y(k) \\ z(k) \end{bmatrix}, \begin{bmatrix} u(k) \\ v(k) \\ w(k) \end{bmatrix}, \begin{bmatrix} \Delta T_{hX}(k) \\ \Delta T_{hY}(k) \\ \Delta T_{hZ}(k) \end{bmatrix} \right) \tag{4}$$

Invert the input-output relationship of equation (4), we can obtain the inverse visual mapping model (from the image feature space to the robotic workspace), such as

$$\begin{bmatrix} \Delta T_{hX}(k) \\ \Delta T_{hY}(k) \\ \Delta T_{hZ}(k) \end{bmatrix} = f\left(\begin{bmatrix} x(k) \\ y(k) \\ z(k) \end{bmatrix}, \begin{bmatrix} u(k) \\ v(k) \\ w(k) \end{bmatrix}, \begin{bmatrix} \Delta u(k) \\ \Delta v(k) \\ \Delta w(k) \end{bmatrix} \right) \tag{5}$$

4. Control Scheme

In order to track the moving object rapidly, we should integrate sensing (in this case, a camera) with control tightly. Since the pose of camera relative to the end-effector is unknown, there are lots of unknown parameters, which are difficult to be estimated on-line by ARMAX model. In our work, we construct an artificial neural network (ANN) according to equation (5), which maps the motion instruction from 3-D image feature space to the robotic workspace. This method avoids the estimation of unknown parameters. Fig.2 shows the control scheme of the visual tracking system.

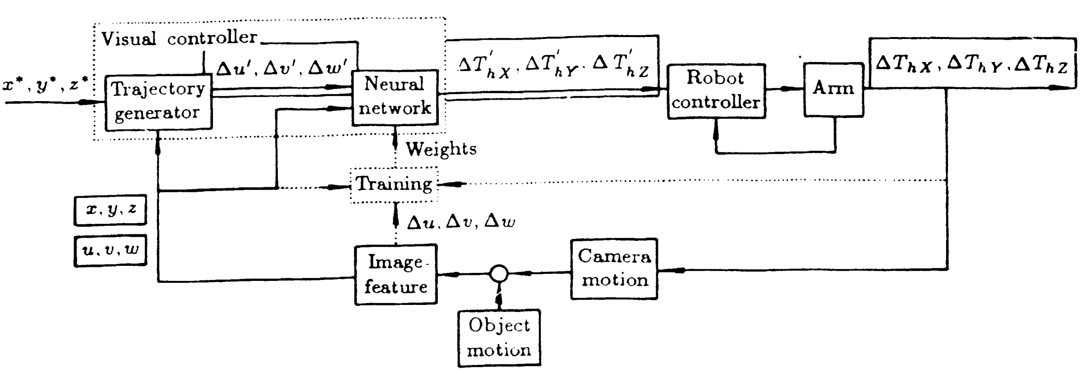

Fig.2 Control scheme of the visual tracking system

4.1 Real-Time Trajectory Generation

Since the distance between the actual and desired features $d(k)$ is often too large to be compensated in a single robot control cycle, the trajectory generator is needed to break up the emotion into realizable segments. The methods of trajectory generation in joint and Cartesian spaces are well known. However, these methods are infeasible in this case because the position of the moving object is difficult to be estimated in robot space. To avoid this problem, the trajectory generator should be performed in the image feature space. Notice that the camera is tracking a moving object, whose position at the sensor sampling time does not correspond to that predicted by the trajectory generator. Thus, we have to plan the velocity instead of trajectory during each vision cycle [11]. In addition, the acceleration of the moving object is usually not equal to zero, so it is necessary to overcome the tracking error caused by the assumption in Section 3. To sum up, the trajectory generator in the paper is a feature-based velocity generator, which can update the velocity in real-time and remove the tracking error caused by all of the above uncertainties.

