



ANFIS-Based Visual Pose Estimation of Uncertain Robotic Arm Using Two Uncalibrated Cameras

Aung Myat San^{*}, Wut Yi Win, Saint Saint Pyone

Department of Mechatronic Engineering, Mandalay Technological University, Mandalay, Myanmar

Email address:

agmyatsanmc@googlemail.com (A. M. San), wutyiwin@gmail.com (W. Yi Win), saintsaint85@gmail.com (S. S. Pyone)

^{*}Corresponding author

To cite this article:

Aung Myat San, Wut Yi Win, Saint Saint Pyone. ANFIS-Based Visual Pose Estimation of Uncertain Robotic Arm Using Two Uncalibrated Cameras. *International Journal of Wireless Communications and Mobile Computing*. Vol. 6, No. 1, 2018, pp. 20-30.

doi: 10.11648/j.wcmc.20180601.13

Received: January 2, 2018; **Accepted:** January 17, 2018; **Published:** February 6, 2018

Abstract: This paper describes a new approach for the visual pose estimation of an uncertain robotic manipulator using ANFIS (Artificial Neuro-Fuzzy Inference System) and two uncalibrated cameras. The main emphasis of this work is on the ability to estimate the positioning accuracy and repeatability of a low-cost robotic arm with unknown parameters under uncalibrated vision system. The vision system is composed of two cameras; installed on the top and on the lateral side of the robot, respectively. These two cameras need no calibration; thus, they can be installed in any position and orientation with just the condition that the end-effector of the robot must remain always visible. A red-colored feature point is fixed on the end of the third robotic arm link. In this study, captured image data via two fixed-camera vision system are used as the sensor feedback for the position tracking of an uncertain robotic arm. LabVolt R5150 manipulator in our laboratory is used as case study. The visual estimation system is trained using ANFIS with subtractive clustering method in MATLAB. In MATLAB, the robot, feature point and cameras are simulated as physical behaviors. To get the required data for ANFIS, the manipulator was maneuvered within its workspace using forward kinematics and the feature point image coordinates were acquired with the two cameras. Simulation experiments show that the location of the robotic arm can be trained in ANFIS using two uncalibrated cameras; and problems for computational complexity and calibration requirement of multi-view geometry can be eliminated. Observing Mean Square Error (MSE), Root Mean Square Error (RMSE), Error Mean and Standard Deviation Errors, the performance of the proposed approach is efficient for using as visual feedback in uncertain robotic manipulator. Further, the proposed approach using ANFIS and uncalibrated vision system has better in flexibility, user-friendly manner and computational concepts over conventional techniques.

Keywords: ANFIS, Forward Kinematics, Two Uncalibrated Cameras, LabVolt R5150 Robot

1. Introduction

The positioning problem of robot manipulators using visual information has been an area of research over the last 40 years. Attention to this subject has drastically grown in recent years. The feedback loop using visual information can solve many problems that limit applications of current robots: automatic driving, long range exploration, medical robotics, aerial robots, etc.

Neural networks are good candidates for approximating non-linear transformation functions because they possess the following desirable features. Firstly, neural networks have the capability to learn from experience. They do not require

explicit programming to acquire the approximate model. Secondly, neural networks may approximate arbitrary non-linear mappings subject to the availability of unlimited number of processing units. Thirdly, because of their massive parallel architecture, the data processing is fast. In the field of robotics, neural networks have been applied in the following problems: to solve the inverse kinematic problem of robots, to map the non-linear relationships in robot dynamics as an inverse dynamics controller, in path or trajectory planning, to map sensory information for robot control and in task planning and intelligent control.

This paper focuses about mapping visual sensory information for robot control. Recently, the three-dimension

(3-D) vision systems for robot applications have been popularly studied. Baek and Lee [9] used two cameras and one laser sensor to recognize the elevator door and to determine its depth distance. Okada et al. [10] used multi-sensors for the 3-D position measurement. Winkelbach et al. [11] combined one camera with one range sensor to find the 3-D coordinate position of the target. Huang [4] addressed a 3-D position control for a robot arm utilizing two-CCD vision geometry and inverse kinematics. Zhou et al. [5] used position sensitive detector (PSD) for high-precision parallel kinematic mechanisms (PKMs) in order to allow them to accurately achieve their desired poses. Dallej et al. [12] developed 3D pose visual servoing for cable driven parallel robots.

An attractive approach is to have a system which learns the nonlinear relationship between the observed 2D feature deviations and the robot moments. Skaar et al. [6] developed a method for learning the image Jacobian, by way of least-squares estimation, from several observations of cues along the approach trajectory. The method was successfully applied to a part-mating task. Neural networks have been applied in many areas of robot control, as described by Torras [13]. Hashimoto et al. [7] used a neural network to learn the direct mapping between the image deviations of four feature points and the joint angles of a 6-dof manipulator. A disadvantage to including the inverse kinematics in the mapping is that the learned relationship is pose-dependent, i.e. it only applies for positioning with respect to the target object in a particular location. In Wells' observation [8], a neural network is used to learn the pose-independent mapping between feature deviations and pose-changes based images sampled from the workspace. Cid et al. [14] developed fixed-camera visual

servoing for planar robot manipulators composing control laws by the gradient of an artificial potential energy plus a nonlinear velocity feedback.

In this paper, the positioning problem of 5-DOF articulated robot manipulators is addressed under two fixed cameras configurations. The main contribution is the development of a new pose-independent learning method for the robotic end-effector positioning using two uncalibrated fixed cameras and robotic forward kinematics. The objective concerning the control is defined in terms of cartesian coordinates which are deduced from visual information.

The paper is organized by six sections. In section 2, the analysis of the R5150 robotic manipulator is performed with the forward kinematic modelling along with the mathematical treatment along with the development of the link coordinate diagram and the kinematic parameters. The theory of ANFIS technique is presented in section 3. In section 4, implementation of proposed system is described. Experimental tests and results are presented in section 5. Finally, the paper is concluded in section 6 with the observed results and future work.

2. Robotic Forward Kinematic Analysis

In this section, the forward kinematic analysis of a robot is described determining D-H parameters and calculating robot forward kinematics. To get the physical robotic model for simulating a robot in MATLAB, the link lengths and joint types of the LabVolt R5150 manipulator are modelled in Figure 1; and the link frame assignments of the robot is shown in Figure 2.

