International Journal of the Physical Sciences Vol. 6(21), pp. 5005-5011, 30 September, 2011
Available online at http://www.academicjournals.org/IJPS
DOI: 10.5897/IJPS11.1068
ISSN 1992 - 1950 ©2011 Academic Journals

Full Length Research Paper

Parallel robots pose accuracy compensation using back propagation network

Dayong Yu

School of Aeronautics and Astronautics, Zhejiang University, Hangzhou 310027, China. E-mail: wy_ydy@yahoo.com.cn. Tel: +86-571-87952100. Fax: +86-571-87951391.

Accepted 1 August, 2011

In order to improve pose accuracy of a parallel robot in the application, a compensator is constructed to predict leg length errors using back propagation neural network. In this method, the back propagation neural network is used with conventional inverse kinematics computation module in parallel. A back propagation neural network is designed and implemented to learn kinematic model errors for parallel robots. The non-linear mapping from the operation variable space for the mobile platform to the joint variable space is accomplished solving the location and posture. The trained neural network can be used to performed on-line pose accuracy compensation in the task. Simulation and experimental results show that this method provides a good pose accuracy improvement and keeps good robustness and adaptability at the same time.

Key words: Parallel robot, pose accuracy, back propagation network, kinematic calibration, error compensation.

INTRODUCTION

Many comprehensive studies and works have been made in the area of parallel robots (Daney, 2003). Parallel robots have the following advantages when compared with serial robots: great dynamic capabilities and rigidity, a high positioning repeatability, and a high positioning accuracy if the actual parameters are known (Masory et al., 1993). The desired poses of a robot are normally specified in Cartesian space, while these poses are achieved by controlling joint variables in the robot's joint space. The transformation from Cartesian space to joint space is called inverse kinematics in robotics. The inverse kinematics is a computationally intensive procedure, the accurate solution of which depends on precise knowledge of the robot parameters. However, due to manufacturing tolerance, assembly errors, wear and tear, transmission errors, compliance, etc., the internal design model used in the robot controller will not describe the inverse kinematics accurately. Therefore, the actual poses achieved by controlling the joint values, obtained from the controller's internal model, will deviate from the desired poses. The solution to compensate this loss of pose accuracy is known as kinematics calibration (David and Emiris, 2001; Abtahi et al., 2009; Pashkevich et al., 2009; Wang et al., 2011).

Kinematics calibration involves the following procedures: (1) set up an appropriate kinematics model; (2) take measurements of the robot pose; (3) identify the actual kinematics parameters to minimize the errors between the poses predicted by the model and the actual measured ones; (4) implement the identified robot kinematics model. In general, accuracy is defined by repeatability and bias (ANSI, A15). Lack of repeatability is due to random error, and it is quantified by the variance of a number of measurements. Bias is a systematic error and it is determined by the mean value. While it is difficult to compensate for the random error, compensation for the systematic error could be done effectively by means of calibration. For parallel robots, many calibration methods have been proposed. Zhuang and Roth (1991) proposed a method to calibrate a 42 parameter model of a Stewart platform. Their idea was to acquire special measurement sets which allow the decomposition of linear sub-models based on Stewart platforms of the error model. Linear sub-models offer the advantage that identification of the kinematic parameters becomes straight forward, no initial guess is needed and the optimum is global. This method of holding one leg at constant legs was further improved by Khalil and Murareci (1997) who combined it with an idea of Daney et al. (2004) to keep the direction of the leg fixed during data acquisition. A method that is currently widely accepted is the double ball bar system. With this system, three position data can be collected for each measurement and pose can be computed simultaneously using three-point method. Wampler et al. (1995) developed a slightly different type of calibration based on implicit loops. By applying five additional passive sensors on one leg, the forward kinematics can be computed so that closed form loop equations can be formed for the remaining five legs, and the calibration algorithm uses this additional data to solve for the kinematic parameters. This is not different from having an independent measure of the manipulator pose. A method to use redundant sensors on passive joints to calibrate parallel manipulators was proposed (Zhuang and Liu, 1996). Redundant sensor data is obtained from as few as three additional sensors. To solve the forward kinematics, the authors implement a numerical method that solves for all joint variables, both passive and active. This allows the formation of measurement residual for the passive measured joints, thus a costs function that is minimized. By imposing appropriate physical constraints on the passive joints, the kinematic parameters of parallel manipulators can be identified only with the measurement data obtained from the actuators. Khalil and Besnard (1999) reported that locking universal and/or spherical joints, with some locking mechanisms, could calibrate Stewart platform autonomously. The locking mechanisms must be very stiff in order to prevent the joint and the link from bending deformation.