Fig.3 shows the velocity generation scheme. For tracking the object rapidly, the planned velocity $V^{'}(k) = [u^{'}(k), v^{'}(k), w^{'}(k)]^T$ should always be pointed at the desired feature $[x^*, y^*, z^*]^T$ at any time. Its amplitude $\mid V^{'}(k+1) \mid$ should be regulated according to the following rules (note that we will use $d(k)$ to represent the distance between the actual and desired features in 3-D feature space henceforth).

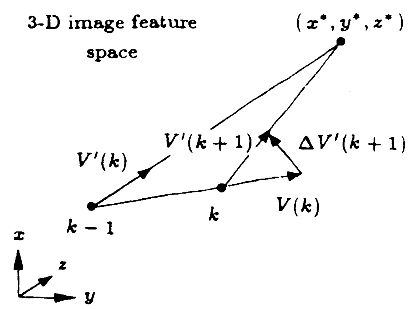

Fig.3 The velocity generation scheme

1) When $d(k) \geq d_1$, increase $\mid V^{'}(k+1) \mid$ until the velocity of the end-effector (in robot space) reaches and keeps at its maximum allowed;

2) When $d_1 > d(k) \geq d_2$, decrease $\mid V^{'}(k+1) \mid$ until the velocity of the end-effector (in robot space) reaches and keeps at a relative low value. Of course, this value is still larger than that of the object;

3) When $d(k) < d_2$, compute $V^{'}(k+1)$ via a PI controller (Proportional-plus-Integral controller). The integration term of PI controller will ensure that the static trajectory error between the end-effector and object converges to zero quickly. The PI controller can be expressed as

$$\begin{bmatrix} u^{'}(k+1) \\ v^{'}(k+1) \\ w^{'}(k+1) \end{bmatrix} = \begin{bmatrix} u(k) \\ v(k) \\ w(k) \end{bmatrix} + c_1 \left(\begin{bmatrix} x(k) - x^* \\ y(k) - y^* \\ z(k) - z^* \end{bmatrix} - c_2 \begin{bmatrix} x(k-1) - x^* \\ y(k-1) - y^* \\ z(k-1) - z^* \end{bmatrix} \right) \tag{6}$$

where $d_1 > d_2 > 0$, and c_1, c_2 are the proportional and integral gain of the PI controller respectively. Finally, notice that the output of the velocity generator is not the velocity $V^{'}(k+1)$, but the acceleration $\Delta V^{'}(k+1) = [\Delta u^{'}(k+1), \Delta v^{'}(k+1), \Delta w^{'}(k+1)]^T$, which is derived by $\Delta V^{'}(k+1) = V^{'}(k+1) - V(k+1)$.

4.2 Neural Network Transformer

A neural network is used to transform the planned acceleration $\Delta V^{'}(k+1)$ in feature domain into the motion instruction in 3-D robotic workspace. We build a BP neural network according to the inverse visual-mapping model described by equation (5). Fig.4 shows the structure of the ANN with 9 inputs and 3 outputs.

The weights of ANN are obtained by off-line learning. In order to acquire the training samples, we ask the manipulator to execute some random motion in the whole 3-D workspace. At the same time, we let the object keep static or move with a constant velocity in the workspace (in brief, the acceleration of object should be equal to zero in order to meet the assumption in Section 3). Then, we record the following parameters as training samples, including $[x(k), y(k), z(k)]^T, [u(k), v(k), w(k)]^T, [\Delta u(k), \Delta v(k), \Delta w(k)]^T$, and the acceleration $[\Delta T_{hX}(k), \Delta T_{hY}(k), \Delta T_{hZ}(k)]^T$ (see Fig.4).