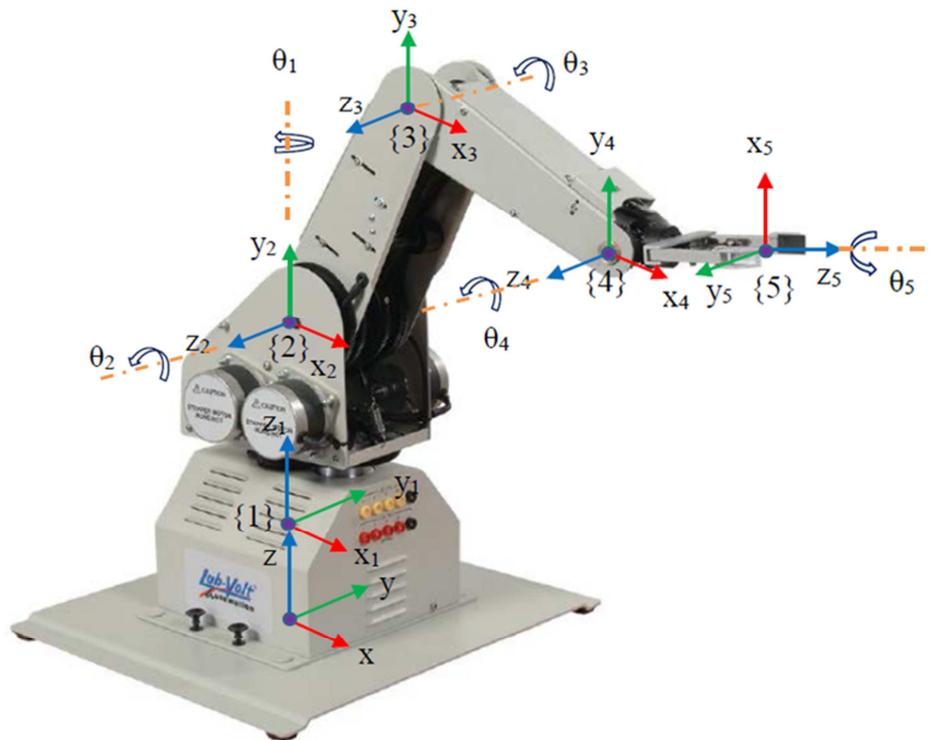


Figure 1. LabVolt R5150 robot.

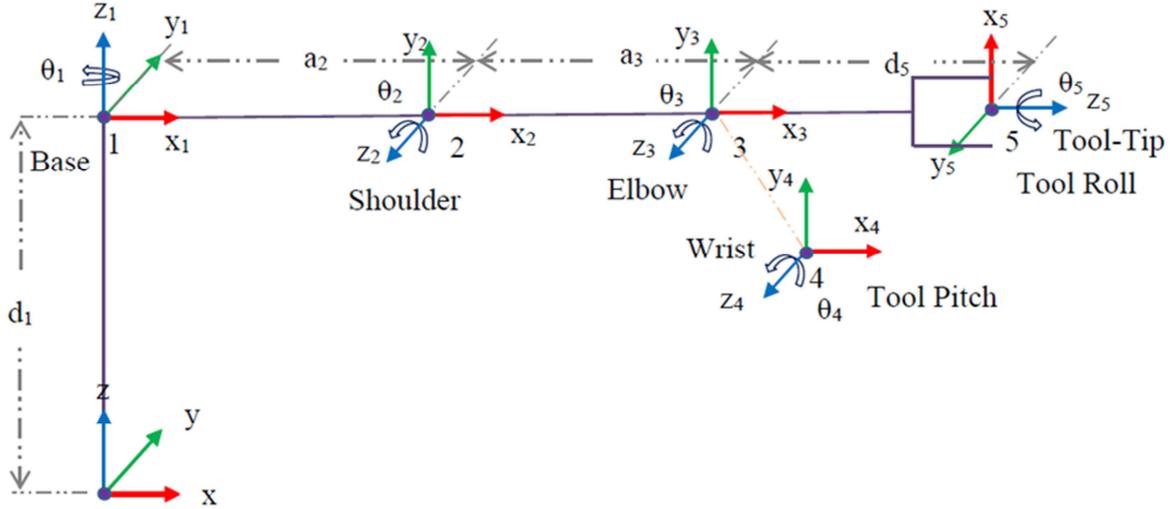


Figure 2. Link frame for the LabVolt R5150 robot.

2.1. Determining D-H Parameters

D-H parameters table is a notation developed by Denavit and Hartenberg, which is intended for the allocation of orthogonal coordinates for a pair of adjacent links in an open kinematic system. It is used in robotics, where a robot can be modelled as a number of related solids (segments) and the D-H parameters are used to define the relationship between the two adjacent segments.

Table 1. D-H parameters.

A	Link	d (m)	a (m)	α	θ	Joint Limit
1	Shoulder	0.2555	0	90°	θ_1	338° (-185°, 153°)
2	Elbow	0	0.19	0	θ_2	181° (-32°, 149°)
3	Wrist	0	0.19	0	θ_3	198° (-147°, 51°)
4	Tool Pitch	0	0	90°	θ_4	185° (-5°, 180°)
5	Tool Roll	0.115	0	0	θ_5	360° (-360°, 360°)

The first step in determining the D-H parameters is to locate links and then, the type of movement (rotation or translation) is determined for each link. As it can be seen in

Figure 1, the robot LabVolt R5150 has five rotational joints. Cranks, axes and rotation angles. They are shown as a simplified diagram in Figure 2. Using D-H parameters defined in the previous steps in Table 1, the robot model was created in MATLAB software using the Robotic Toolbox. Robot model in addition to previously determined D-H parameters contains physical parameters which is using in the calculation of the dynamics movement.

2.2. Forward Kinematic Model

The forward kinematic model represents the relations calculating the operational coordinates, giving the location of the end-effector, in terms of the joint coordinates. After establishing D-H coordinate system for each link as shown in Table 1, a homogeneous transformation matrix can easily be developed considering frame $\{i-1\}$ and frame $\{i\}$ transformation.

