International Journal of Soft Computing 3 (3): 201-204, 2008
ISSN: 1816-9503

Neural Networks Approach to 3D Rigid Motion Estimation from Feature Points

Yaming Wang, Li Sun, Jueliang Hu, Yunhua Zhang and Wenqing Huang
College of Informatics and Electronics, Zhejiang Sci-Tech University,
Xia-Sha High Education Zone, Hangzhou 310018, P.R. China

Abstract: 3D rigid motion estimation from images is crucial for many important applications. In this study, an approach is proposed for 3D rigid motion estimation from feature points using feed-forward neural-networks. The correspondence of feature points between consecutive images is assumed to be established beforehand. The proposed neural network is composed of 3 layers and 3 points are randomly selected from all points on the object to train the network using Newton-Raphson procedure. Experimental results from synthetic data are presented for validating the proposed approach.

Key words: Computer vision, motion estimation, 3D feature points, neural-networks

INTRODUCTION

3D rigid motion analysis is a fundamental problem in computer vision research area. Its importance stems from wide applications it has in surface matching, surface registration and motion pattern recognition (Srinark *et al*., 2006). In the literature, a large number of techniques have been developed to solve these problems. In general, the problem of 3D rigid motion estimation is an ill-posed problem and can be solved by adding constraints. Much of the existing work advanced can be divided into two methods, based feature correspondence and based optical flow. Huang (1986) estimated motion parameters by eight feature points or more. Longuet-Higgins (1981) estimated motion parameters by computing fundamental matrix. However, these methods have to match points firstly and decompose matrix which consumes large calculation. The Iterative Closest Point (ICP) algorithm by Besl and McKay (1992) is the most popular method. It has many derivatives improving the original one, e.g., using point-to-normal in the distance evaluation instead of point-to-closest point (Chen and Medioni, 1991). However, ICP based methods requires good initial approximations for their convergence.

In order to improve the limitations of the classical motion estimation methods, some researchers have started to use neural-networks to solve the estimation problem. Hutchinson *et al*. (1988) claimed that they could compute optical flow by injecting currents into resistive networks and recording the stationary voltage distribution at each node. Chen *et al*. (1995) used neural networks to estimate 3D rigid motion parameters based 3D feature points. Tzovars *et al*. (2000)used neural networks with adjusting weights by Newton-Raphson procedure to estimate 3D rigid motion parameters with initial 2D motion data. Chen's method used only one point to train network, which has problems of convergence in some case. Tzovars used CAHV camera model whose calibration is harder than the normal pinhole camera model.

RIGID MOTION MODEL AND CAMERA MODEL

3D rigid motion model: The rigidity assumption implies that the object motion can be decomposed into a rotation about a point termed the center of rotation and a translation of that center of rotation. Let us assume a 3D point of object p whose world coordinates is $P = (X, Y, Z)^T$ before motion. After motion, the coordinates is $P' = (X', Y', Z')^T$. The motion model is the following:

$$P' = RP + T \tag{1}$$

or

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = R \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + T \tag{2}$$

Where R is rotation matrix and T is translation vector.

The problem of 3D rigid motion estimation is to estimate rotation matrix R and translation vector T.

Camera model: Assuming intrinsic parameters of camera is:

$$\begin{pmatrix} f_x & 0 & u_x \\ 0 & f_y & u_y \\ 0 & 0 & 1 \end{pmatrix} \tag{3}$$

Corresponding Author: Yaming Wang, College of Informatics and Electronics, Zhejiang Sci-Tech University, Xia-Sha High Education Zone, Hangzhou 310018, P.R. China

Where we ignore the distortion of camera. Let us assume the projection of a 3D point P (X, Y, Z) onto the image plane is p (x, y) while the center of camera is the origin of world coordinate system. We have:

$$s\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & u_x \\ 0 & f_y & u_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{4}$$

or equally

$$\begin{cases} x = f_x X/Z + u_x \\ y = f_y Y/Z + u_y \end{cases} \tag{5}$$

3D RIGID MOTION ESTIMATION BASED FEATURE POINTS

Let us assume the correspondence of feature points is available beforehand, so the translation vector of rigid motion is simply the translation between 2 centroids (Moravec,1981).

$$T = \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix} = \begin{bmatrix} C'_x \\ C'_y \\ C'_z \end{bmatrix} - \begin{bmatrix} C_x \\ C_y \\ C_z \end{bmatrix} \tag{6}$$

Where$C1' = (C'_x, C'_y, C'_z)^T$ is the centroid of object after motion and $C = (C_x, C_y, C_z)^T$ is before motion. After estimating translation vector T, we can subtract T from every point of object to make sure only rotation motion is left.

$$P'' = P' - T = R.P \tag{7}$$

The proposed feed-forward neural-networks are composed of 3 layers as shown in Fig.1.