This paper presents pose accuracy compensation method based on an artificial neural network. In this method, an artificial neural network is used with conventional inverse kinematics computation module in parallel. The network can automatically learn the internal design model error. The neural network output is added to the inverse kinematics computation module output in a joint coordinate. The joint coordinate is used to drive the robot. To demonstrate the feasibility of the proposed approach, poses compensated by the trained network are compared with actual poses. It shows that the neural network can effectively improve parallel robot pose accuracy.

POSE ACCURACY COMPENSATION

Artificial neural networks and back propagation network

The inspiration for neural network comes from researches in biological neural networks of the human brains. Artificial neural network is a one of those approaches to imitate the mechanisms of learning and problem solving functions of the human brain which is flexible, highly parallel, robust and fault tolerant. Artificial neural networks are widely accepted as a technology offering an alternative way to tackle complex and ill-defined problems. The structure of this information processing system is composed of highly interconnected processing elements, called neurons working in parallel to solve problems. A neural network helps when it is highly complex to formulate an algorithmic solution and also where there is a need to pick out the structure from the existing data. Neural networks learn by example and they cannot be programmed to perform a specific task. They are fault-tolerant, that is, they are able to handle noisy and incomplete data that are able to deal with non-linear problems and once trained can assist in prediction and generalization at high speed.

In artificial neural networks implementation, knowledge is represented as numeric weights, which are used to gather the relationship within data that are difficult to relate analytically, and it iteratively adjusts the network parameters to minimize the sum of squared approximation errors using a gradient descent method. Neural networks can be used to model complex relationship without using simplifying assumptions, which are commonly used in linear approaches. In more practical terms, neural network is a non-linear statistical data modeling tool. The tasks for which artificial neural networks that are useful fall into various applications such as control, pattern recognition, forecasting, optimization, etc. In this study, artificial neural network is applied for the prediction of leg length errors from normal pose.

One of its implementation is a back propagation network that is trained with supervision, using gradient-descent training technique that minimizes the squared error between the actual output of the network and the desired outputs. In the back propagation network, a network consists of many non-linear computational elements called nodes. Each node can take many inputs but has a single output, which can fan out to other nodes in the next level. Two nodes are interconnected via a link, which is a one-way connection. A link takes the output value of a node, transforms it and then submits the outcome as an input to another node. Each connection to a node is associated with a quantity called a connection strength or weight. The nodes are arranged in three different layers. The bottom layer is the input layer of nodes. The top layer is the output layer of nodes. The hidden layer can have more than one layer between the input and the output layer. Nodes in the input layer receive the values of the input variables and propagate upward to the network, layer by layer. The output nodes at the output layer form the output variables. Figure 1 shows a three-layer back propagation neural network. The iterative gradient algorithm is performed to minimize the mean square error between the desired output and the actual output of a feed-forward network. It requires continuous differentiable non-linearities (Fukuda, 1992).

Controller configuration

In this method, an artificial neural network is used in parallel with the conventional inverse kinematics

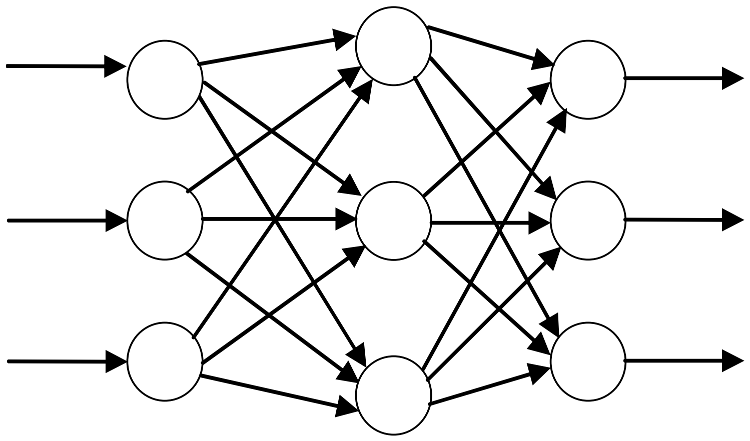

Figure 1. A three-layer back propagation neural network.

