Visual Servoing Application for Inverse Kinematics of Robotic Arm Using Artificial Neural Networks

Ayca AK [1]*, Vedat TOPUZ[1], Emregul ERSAN[2]

[1] Marmara University Vocational School of Technical Sciences, Goztepe, Istanbul/Turkey
aycaak@marmara.edu.tr (**Corresponding author*)

[1] Marmara University Vocational School of Technical Sciences, Goztepe, Istanbul/Turkey
vtopuz@marmara.edu.tr

[3] Zeytinburnu Industrial Vocational School, Zeytinburnu, Istanbul/Turkey
emregulersan@hotmail.com

Abstract: This paper presents novel approach for a visual servoing application of six axis robotic arm. Basic image-processing techniques were used for object recognition and position determination of robotic arm. The inverse kinematics solution of the robot arm was performed with artificial neural networks. Afterwards the robot's inverse kinematics solution was completed, the determined joint-angle values were used to control the robot arm. Performance of radial basis function network (RBF) and multilayer perceptron (MLP) were also compared.

Keywords: Image processing, Robot control, Artificial neural network, Visual servoing, Inverse kinematic.

1. Introduction

Robot visual servo with one or more CCD cameras has received increasing attention recently. This field spans several disciplines including robotics, computer vision, and control theory. Vision servoing have been inserted to increase the accuracy and flexibility of control systems. Generally, vision can be used to control different dynamic systems, e.g., submarines, aircrafts, and vehicles. The purpose of the visual servoing approach is to control a system by using the information providing a vision system [12]. Vision systems used for robotic applications can typically categorize according to the function of vision sensors' number.

A camera is used for monocular visual servoing, either attached to a stable plane pointing toward the robotic workspace (fixed camera configuration) or assembled at the end effector of the robot (eye-in-hand configuration). Multicamera vision systems, in which multiple cameras are placed in the workspace, are used to obtain the mission-specific data [24].

Robotic image-processing techniques have been extensively studied. Knoeppel et al. determined, with two CMOS cameras mounted on a car's rear window, whether a vehicle was following the car at a maximum distance of 150 m away and between other vehicles [19].

Mondi et al. developed a ping-pong-playing robot arm. The coordinates of ping-pong balls were determined with image-processing techniques applied to real-time images taken with a CCD video camera [21].

Mundhra et al. implemented a mobile robot catching a ball that had fallen [22]. Choi, Ryu, and Kim realized a mobile robot moving without hitting the wall with sensors in a confined space, such as a maze [8]. Bustamante and Gu located an electrical plug and its coordinates with a camera attached to the robot arm by using pattern-recognition algorithms [6]. Claudio et.al. realized an application using two visual-servoing on the humanoid robot [9].

Artificial neural networks (ANN) are also used for visual servoing applications and inverse kinematics solutions. Al-Junaid realized ANN based control of a nonlinear system with visual servoing. ANN was used to control a 6-axis robot arm in his study [2]. Feng et.al. used extreme learning machine to obtain PUMA 560 robot joint angles. Feedforward neural networks used in their study had a single hidden layer [13]. Raptis & Tzafestas obtained inverse kinematics of the PUMA 3R manipulator via Neurofuzzy and Neural Networks [23]. Almusawi et.al., proposed a multilayer neural network with 6 input variables to solve inverse kinematics of Denso VP6242 robotic arm [3]. Jha & Biswal compared the ANN and ANFIS results of the inverse kinematic solution of 5R Manipulator [17]. Srisuk et.at., found inverse kinematics solution of the robotic arm with 3 DOF an ANN in MLP structure in 3D space [25].

https://doi.org/10.24846/v27i2y201806

Sliding Mode Control (SMC) is also investigated to control the robotic manipulators in the presence of uncertainties, unknown external load disturbances [1]. Kara & Mary also presented an adaptive and robust control scheme, which is based on with proportional derivative control for trajectory tracking of nonlinear robotic manipulators [18].

In this study, an image-processing interface has been prepared for image acquisition, image processing, object coordinate finding, forward kinematics, inverse kinematics, simulation, and running functions. Monocular visual-servoing method is used to detect an object in the workspace.