So, the link transformation matrix between coordinate frames $\{i-1\}$ and $\{i\}$ has the following form [3]:

$${}^{i-1}T_i = Rot(z, \theta_i).Trans(z, d_i).Trans(x, a_i).Rot(x, \alpha_i) = \begin{bmatrix} \cos \theta_i & -\sin \theta_i \cdot \cos \alpha_i & \sin \theta_i \cdot \sin \alpha_i & a_i \cdot \cos \theta_i \\ \sin \theta_i & \cos \theta_i \cdot \cos \alpha_i & -\cos \theta_i \cdot \sin \alpha_i & a_i \cdot \sin \theta_i \\ 0 & \sin \alpha_i & \cos \alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1}$$

The basic transformations and the product of these matrices are:

$${}^0T_1 = \begin{bmatrix} c_1 & 0 & s_1 & 0 \\ s_1 & 0 & -c_1 & 0 \\ 0 & 1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}, {}^1T_2 = \begin{bmatrix} c_2 & -s_2 & 0 & a_2 c_2 \\ s_2 & c_2 & 0 & a_2 s_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, {}^2T_3 = \begin{bmatrix} c_3 & -s_3 & 0 & a_3 c_3 \\ s_3 & c_3 & 0 & a_3 s_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, {}^3T_4 = \begin{bmatrix} c_4 & 0 & s_4 & 0 \\ s_4 & 0 & -c_4 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$${}^4T_5 = \begin{bmatrix} c_5 & -s_5 & 0 & 0 \\ s_5 & c_5 & 0 & 0 \\ 0 & 0 & 1 & d_5 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$${}^0T_5 = {}^0T_1 \cdot {}^1T_2 \cdot {}^2T_3 \cdot {}^3T_4 \cdot {}^4T_5 = \begin{bmatrix} c_1 c_{234} c_5 + s_1 s_5 & s_1 c_5 - c_1 c_{234} s_5 & c_1 s_{234} & c_1 (a_3 c_{23} + a_2 c_2 + d_5 s_{234}) \\ s_1 s_{234} c_5 - c_1 s_5 & -c_1 c_5 - s_1 c_{234} s_5 & s_1 s_{234} & s_1 (a_3 c_{23} + a_2 c_2 + d_5 s_{234}) \\ s_{234} c_5 & -s_{234} s_5 & -c_{234} & d_1 + a_3 s_{23} + a_2 s_2 - d_5 c_{234} \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2)$$

where,

$$s_{234} = \sin(\theta_2 + \theta_3 + \theta_4), c_{234} = \cos(\theta_2 + \theta_3 + \theta_4), s_j = \sin(\theta_j) \text{ and } c_j = \cos(\theta_j) \text{ for } j=1,2,3,4,5.$$

The observed robot has 3-DOF three links with a 2-DOF wrist mechanism. In this work, the location of the end of the first three links is tracked using one colored feature point. Since a coloured feature point is attached to the end of the first three links of the robot, the first three basic transformation matrices are calculated to get the location of the feature point. The final homogeneous transformation matrix for locating the feature point is got from the product of three basic transformation matrices:

$${}^0T_3 = {}^0T_1 \cdot {}^1T_2 \cdot {}^2T_3 = \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} Q^{3 \times 3} & P^{3 \times 1} \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3)$$

where,

$$Q = \begin{bmatrix} \cos(\theta_1) \cos(\theta_2 + \theta_3) & -\cos(\theta_1) \sin(\theta_2 + \theta_3) & \sin(\theta_1) \\ \sin(\theta_1) \cos(\theta_2 + \theta_3) & -\sin(\theta_1) \sin(\theta_2 + \theta_3) & -\cos(\theta_1) \\ \sin(\theta_2 + \theta_3) & \cos(\theta_2 + \theta_3) & 0 \end{bmatrix}$$

$$P = \begin{bmatrix} \cos(\theta_1) (a_3 \cos(\theta_2 + \theta_3) + a_2 \cos(\theta_2)) \\ \sin(\theta_1) (a_3 \cos(\theta_2 + \theta_3) + a_2 \cos(\theta_2)) \\ d_1 + a_3 \sin(\theta_2 + \theta_3) + a_2 \sin(\theta_2) \end{bmatrix}$$

From the position matrix P , the location of the feature point is calculated as along as the movement of robotic arm.

In this work, the forward kinematics of the robot is used to simulate and drive the robot for learning ANFIS networks and driving the robot to a specified trajectory or location.

3. Adaptive Neural-Fuzzy Inference System

Adaptive Neural-Fuzzy Inference System (ANFIS) is developed by R. Jang [1]. ANFIS is a hybridization of neural network and fuzzy logic methods. This is basically type of a feed forward neural network which involves fuzzy inference system through the structure of neural network and their neurons. It gives the learning ability of neural network to fuzzy inference system. The method is mainly developed for the evaluations of nonlinear functions that generally identifies nonlinear elements on line for control system design and predicts chaotic time series.

ANFIS structure is consists of five different layers such as fuzzy or input layer, normalization layer, product layer, defuzzification layer, and summation layer. Basic structure of the ANFIS is given in Figure 3, in which fixed node is given by circle and adjustable node is given by square. Suppose if there is two inputs x and y with one output z then ANFIS can be used as a first order Sugeno FIS. There are many fuzzy systems like Sugeno, Mamdani etc., but most popular and widely used system is Sugeno model due to its high interpretability and computational efficiency with default optimal and adaptive tools.

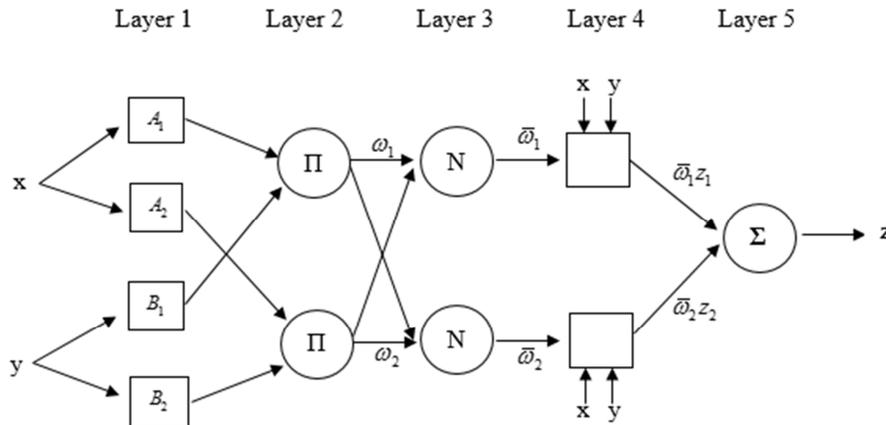


Figure 3. Architecture of ANFIS.

Therefore, first order Sugeno fuzzy rule can be expressed as follows:

First Rule: If x is A_1 and y is B_1 , then $Z_1 = p_1 x + q_1 y + r_1$

Second Rule: If x is A_2 and y is B_2 , then $Z_2 = p_2 x + q_2 y + r_2$ where, A_i and B_i are fuzzy sets; and p_i, q_i and r_i are parameters which is assigned during training process. As

presented in Figure 3, ANFIS structure consists all five layers.

i. Layer 1 (Input layer)

In this layer, each node is equal to a fuzzy set and output of a node in the respective fuzzy set is equal to the input variable membership grade. The parameters of each node determine the membership function in the fuzzy set of that node.