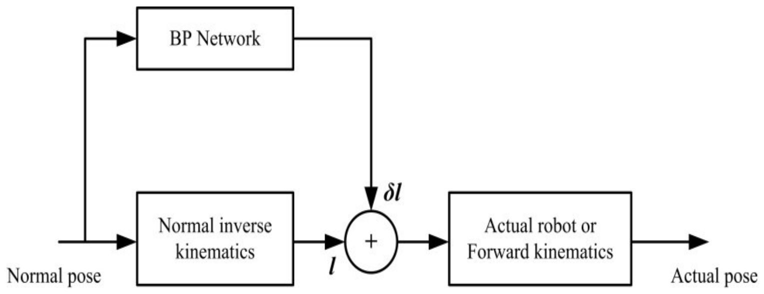

Figure 2. Pose accuracy compensation based on ANN.

computation module. Figure 2 shows the robot controller configuration.

The conventional inverse kinematics computation module *l*(*X*) and the network δ*l*(*X*) were provided with a desired pose *X*. Here, the pose is specified with position (P_x, P_y, P_z) and orientation (O_x, O_y, O_z).

The network consists of an input layer, a hidden layer and an output layer, as shown in Figure 1. The input layer and the output layer are made up of six nodes and six nodes, respectively. Number of the hidden layer was 100. Output functions for the input layer and the hidden layer were sigmoid functions. That is, the output layer was a linear function.

Node outputs in the network of each layer relate to a sum of weighted input level. Outputs in each layer are obtained by next equations.

$$O_i^k = f(N_i^k) \tag{1}$$

$$N_i^k = \sum_j w_{ij}^{kk-1} \cdot O_j^{k-1} - \theta_i^k \tag{2}$$

where O_i^k : *i*th node output vales in layer k, w_{ij}^{kk-1} : modifiable weight element between *i*th node in layer k and jth node in lay k-1, θ_i^k : *i*th offset value in layer k, $f(x)$: $1/(1+e^{-x})$ [Input layer and hidden layer], x : [Output layer].

Six desired pose elements, position (P_x, P_y, P_z) and orientation (O_x, O_y, O_z), are used as the input value. Six nodes value of the output layer was used as correcting joint coordinate δ*l*.

Learning procedure

The error back propagation model has been adopted as the network learning method. On the learning procedure, the difference between the desired pose *X* and the measured pose *Y* is used for the supervising signal as δ*X*. Because, the network output should be described in the joint coordinate, the supervising signal in the joint coordinate is derived from δ*X* by Equation 3.