Instead of training the network only based on one point (Chen *et al*., 1995). We train the network based on 3 randomly selected points and a bigger neural-network composed of 3 small neural network are shown in Fig.1. The purpose of neural-networks is to minimize the following error measure:

$$\min \sum \| [P_1 - R \cdot P_1, P_2 - R \cdot P_2, P_3 - R \cdot P_3] \| \tag{8}$$

Where P_i (i = 1, 2, 3) is the points' coordinates which have subtracted translation vector and P_i (i = 1, 2, 3) is points' coordinates before motion.

In the neural-networks, the inputs of the first layer are three points' coordinates. In small neural network, the

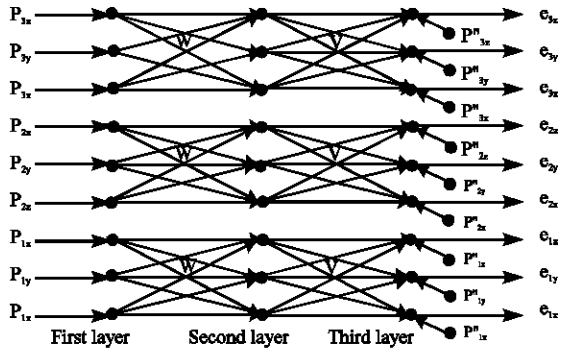

Fig. 1: The proposed neural-networks for 3D rigid motion estimation

weights between the first and the second layer is the rotation matrix:

$$W = \begin{bmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \end{bmatrix} \tag{9}$$

Where W_{ij} is weight between the No. j neuron of the first layer and the No. i neuron of the second layer, so W = R. The weights between the second and the third layer are constan,

$$V = \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix}. \tag{10}$$

Let $e_i = [e_{ix}, e_{iy}, e_{iz}]^T$ be the output of the small neural-network,

$$e_i = P_i - R.P \tag{11}$$

In order to minimize error measure, let the output vector be

$$E = [e_{1x}^2, e_{1y}^2, e_{1z}^2, e_{2x}^2, e_{2y}^2, e_{2z}^2, e_{3x}^2, e_{3y}^2, e_{3z}^2]^T \tag{12}$$

The motion parameter vector is:

$$W = [w_{11}, w_{12}, w_{13}, w_{21}, w_{22}, w_{23}, w_{31}, w_{32}, w_{33}]^T \tag{13}$$

In ideal case, the outputs of neural network are zero. Weights are updating by Newton-Raphson procedure:

$$W^{(n+1)} = W^{(n)} - \mu J^{-1} E \tag{14}$$

Where μ is learning rate and$J = \partial E/\partial W$ is Jacobian matrix which is calculated by the following,

Table 1: Results of rigid motion estimation

μ	Actual motion	Estimated result	Error
0.005	$R\begin{pmatrix} 0.69350493 & -0.33263814 & 0.63906705 \\ 0.63899374 & 0.69361573 & -0.33239859 \\ -0.33270556 & 0.63889432 & 0.69358337 \end{pmatrix}$	$\begin{pmatrix} 0.69353509 & -0.33259091 & 0.63905579 \\ 0.63905579 & 0.69353509 & -0.33259091 \\ -0.33259091 & 0.63905579 & 0.69353509 \end{pmatrix}$	0.000175
0.01	$R\begin{pmatrix} 0.69353509 & -0.33259091 & 0.63905579 \\ 0.63905579 & 0.69353509 & -0.33259091 \\ -0.33259091 & 0.63905579 & 0.69353509 \end{pmatrix}$	$\begin{pmatrix} 0.69358748 & -0.33233473 & 0.63902193 \\ 0.63881004 & 0.69364578 & -0.33266783 \\ -0.33271891 & 0.63899392 & 0.69351596 \end{pmatrix}$	0.000237
0.01	$R\begin{pmatrix} 0.41496068 & -0.77504909 & 0.47653762 \\ 0.48562741 & 0.63159001 & 0.60423452 \\ -0.76929653 & -0.01932884 & 0.63845146 \end{pmatrix}$	$\begin{pmatrix} 0.41492507 & -0.77503961 & 0.47660345 \\ 0.48553500 & 0.63161546 & 0.60441518 \\ -0.76947576 & -0.01937935 & 0.63838190 \end{pmatrix}$	0.000170
0.01	$R\begin{pmatrix} 0.69353509 & -0.33259091 & 0.63905579 \\ 0.63905579 & 0.69353509 & -0.33259091 \\ -0.33259091 & 0.63905579 & 0.69353509 \end{pmatrix}$	$\begin{pmatrix} 0.69350070 & -0.33240741 & 0.63910413 \\ 0.63898039 & 0.69355994 & -0.33265257 \\ -0.33270904 & 0.63906664 & 0.69342661 \end{pmatrix}$	0.000156