Now output node will be defined by

$$\begin{aligned} O_i^1 &= \mu_{A_i}(x), \quad i = 1, 2 \\ O_i^1 &= \mu_{B_i}(y), \quad i = 3, 4 \end{aligned} \tag{4}$$

where, $\mu_{A_i}(x)$ and $\mu_{B_i}(y)$ can hold any membership function (MF). For example, widely used membership function i.e. Gaussian MF is used throughout the work.

$$gaussmf(x, c, \sigma) = e^{-\frac{1}{2} \left(\frac{x-c}{\sigma} \right)^2} \tag{5}$$

where, x is the input value of the node; and ‘ c ’ and ‘ σ ’ determine the Gaussian membership function center and its width, respectively. Parameters in this layer are referred to as *premise parameters*.

ii. Layer 2 (Product layer)

The output of each node represents the weighting factor of rule or product of all incoming signals. In which μ_{A_i} is the membership grade of x in A_i fuzzy set and μ_{B_i} is the membership grade of y in fuzzy set B_i . Here AND (\cap) operator is used to product the input membership values.

$$O_i^2 = \omega_i = \mu_{A_i}(x) \times \mu_{B_i}(y), \quad i = 1, 2 \tag{6}$$

Each node output represents the firing strength of a rule.

iii. Layer 3 (Normalization layer)

Every node (circle) in this layer is a fixed node labelled as N . This layer is also called normalized layer. It calculates the ration of weight factor of the rule with total weight factor. In this layer, the average is calculated based on weights taken from fuzzy rules:

$$O_i^3 = \bar{\omega}_i = \frac{\omega_i}{\omega_1 + \omega_2}, \quad i = 1, 2 \tag{7}$$

where, $\bar{\omega}_i$ are normalized firing strengths.

iv. Layer 4 (Defuzzification Layer)

The output of every node is calculated by multiplying the normalized one with the consequent parameters (p_i, q_i, r_i) of the linear function. Every i^{th} node in the fourth layer is an adaptive node given by the following node function:

$$O_i^4 = \bar{\omega}_i z_i = \bar{\omega}_i (p_i x + q_i y + r_i), \quad i = 1, 2 \tag{8}$$

v. Layer 5: (Summation Layer)

The single node here is a fixed node, labelled as Σ , which compute the overall output as the summation of all incoming

signal. It can be expressed as follows:

$$O_i^5 = \sum \bar{\omega}_i z_i = \frac{\omega_1 z_1 + \omega_2 z_2}{\omega_1 + \omega_2} \tag{9}$$

4. Proposed System for ANFIS-Based Visual Positioning Approach

In this work, the visual positioning is trained using ANFIS as well as robotic forward kinematics and multi-view geometry. The flowchart of training process is shown in Figure 4.

In this research, the first step is to test the working space of the robotic arm in vision. In order to do so, a m-file have been created in MATLAB based on the direct kinematics of the robotic arm and the epipolar geometry of two cameras. In this work, two pin-hole cameras are used; installed above and on the lateral side of the robot, respectively. The cameras’ focal length and view limits are identical as 0.002 in m and [0, 1024, 0, 1024] in pixels, respectively.

The required data for ANFIS-learning is created in the robotic working space by varying the relative position between the robotic arm elements and acquiring image data from two cameras. The displacement between two consecutive elements was limited to their maximum and minimum ranges. This is so called as motor babbling phase, and the code in MATLAB is presented in Figure 5, and the simulation setup of robot and two cameras is shown in Figure 6. Figure 7 represents the maneuvering points of the robotic arm captured by two camera views during the motor babbling phase. In this research, Corke’s RVC v9.10 MATLAB toolbox [2] is used for the simulation of robot and two cameras.

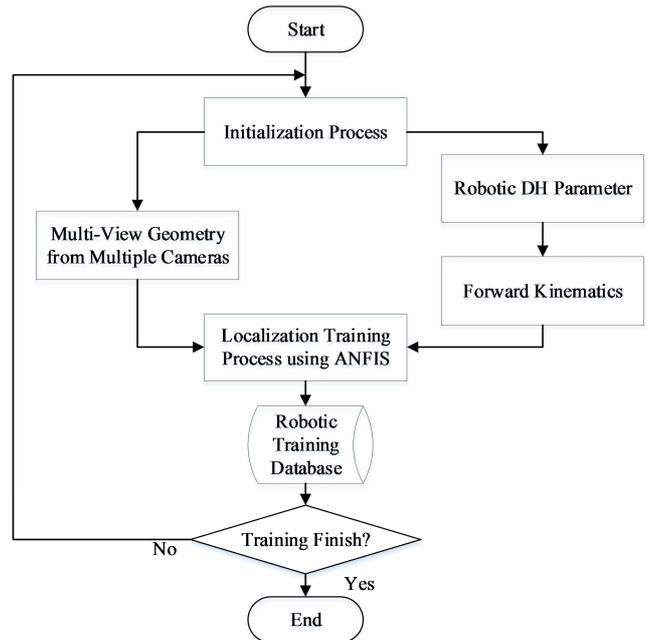


Figure 4. Flowchart of training process.

To get the required data, the manipulator was maneuvered within its workspace using forward kinematics, and the end-effector image coordinates were acquired from two cameras as shown in Figure 8. The image coordinates of these cameras are $[u_1, v_1]$ and $[u_2, v_2]$, and they are used as training data. Therefore, there are four inputs for ANFIS training.

```

for k_1 = th1_min:30*pi/180:th1_max
    for k_2 = th2_min:30*pi/180:th2_max
        for k_3 = th3_min:30*pi/180:th3_max
            m = m + 1;
            q = [k_1, k_2, k_3, 0, 0];
            T = LabVolt.fkine(q);
            P_3 = [T(1,4); T(2,4); T(3,4)];
            plot_sphere(P_3, 0.01, 'b');
            q_feature_pt = [q(1), q(2), q(3)];
            T_feature_pt = LV_FK_FP(q_feature_pt);
            P_fp = [T_feature_pt(1,4); T_feature_pt(2,4); T_feature_pt(3,4)];
            p1 = cam1.project(P_fp);
            UV_1(m,:) = p1;
            p2 = cam2.project(P_fp);
            UV_2(m,:) = p2;
            figure(1)
            hold on
            LabVolt.plot(q);
            view(24,24);
            hold on
            THETA(m,:) = q;
            WS(m,:) = P_3;
            FP(m,:) = P_fp;
        end
    end
end

```

Figure 5. Motor babbling code in MATLAB.