$$\delta l = J \cdot \delta X \tag{3}$$

Here,

$$\delta X = Y - X \tag{4}$$

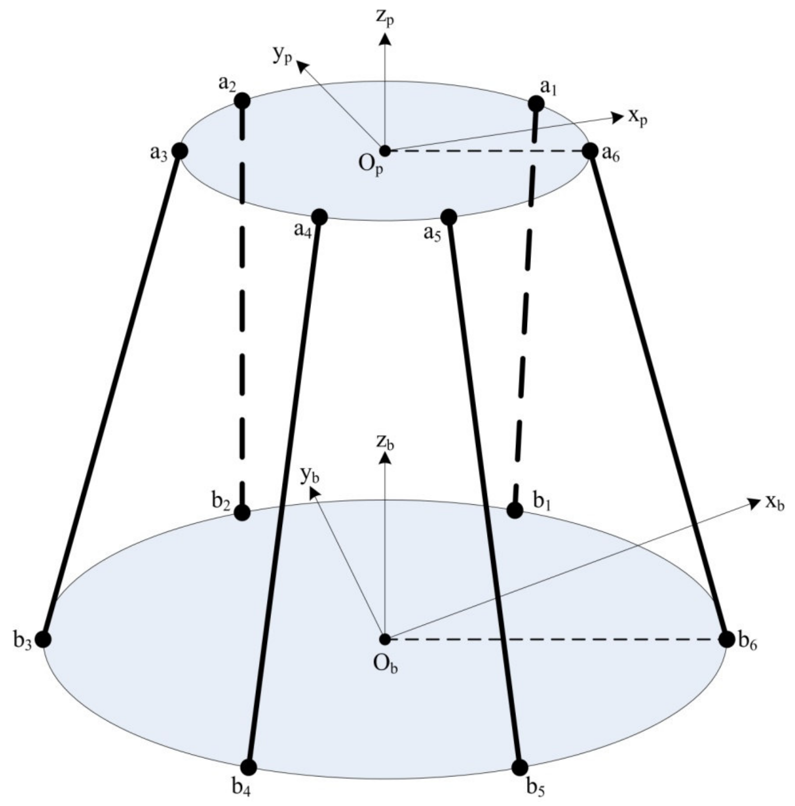

Figure 3. Parallel robots.

J is the Jacobian matrix for the robot at desired pose *X* and was efficiently derived by Ropponen' method (Ropponen and Arai, 1995).

In order to learn the kinematics model error in the network, the weight elements w_{ij}^{kk-1} are modified using Equation 5.

$$w_{ij}^{kk-1} = w_{ij}^{kk-1} - \varepsilon d_i^k \cdot O_j^{k-1} \tag{5}$$

Once the robot model error is learned on link weight elements through the learning procedure, the network generates compensation joint lengths, even if the desired pose is not included in the learning data set.

MODELING AND KINEMATICS OF PARALLEL ROBOTS

Here, we describe the parallel robot and its kinematics model. The robot consists of two rigid bodies, the base and the mobile platform, connected by 6 legs. The leg linear actuator provides 6° of freedom for the platform pose relative to the base, corresponding to position *P* and rotation matrix *R*. A pose *X*=[*P*, *R*] is associated to 6 length variations l_i measured by internal leg sensors, *i*=1, …, 6.

Each leg is attached to the base by a hook joint and to the platform by a hook joint; 23 parameters are required to model each leg. But as shown in Masory et al. (1997), the principal source of error in positioning is due to limited knowledge of the joint centers and to the fact that the part of the length is not given by the sensors. We thus, use a simpler model with attachment point's a_i in the mobile frame, b_i in the reference frame and offset lengths l_i. This gives 7 parameters per leg, therefore 42 overall, denoted by *ρ*.

Inverse kinematics

The inverse kinematics problem of the parallel robot, deals with calculating the leg lengths when the pose is given and the kinematics parameters are known. In effect, it is a mapping from global pose to local leg transducer readings. The inverse kinematics of a parallel robot is uncomplicated, yielding a non-linear closed form solution. The vector chain in Figure 3 can be expressed as:

$$l_i = Ra_i + P - b_i \tag{6}$$

The length of leg i can then be determined by taking the

Table 1. Normal parameters of the parallel robot (mm).

Joint number	a_{ix}	a_{iy}	a_{iz}	b_{ix}	b_{iy}	b_{iz}	$l_{0,i}$
1	544.7	130	-200	784.21	908.3	100	1830
2	-159.77	536.73	-200	394.5	1133.3	100	1830
3	-384.93	406.73	-200	-1178.7	225	100	1830
4	-384.93	-406.73	-200	-1178.7	-225	100	1830
5	-159.77	-536.73	-200	394.5	-1133.3	100	1830
6	544.7	-130	-200	784.21	-908.3	100	1830

magnitude of Equation 6.

$$\lambda_i = \|l_i\| = \|Ra_i + P - b_i\| \tag{7}$$

And the leg sensor reading can be obtained by;

$$s_i = \lambda_i - l_{0,i} \tag{8}$$

Forward kinematics

For the parallel robots, the forward kinematics is difficult to compute since it consists in solving Equation 6 for *P* and *R* given l_i and *ρ*.

Defining the vector function to describe the difference between the estimated sensor reading (s_i) and the actual sensor reading ($\hat{s}_i$).

$$f = \begin{bmatrix} f_1 \\ \vdots \\ f_6 \end{bmatrix} = \begin{bmatrix} s_1^2 - \hat{s}_1^2 \\ \vdots \\ s_6^2 - \hat{s}_6^2 \end{bmatrix} \tag{9}$$

The Newton-Raphson algorithm can be stated as:

1. Measure $\hat{s}$ and select an initial guess for the pose *X*
2. Compute s based on X_0
3. Form *f*
4. If $X^T X$<tolerance$_1$, exit with *X* as the solution
5. Compute the partial derivative matrix $J = \partial f / \partial X$ such that $J_{i,j} = \partial f_i / \partial X_j$
6. Solve for the update δ*X* from *J*δ*X*=-*f*
7. If $\delta X^T \delta X$<tolerance$_2$, exit with *X* as the solution
8. Update *X* by *X*=*X*+δ*X* and go to step 2

In steps one, an initial pose vector *X* must be guessed. This is usually taken as the last pose of the mobile platform. In step two, the estimated length can be computed with the inverse kinematics. Step three and four are straightforward, with *f* formed through Equation 9 and tolerance1 being the allowed error in the pose calculation. The partial derivatives required in step five can be computed. Step six involves a 6 by 6 matrix inversion to calculate δ*X*, and then in step seven, the norm of δ*X* is tested to see if the update is significant. If the update is considered significant, then the algorithm repeats from step two with the update pose vector.

SIMULATIONS AND EXPERIMENTAL RESULTS

Simulations using the kinematics model have been performed according to the steps as follows:

1. Preparation of data for neural network training
a. Calculate joint length set *l* from a set of poses *X*, using the inverse kinematics (Equation 7).
b. Determine actual poses *Y* from *l*, using the forward kinematics (Equation 9), which includes model error.
2. Training and testing of neural network
a. Building a network
b. Training a network (carry out the error back propagation learning with *X*, *l*, *Y*)
c. Testing a network
d. Save a network
3. Evaluation of neural network
a. Various poses *X'*, are given to the network and the inverse kinematics module. The sum of *l* and δ*l* is sent to the forward kinematics.
b. The compensated poses, *Y'*, are derived by Equation 9, which includes model error.

The normal kinematics parameters used for compensation are listed in Table 1. The actual kinematics parameters are simulated, with the parameter deviations obtained from uniform distribution with variance of 2 mm and listed (Table 2).

A precision coordinate measuring machine (CMM) is to measure poses of the parallel robot. As the work is concerned here with robot pose accuracy in the workspace, where most fine operations are executed, an area of 200 by 300 by 300 mm in Cartesian space and 10° by 10° by 10° in the orientation space (Euler angles) has been chosen as compensation area. 200 data points, uniformly distributed in the compensation area, have been collected for training, testing and evaluation.

The power of neural networks is due to their large generalization ability. 100 data points are randomly

Table 2. Actual parameters of the parallel robot (mm).

Joint number	a_{ix}	a_{iy}	a_{iz}	b_{ix}	b_{iy}	b_{iz}	$l_{0,i}$
1	543.42	130.28	-200.13	785.23	906.91	100.86	1831.6
2	-160.2	534.84	-200.63	395.87	1134.4	101.99	1830.7
3	-383.03	407.03	-198.92	-1178.4	224.44	99.496	1828.5
4	-384.43	-405.79	-201.4	-1176.8	-223.45	98.743	1829.9
5	-160.11	-538.45	-198.46	393.48	-1134.3	98.695	1831.9
6	546.03	-130.74	-198.41	783.55	-908.67	100.5	1830.1

Table 3. Compensation results for the parallel robot (position in mm and orientation in degrees).

Statistical measures	Before compensation		Kinematics calibration		Neural network compensation	
	Position	Orientation	Position	Orientation	Position	Orientation
Average	0.9550	0.4405	0.4298	0.1910	0.2317	0.0982
SD	0.0857	0.0309	0.0403	0.0206	0.0219	0.0017
Max	1.1508	0.5475	0.5754	0.3452	0.4218	0.2837

Table 4. Experimental evaluation of compensation results (position in mm and orientation in degrees).

Statistical measures	Position error in length		
	Before compensation	Kinematics calibration	Neural network compensation
Average	1.0502	0.8712	0.7524
SD	0.1081	0.0721	0.0709
Max	1.478	1.0493	0.9418

chosen from the collected data set of 200 points for neural network training. The trained network can generalize well in the compensated area. Three statistical measures (average error, standard deviation and maximum deviation) are used to evaluate the robot pose accuracy compensation results. Using kinematics calibration and neural network, compensation are listed as shown in Table 3, based on 100 randomly chosen test data points from the whole data set. The compensated positions and orientations are calculated using the compensated kinematics model, and were then compared with the data collected.