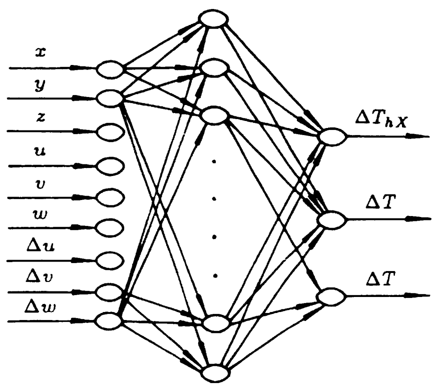

Fig.4 The structure of the neural network

When the neural network is used for tracking a moving object with variable velocity, its input should be changed to the following terms: $[x(k), y(k), z(k)]^T, [u(k), v(k), w(k)]^T$, and the acceleration $[\Delta u'(k+1), \Delta v'(k+1), \Delta w'(k+1)]^T$ planned by the trajectory generator. The output of the ANN is $[\Delta T'_{hX}(k+1), \Delta T'_{hY}(k+1), \Delta T'_{hZ}(k+1)]^T$, which is executed by the robot servo controller as the motion instruction (see Fig.2). Note that, here we have removed the previous assumption, $a_0(k) \equiv 0$, which is just satisfied in training period.

5. Simulation Results

Simulations are conducted by means of neural network toolbox of MATLAB 5.1. The range of the 3-D training space is as follows: $X \in [-0.2, 0.2], Y \in [-0.2, 0.2]$, and $Z \in [-0.2, 0.2]$.

Figs.5(a)~(b) show the coordinate curves of the manipulator tracking a moving object with constant and variable velocity. Notice that, in Fig.5, $[X_h, Y_h, Z_h]^T$ represents the coordinate of the end-effector (hand) in the base frame, while $[X_0, Y_0, Z_0]^T$ represents that of the object. The weights of the network are obtained by 3061 training samples via 20000 off-line iterations. The number of hidden units is 40. The roll angle and pitch angle of the camera frame relative to the end-effector frame are, respectively, 30° and 20°. The main parameters of the visual tracking controller are as follows: $c_1 = 40, c_2 = 0.5, d_1 = 0.003, d_2 = 0.001$. We can see that the tracking performance is perfect and the static trajectory error converges to zero rapidly. Additionally, although the object moves out of the range of training space (see Fig.5(a)), the manipulator still tracks it successfully.

Fig.5(c) shows the simulation results when the pose relationship between the camera and end-effector changes obviously. In this case, the roll angle and pitch angle have been changed to 0° and −20° respectively. Note that here we still adopt the previous neural network without being trained again.

Fig.5(d) shows the tracking performance when the neural network is trained insufficiently. The weights of ANN are obtained just by 638 samples via 1000 off-line iterations. The number of hidden units is 16. Although the neural network is trained roughly, the manipulator can still track the object effectively.

6. Conclusions

In this paper, a scheme for the control of a robotic manipulator with a visual sensor is described. We present a non-linear visual-mapping model to replace the image Jacobian and realize it with a neural network. The new method is suitable for the whole robotic workspace. Therefore, there is no necessity to update it on-line. Moreover, the introduced scheme does not require any type of calibration, so it can be used in a variety of industrial applications. Simulation results show that this method is feasible. We believe that similar techniques can be applied to the full 3-D tracking problem wherein there is coupling between rotation and translation.