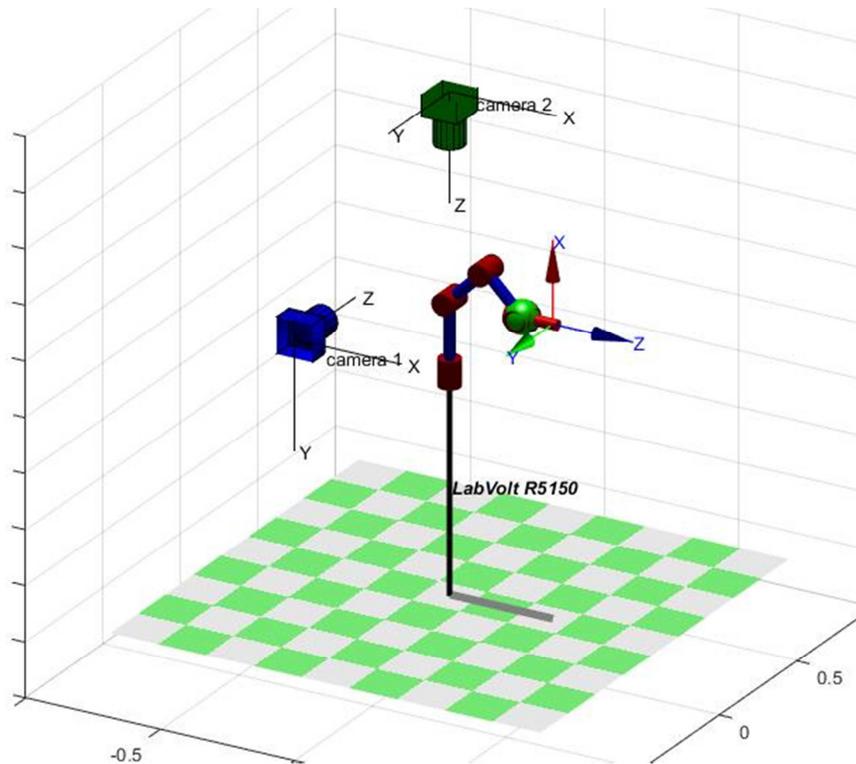


Figure 6. Simulation setup of robot and two cameras in MATLAB.

ANFIS network is trained with the Gaussian membership function with a hybrid learning algorithm. For the neuro-fuzzy model in this work, 588 data points analytically obtained using forward kinematics, of which 294 are used for training and the remaining 294 are used for validating.

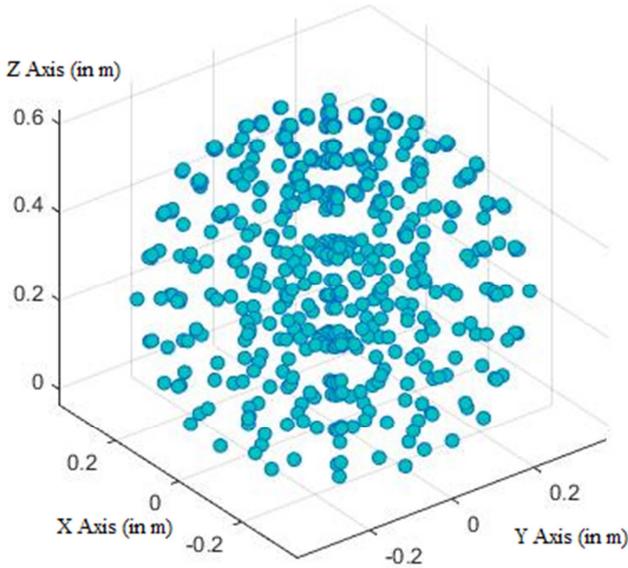


Figure 7. Workspace of the robot in two camera views.

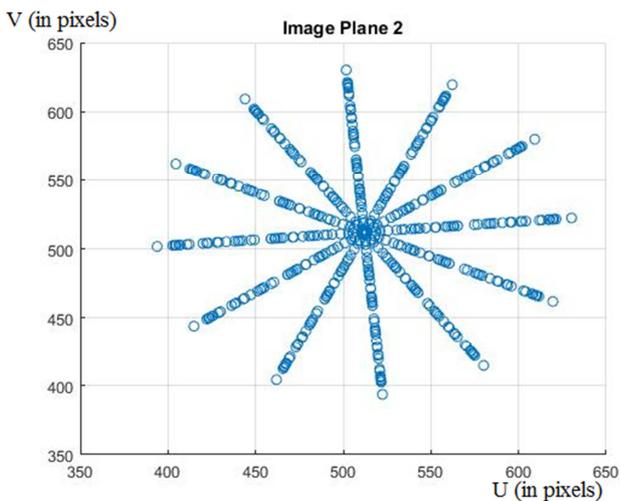
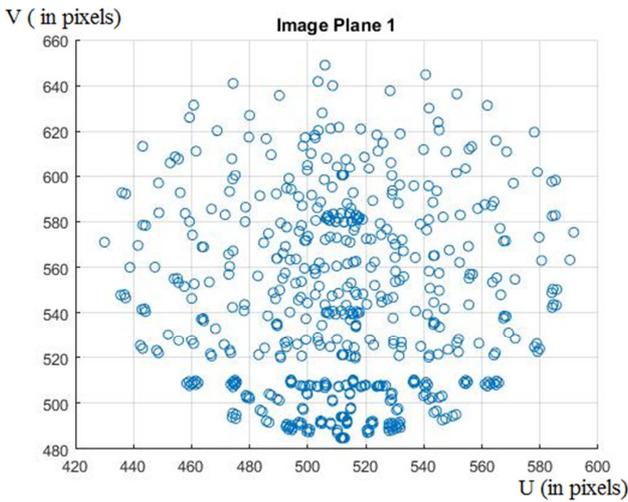


Figure 8. Image planes of lateral side and top cameras.

Since ANFIS is a judicious integration of FIS and ANN, it is capable of learning, high-level thinking and reasoning; and

combines the benefits of these two techniques into a single capsule. The success for FIS is the finding of the rule base. The reason being that there are no specific techniques for converting the knowledge of human beings into the rule base and also in order to maximize the performance of the model and to minimize the output error, further fine tuning of the membership functions is required. Thus, when generating a FIS using ANFIS, it is important to select proper parameters, including the number of membership functions (MFs) for each individual antecedent variable. It is also vital to select appropriate parameters for learning and refining process, including the initial step size (ss). In the present work, the commonly used rule extraction method applied for FIS identification and refinement is subtractive clustering. The MATLAB Fuzzy Logic Toolbox has been used for ANFIS model development. The flowchart of the ANFIS training in the work is shown in Figure 9.