As shown in Table 3, it can be seen that the neural network accuracy compensation can achieve accuracy improve factor of about 5. The compensation based on neural network has more satisfactory results in terms of accuracy when compared with kinematics calibration.

Table 4 lists the experimental evaluation results of joints compensations based on 20 test points across the compensation area. The positioning errors (expressed in x, y and z components) before compensation are obtained by measuring the positions that the robot achieved by controlling the joints lengths as recommended by the robot controller. The positioning errors after compensation are obtained by measuring the positions achieved by controlling the joints lengths, updated by different compensation approached. The average position error (in length) decreased from 4.20 mm before compensation, to 1.52 mm after kinematics calibration, to 1.57 mm after the neural network compensation. The improvements in accuracy indicated by the experimental results are less significant when compared with the improvement shown by the simulation results. This can partially be explained by the fact that the measurements for compensation and the measurements for evaluation were made at different times, system error may therefore have occurred in the coordinate measuring machine. Of course, robot repeatability contributed to the final residual error. However, experimental results show that the neural network compensation approach can improve parallel robot pose accuracy.

Conclusions

The simulation and experiment results show the effectiveness of the proposed method. This simple computation scheme improves the parallel robot pose accuracy. Also, a simple back propagation network can learn highly non-linear function, and has been applied successfully to approximate the complex mapping between robot poses and robot joints length-compensations.

A neural network compensation approach eliminates the identification procedure, and compensation can be implemented on-line.

REFERENCES

Daney D (2003). Kinematic calibration of the Gough platform. Robotica. 21(4): 677-690.

Masory O, Wang J, Zhuang HQ (1993). On the accuracy of a Stewart platform–part II kinematic calibration and compensation. Proc. 1993 IEEE Int. Conf. Robot. Automat., Atlanta, 1: 725-731.

David D, Emiris IZ (2001). Robust parallel robot calibration with partial information. Proc. 2001 IEEE Int. Conf. Robot. Automat., Seoul, 1: 3262-3267.

Abtahi M, Pendar H, Alasty A, Vossoughi GHH (2009). Calibration of parallel kinematic machine tools using mobility constraint on the tool center point. Int. J. Adv. Manuf. Technol., 45(5-6): 531-539.

Pashkevich A, Chablat D, Wenger P (2009). Kinematic calibration of Orthoglide-type mechanisms from observation of parallel leg motions. Mechatronics, 19(4): 478-488.

Wang LP, Xie FG, Liu XJ, Wang LS (2011). Kinematic calibration of the 3-DOF parallel module of a 5-axis hybrid milling machine. Robotica. 29(4): 535-546.

American National Standards Institute. ANSI A15 standards on finding target features with optical equipment.

Zhuang HQ, Roth Z (1991). A method for kinematic calibration of Stewart platforms. Proc. ASME Annu. Winter Meet., Atlanta, 1: 43-48.

Khalil W, Murareci D (1997). "Autonomous calibration of parallel robots." Proc. 5th Symp. Robot Control, pp. 405-410.

Daney D, Papegay Y, Neumaier A (2004). Interval methods for certification of the kinematic calibration of parallel robots. Proc. of the 2004 IEEE Int. Conf. Robot. Automat., New Orleans, 1: 1913-1160.

Wampler CW, Hollerbach JM, Arai T (1995). An implicit loop method for kinematic calibration and its application to closed-chain mechanisms. IEEE Trans. Robot. Automat., 11(5): 710-724.

Zhuang HQ, Liu L (1996). Self-Calibration of a class of parallel manipulators. Proc. IEEE Int. Conf. Robot. Automat., Minneapolis, 1: 994-999.

Khalil W, Besnard S (1999). Self calibration of Stewart-Gough parallel robots without extra sensors. IEEE Trans. Robot. Automat., 15(6): 1116-1121.

Fukuda T (1992). Theory and applications of neural networks for industrial control systems. IEEE Trans. Ind. Electron., 39(6): 472-489.

Ropponen T, Arai T (1995). Accuracy analysis of a modified Stewart platform manipulator. Proc. 1995 IEEE Int. Conf. Robot. Automat., Nagoya, 1: 521-525.

Masory O, Wang J, Zhuang HQ (1997). Kinematic modeling and calibration of a Stewart platform. Adv. Robot., 11(5): 519-539.