$$J = \partial E/\partial W = \begin{bmatrix} 2e_{1x}P_1x & 2e_{1x}P_1y & 2e_{1x}P_1z & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2e_{1y}P_1x & 2e_{1y}P_1y & 2e_{1y}P_1z & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2e_{1z}P_1x & 2e_{1z}P_1y & 2e_{1z}P_1z \\ 2e_{2x}P_2x & 2e_{2x}P_2y & 2e_{2x}P_2z & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2e_{2y}P_2x & 2e_{2y}P_2y & 2e_{2y}P_2z & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2e_{2z}P_2x & 2e_{2z}P_2y & 2e_{2z}P_2z \\ 2e_{3x}P_3x & 2e_{3x}P_3y & 2e_{3x}P_3z & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2e_{3y}P_3x & 2e_{3y}P_3y & 2e_{3y}P_3z & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2e_{3z}P_3x & 2e_{3z}P_3y & 2e_{3z}P_3z \end{bmatrix} \quad (15)$$

RESULTS

In the simulation experiment, feature points are randomly generated. The range of coordinates are between (0, 200) and the intrinsic parameters of camera are assumed as:

$$\begin{pmatrix} 10 & 0 & 160 \\ 0 & 10 & 120 \\ 0 & 0 & 1 \end{pmatrix}$$

As the rotation parameters have only three degrees of freedom, it is generated and converted to rotation matrix using Eq. (16) as follows:

$$\begin{aligned} r &= r\,|\,r\,| \\ R &= \cos(|\,r\,|)\cdot I + (1-\cos(|\,r\,|))rr^T + \sin(|\,r\,|)[r]x \end{aligned} \quad (16)$$

Where $|\,r\,|$ is the norm of rotation vector r and

$$[r]_\times = \begin{bmatrix} 0 & -r_z & r_y \\ r_z & 0 & -r_x \\ -r_y & r_x & 0 \end{bmatrix} \quad (17)$$

The results of 3D motion estimation are shown in the Table 1. It should be point out that the estimated translation vector is almost right, so it is not listed in Table 1. Meanwhile the estimated results are compared with the actual and error is calculated by following equation:

$$\text{error} = \frac{\|\,\text{estimate} - \text{true}\,\|}{\|\,\text{true}\,\|} \quad (18)$$

Where *estimate* is the estimated rotation matrix and *true* is the actual motion parameters.

CONCLUSION

In this study, the neural-networks approach is proposed to 3D rigid motion estimation problem based on feature points. Three points are randomly selected to train the neural-networks. The weights are adjusted by Newton-Raphson procedure. Experimental results show good robustness of our approach.

ACKNOWLEDGEMENT

This project is supported by the National Science Foundation of China under grants 60473038 and 60773204.

REFERENCES

Besl, P.J. and N.D. McKay, 1992. A method for registration of 3-d shapes. IEEE Trans. Pattern Anal. Machine Intelligence, pp: 239-256.

Chen, T., W.C. Lin and C.T. Chen, 1995. Artificial neural networks for 3-D motion analysis I: Rigid motion. IEEE. Trans. Neural Networks, pp: 1386-1393.

Chen, Y. and G. Medioni, 1991. Object modeling by registration of multiple range images. Int. Conf. Robotics and Automation, Sacramento, CA, USA., pp: 2724-2729.

Huang, T.S., 1986. Handbook of Pattern Recognition and Image Processing Chapter14: Determining 3 dimensional motion and structure from 2 perspective views. Academic Press.

Hutchinson, J., C. Koch, J. Luo and C. Mead. Computing Motion Using analog and binary resistive networks. IEEE. Comput. Mag., pp: 52-63.

Longue-Higgins, H.C., 1981.A computer algorithm for reconstructing a scene from 2 projectios. Nature, pp: 133-135.

Moravec, H.P., 1981. Robot Rover Visual Navigation. UMI Research Press.

Srinark, T., C. Kambhamettu and M. Stone, 2006. A hierarchical method for 3D rigid motion estimation. Asian Conference on Computer Vision (ACCV), Hyderabad, India, pp: 791-800.

Tzovaras, D., N. Ploskas and M.G. Strintzis. Rigid 3-D motion estimation using neural networks and initially estimated 2-D motion data. IEEE. Trans. Circuits Sys. Video Technol., pp: 158-165.