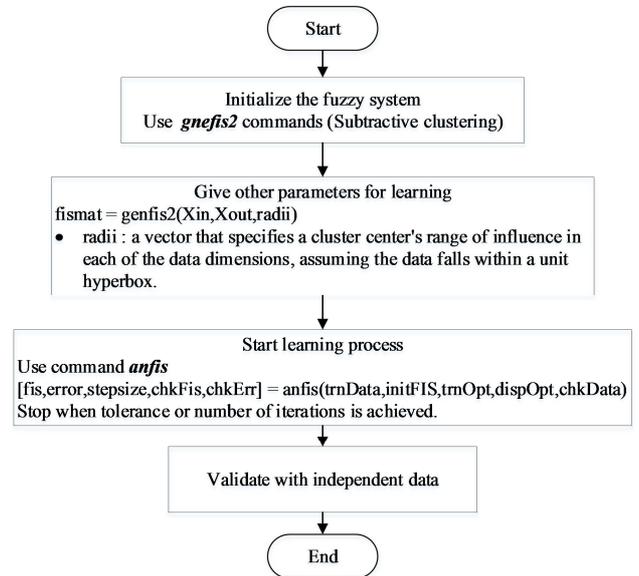


Figure 9. Flowchart of the ANFIS training.

Here the initial parameters of the ANFIS are identified using the subtractive clustering method. However, it is vital to properly define the subtractive clustering parameters, of which the clustering radius is the most important. It is determined through a trial and error approach. By varying the clustering radius r_a with varying step size, the optimal parameters are obtained by minimizing the root mean squared error (RMSE) based on the validation datasets. Clustering radius r_b is selected as $1.5 r_a$. Gaussian membership functions are used for each fuzzy set in the fuzzy system. The number of membership functions and fuzzy rules required for a particular ANFIS is determined through the subtractive clustering algorithm. Parameters of the Gaussian membership function are optimally determined using the hybrid learning algorithm. Each ANFIS is trained for 400 epochs.

Gaussian membership function has been used as the input membership function and linear membership function for the

output function. Here, separate sets of input and output data has been used as input arguments. In MATLAB, “genfis2” generates a Sugeno-type FIS structure using subtractive clustering. genfis2 is generally used where there is only one output; hence here it has been used to generate initial FIS for training the ANFIS. On the other hand, “genfis2” achieves this by extracting a set of rules that simulates the data values. In order to determine the number of rules and antecedent membership functions, “subclust” function has been used by the rule extraction methods. Further it uses the linear least squares estimation to determine each rule’s consequent

equations.

However, ANFIS itself is only suitable for single output system. For a system with multiple outputs, ANFIS will be placed side by side to produce a Multiple-output ANFIS (MANFIS) [1]. The number of ANFIS required depends on the number of required output. In this research, the cartesian coordinate points have to be outputted as ANFIS outputs. Figure 10 shows a MANFIS with three outputs; x , y and z . Since the input data remains the same for each ANFIS, they also have the same initial parameter such as initial step size, membership function (MF) type and number of MF.

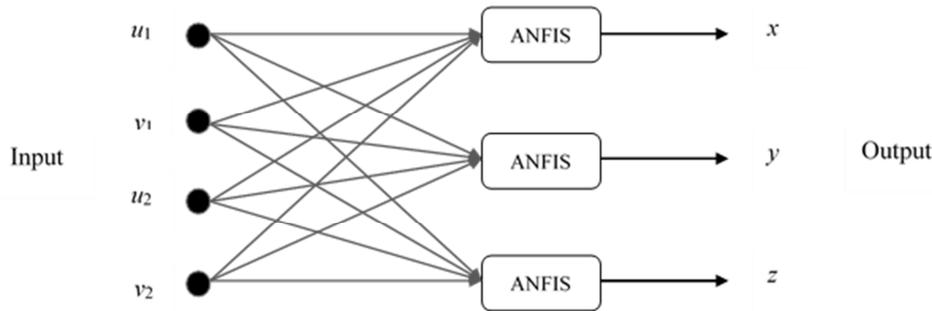


Figure 10. MANFIS with three outputs.

The parameters used in the model for training ANFIS are given in Table 2 and the rule extraction method used is given in Table 3. Table 4 summarizes the results of types and values of model parameters after training MANFIS.

Table 2. Parameters used in all the models for training ANFIS.

Parameters used in all the models	Subtractive clustering
Input MF type	Gaussian membership ('gaussmf')
Input partitioning	'subclust'
Output MF type	Linear
Number of output MFs	One
Training algorithm	Hybrid learning
Training epoch number	400
Initial step size	0.01

Table 3. Rule extraction method used for training ANFIS.

Rule extraction method used	Type
AND method	'prod'
Or method	'probor'
Implication method	'prod'
Aggregation method	'max'
Defuzzy method	'wtever'

Table 4. Results of types and values of parameters after training MANFIS.

Results of types and values of parameters	x	y	z
No. of nodes	87	77	117
No. of linear parameters	40	35	55
No. of non-linear parameters	64	56	88
Total no. of parameters	104	91	143
No. of training data pairs	294	294	294
No. of testing data pairs	294	294	294
No. of fuzzy rules	8	7	11

5. Simulation Tests and Results for Visual Pose Estimation

Three different ANFIS are designed for visual pose estimation of 5-DOF robot; x , y and z , respectively. The proposed method gives good estimation of the position of the 5-DOF robotic end-effector. A data set of 588 cartesian points analytically obtained using forward kinematics, and feature points captured by two cameras in the motor babbling phase is used for training and validation; 294 and 294, respectively.

After the training is complete, the model is validated using a different set of data from the one used before to train the FIS. In Figure 11-13, the rule viewers for x , y and z are presented. The rule viewer displays a roadmap of the whole fuzzy inference process. This represents a very useful tool for modifying and changing the fuzzy rules.

The validation of individual data set using ANFIS is done by calculating the difference between the cartesian coordinates deduced using robotic forward kinematics and the ones using ANFIS. A total of 294 observation points generated in the workspace for validating purpose are considered to find the error of the cartesian coordinates. The plot of the comparative results for deduced and predicted cartesian data is shown in Figure 14. Observing the results, the differences between FK-based and ANFIS-based data for individual Cartesian coordinate (X , Y , Z) are not much in 10^{-3} . Therefore, validating has a good estimation using the specified ANFIS models.