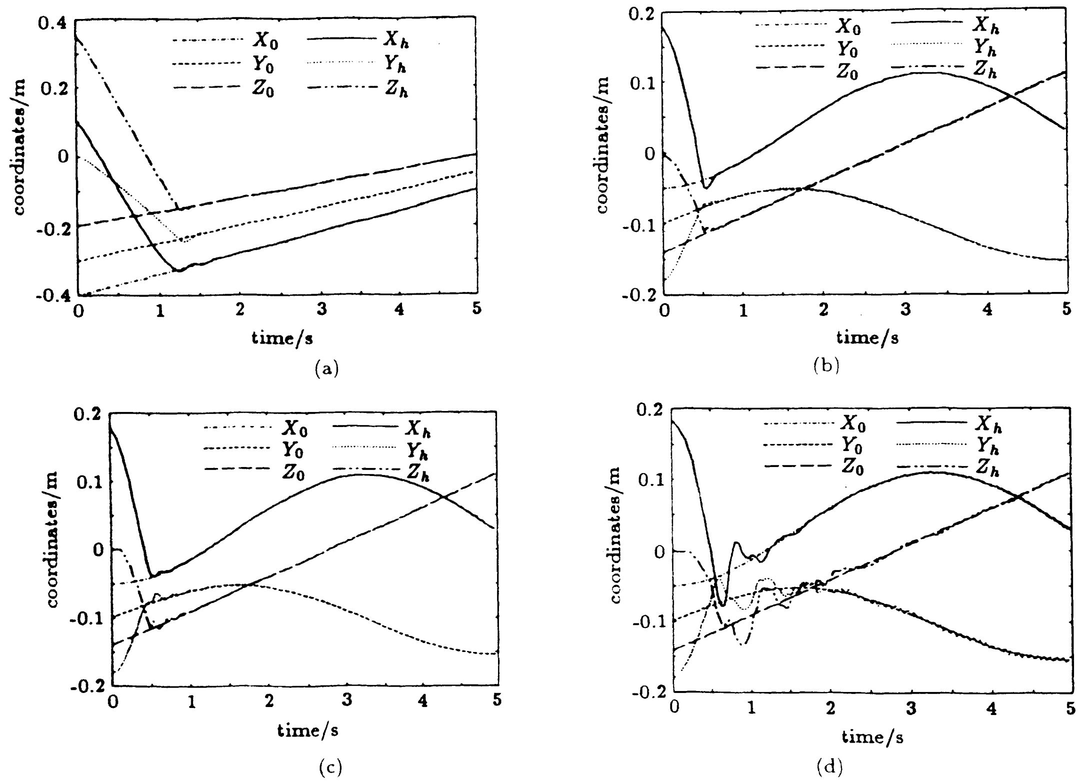

Fig.5 Simulation results

REFERENCES

[1] Hutchinson S, Hager D, Corke P. A Tutorial on Visual Servo Control. IEEE Trans. on Robot. Automat., 1996, 12: 651~670.

[2] Sanderson A C, Weiss L E, Neuman C P. Dynamic Sensor-Based Control of Robots with Visual Feedback. IEEE Trans. on Robot. Automat., 1987, 3: 404~417.

[3] Yoshimi B H, Allen P K. Alignment Using an Uncalibrated Camera System. IEEE Trans. on Robot. Automat., 1995, 11(4): 516~521.

[4] Hespanha J, Dodds Z, Hager G D et al. What Can be Done with an Uncalibrated Stereo Systems? Proc. 1998 IEEE Inter. Conf. on Robot. Automat., 1998: 1366~1372.

[5] Hager G. Calibration-Free Visual Control Using Projective Invariance. Proc. Int. Conf. Comupter Vision, 1995: 1009~1015.

[6] Sutanto H, Sharma R, Varma V. Image Based Autodocking Without Calibration. Proc. 1997 IEEE Inter. Conf. on Robot. Automat., 1997: 974~979.

[7] Scheering C, Kersting B. Uncalibrated Hand-Eye Coordination with a Redundant Camera System. Proc. 1998 IEEE Inter. Conf. on Robot. Automat., 1998: 2953~2958.

[8] Feddema J T, Geogre Lee C S. Adaptive Image Feature Predctionand Control for Visual Tracking with a Hand-Eye Coordinated Camera. IEEE Trans. on S. M. C., 1990, 20(5): 1172~1183.

[9] Miller W T. Sensor Based Control of Robotic Manipulators Using a General Learning Algorithm. IEEE Trans. on Robotics Automat., 1989, 3: 825~831.

[10] Hashimoto H, Kubota T, Sato M et al. Visual Control of Robotic Manipulator Based on Neural Networks. IEEE Trans. on Industrial Electronics, 1992, 139(6): 490~496.

[11] Feddema J T, Mitchell O R. Vision-Guided Servoing with Feature-Based Trajectroy Generation. IEEE Trans. on Robotics Automat., 1989, 5(5): 691~700.