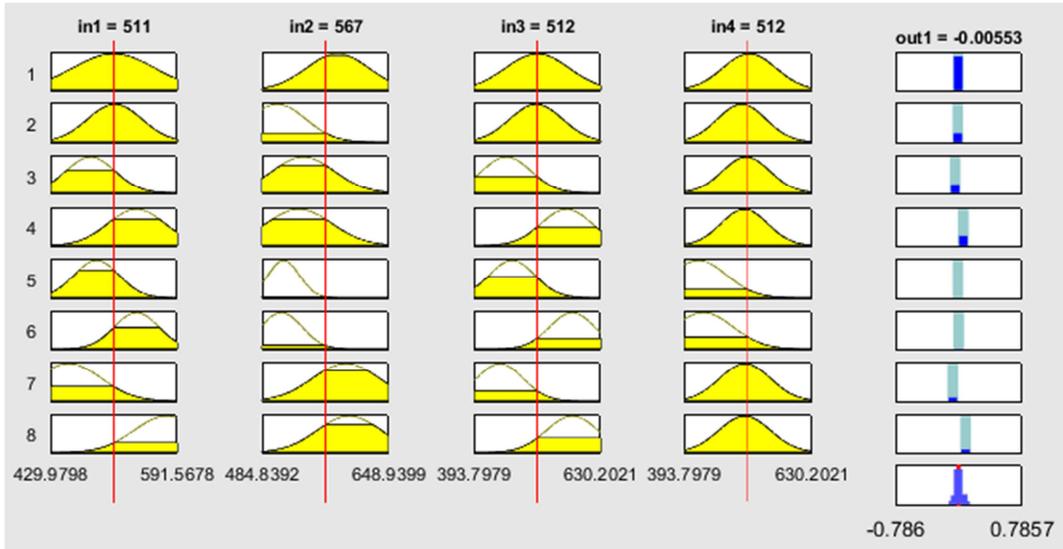


Figure 11. Rule Viewer for ANFIS x.

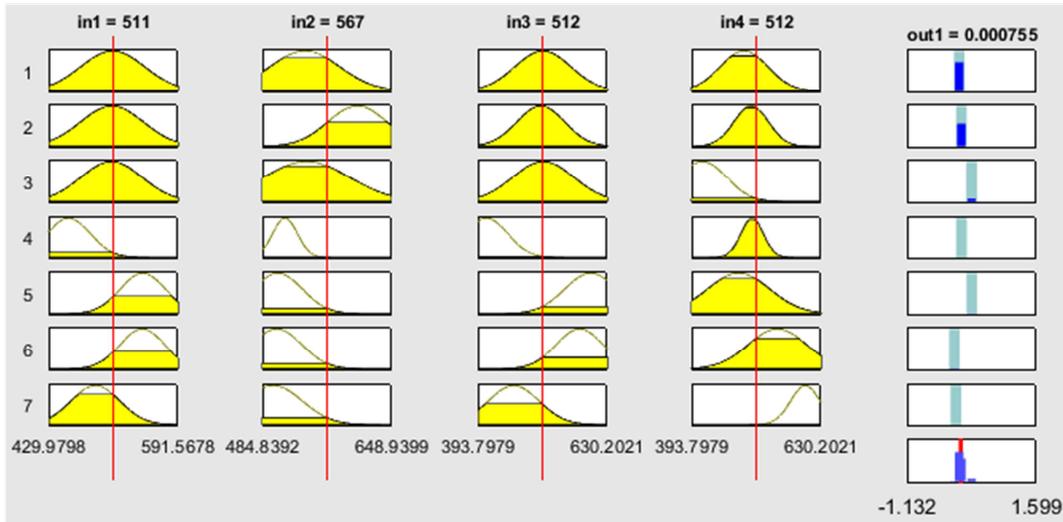


Figure 12. Rule Viewer for ANFIS y.

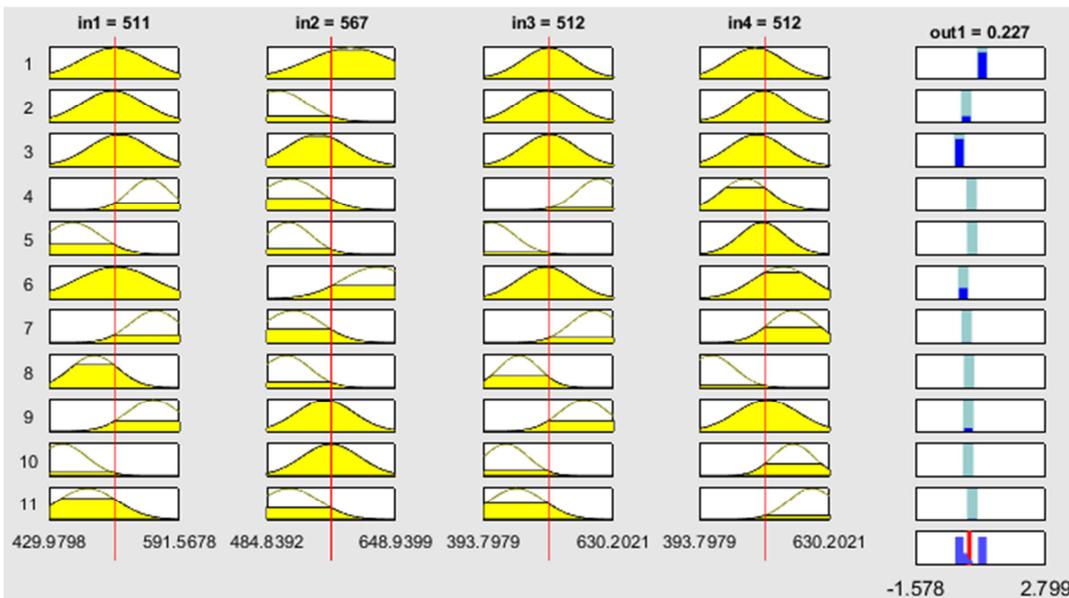


Figure 13. Rule Viewer for ANFIS z.

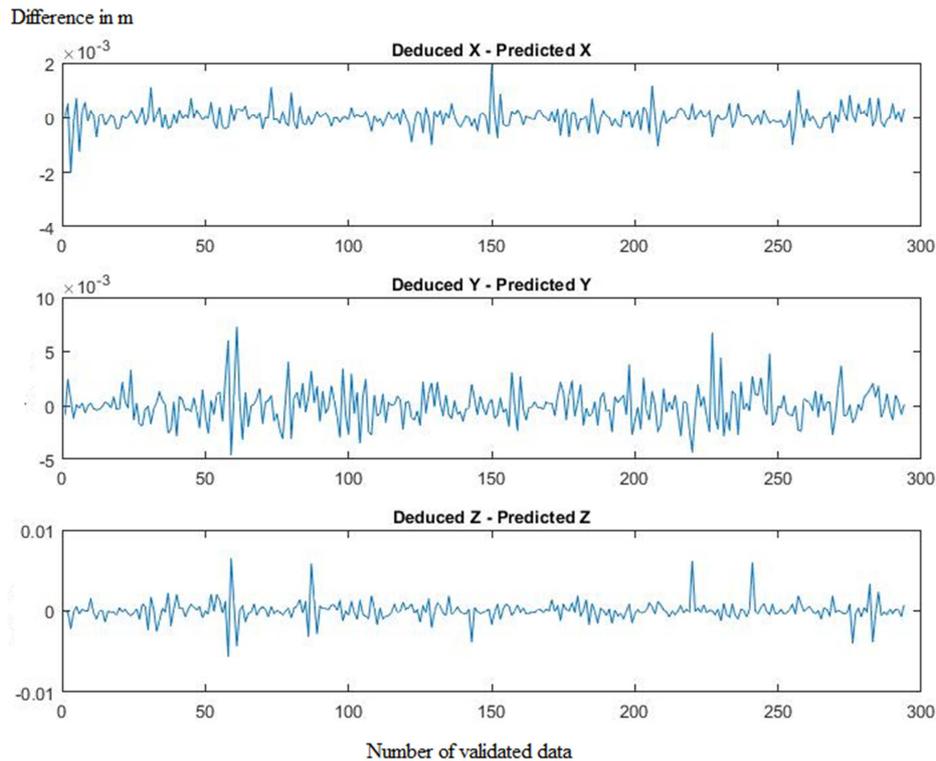


Figure 14. Comparative results for deduced and predicted cartesian data.

After testing the ANFIS networks, the MSE, RMSE, Error Mean and Standard Deviation (STD) Errors are calculated to check the estimation performance; described in Table 5. RMSE is a useful tool for comparing the forecasting errors. STD is one of the indicators that show the distribution of data on average how much the average value away. If the standard

deviation of the data set is close to zero, it means that the data are close to the average and dispersion are small, while large standard deviation indicates a significant distribution data. Observing the errors, it can be concluded that the proposed approach is efficient in estimating the location of the uncertain robotic arm.

Table 5. Errors in training and validating of ANFIS networks.

Data Errors	X		Y		Z	
	Train	Check	Train	Check	Train	Check
MSE	6.613×10^{-6}	7.789×10^{-6}	8.747×10^{-5}	8.942×10^{-5}	2.09×10^{-5}	2.471×10^{-5}
RMSE	0.0025715	0.0027908	0.0093527	0.0094563	0.0045716	0.004971
Error Mean	-7.293×10^{-17}	5.507×10^{-5}	-5.476×10^{-16}	-1.141×10^{-5}	-1.447×10^{-16}	5.924×10^{-6}
Error St. D.	0.0025759	0.002795	0.0093686	0.0094724	0.0045794	0.0049795

6. Conclusions

An ANFIS-based visual positioning approach using two cameras is proposed in this paper. The idea of using forward kinematic equations and two cameras for generating training data for ANFIS led to a nearly accurate training of the ANFIS network. Simulation experiments show that the location of the robotic arm can be trained in ANFIS using two uncalibrated cameras. Observing the errors, the estimated position of the robotic arm is efficient for visual feedback control. Further, the proposed ANFIS based approach is very useful for obtaining the position of the robotic arm in Cartesian coordinate system as it can work as a control algorithm. The Cartesian-coordinate-based learning can be used in robotic calibration, visual servoing and Cartesian controller. The authors are planning to use the pose

tracking using MANFIS and uncalibrated cameras for the visual servoing of the robot in future.

References

- [1] Jang, J. -S. R.; Sun, C. -T. and E. Mizutani. Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence. Prentice Hall. 1997
- [2] Peter Corke, Robotics, Vision and Control. Bruno S., Oussama K., Frans G., Ed. Belin, Germany. Springer Verlag. 2011.
- [3] J J. Denavit, R. S. Hartenberg, "A kinematics Notation For Lower-Pair Mechanisms Based On atrices", ASME Journal of Applied Mechanics, vol. 22, pp. 215-221, 1955.
- [4] C.-H. Huang, C.-S. Hsu, P.-C. Tsai, R.-J. Wang, and W.-J. Wang, "Vision Based 3-D Position Control for a Robot Arm," IEEE Control Systems, Nov. 2011, pp. 1699-1703.

- [5] E. Zhou, M. Zhu, A. Hong, G. Nejat and B. Benhabib, "Line-Of-Sight Based 3D Localization of Parallel Kinematic Mechanisms," *International Journal on Smart Sensing and Intelligent Systems*, vol. 8, no. 2, Jun. 2015, pp. 842-868.
- [6] S. Skaar, W. Brockman and W. Jang. "Three-dimensional camera space manipulation," *Int. J. Robotics Research*, Apr. 2009, pp. 1172-1183.
- [7] H. Hashimoto, T. Kubota, M. Kudou and F. Harashima. "Self-organizing visual servo system based on neural networks," *IEEE Control Systems*, Apr. 1992.
- [8] Gordon Wells, Christophe Venaille, Carme Torras, "Vision-based robot positioning using neural networks", *Image and Vision Computing* 14, Elsevier, 1996, pp. 715-732.
- [9] J.-Y. Baek and M.-C. Lee. "A study on detecting elevator entrance door using stereo vision in multi floor environment." in *Proc. ICROS-SICE Int. Joint Conf.*, Fukuoka, Japan, Aug. 2009, pp. 1370-1373.
- [10] K. Okada, M. Kojima, S. Tokutsu, T. Maki, Y. Mori, and M. Inaba. "Multi-cue 3D object recognition in knowledge-based vision-guided humanoid robot system," in *Proc. 2007 IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, CA, USA, Oct. 2007, pp. 3217-3222.
- [11] S. Winkelbach, S. Molkenstruck, and F. M. Wahl. "Low-cost laser range scanner and fast surface registration approach," in *Proc. DAGM Symp. Pattern Recognit.*, Berlin, Germany, Sep. 2006.
- [12] T. Dallej, M. Gouttefarde, N. Andreff, M. Michelin, and P. Martinet, "Towards vision-based control of cable-driven parallel robots", *IEEE International Conference Intelligent Robots and Systems, IROS'11.*, Sep. 2011, San Francisco, United States. sur CD ROM, pp. 2855-2860.
- [13] C. Torras. "Neural learning for robot control," *Proc. 11th Euro. Conf. on Artificial Intelligence (ECAI '94)*, Amsterdam, Netherlands, August 1994, pp. 814-819.
- [14] J. Cid and F. Reyes, "Visual Servoing Controller for Robot Manipulators," *4th WSEAS International Conference on Mathematical Biology and Ecology (MABE'08)*, Acapulco, Mexico, Jan. 2008, pp. 25-27.