The process comprises four main stages: object recognition, determination of the object's location, inverse kinematics solution of the robot arm, and movement of the robot arm. The image of the object is transferred to the system by a camera placed at the top of the workspace. After the captured image is passed through some image-processing stages, it is determined whether the object is the desired one. Then, robotic arm inverse kinematics solution is realized using ANNs. Finally, objects are moved to the designated warehouse area using a six-axis educational robot arm.

This paper is organized as follows: Section 2 presents the forward and inverse kinematics analysis of robotic arm. The proposed ANN is described in Section 3. Section 4 illustrates the image processing applications. Section 5 presents the experimental work and results. Finally, Section 6 concludes this paper.

2. Visually Guided Robot Arm

At the designed application, objects found in the workspace are firstly identified through image-processing techniques. Then, the inverse kinematics solution of the robot is computed. Finally, the object is moved to the desired location by the robotic arm. A block diagram of the designed system is shown in Figure 1.

2.1 Forward Kinematic

In the forward kinematics problem, the end-effector's location in the Cartesian space. Conversely, given a desired end-effecter position and orientation, the inverse kinematics problem refers to finding the values of the joint variables that allow the manipulator to reach the given location. Denavit-Hartenberg (D-H) approach is used to determine the forward kinematics of serial robotic arm. In this approach, the D-H parameters are given using four parameters. These parameters completely specify the configuration of the frame (i) system relative to the frame $(i-1)$ system [11].

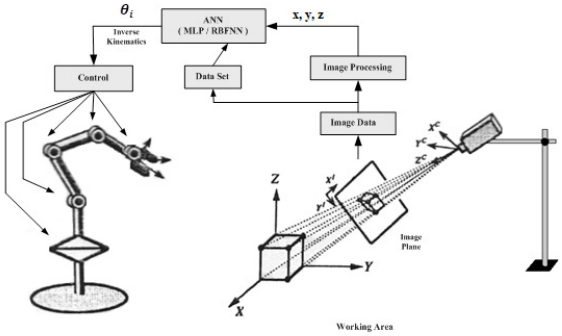


Figure 1. Designed Visual Servoing Application Block Diagram

For any particular link (i), both rotation (α_i) and displacement (a_i) are constant quantities solely determined by particular kinematic configuration of robotic arm. Furthermore, the two remaining parameters vary: rotation of axis (θ_i) if joint (i) is revolute, or offset (d_i) if joint (i) is prismatic. The complete configuration coordinate transformation matrix $({}^{i-1}_{i}T)$ associated with these four operations can readily be determined by composition of the four transformation matrices associated with each individual operation.

$$ {}^{i-1}_{i}T = R_{x(\alpha_{i-1})} D_{x(a_{i-1})} R_{z(\theta_i)} D_{z(d_i)} \tag{1} $$

where R is a rotation and D is displacement matrices about axis x and z respectively. This equation can be clearly expressed as follows:

$$ {}^{i-1}_{i}T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & cos\alpha_{i-1} & -\sin\alpha_{i-1} & 0 \\ 0 & \sin\alpha_{i-1} & cos\alpha_{i-1} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & a_{i-1} \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 & 0 \\ \sin\theta_i & \cos\theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2} $$

As a result, the general transformation matrix is obtained as Eq. 3.

The orientation of the end effector is represented in the first three columns of the matrices, whereas the position of the end effector is in the last column. In this way, the D-H parameters can constitute to calculate the forward kinematics of the robot.

$$ {}^{i-1}_{i}T = \begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 & a_{i-1} \\ \sin\theta_i * cos\alpha_{i-1} & \cos\theta_i * cos\alpha_{i-1} & -\sin\alpha_{i-1} & -\sin\alpha_{i-1} * d_i \\ \sin\theta_i * \sin\alpha_{i-1} & \cos\theta_i * \sin\alpha_{i-1} & cos\alpha_{i-1} & \cos\alpha_{i-1} * d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3} $$

Used robotic arm and their rotation axes and links are given in Figure 2. While the first 5 joints are the basic joints that determine the position of the end effector, the position of the sixth joint is related to the on/off state of the end effector and is not included in the kinematic calculations. The rotation angle limit values of each joint are given in Table 1.

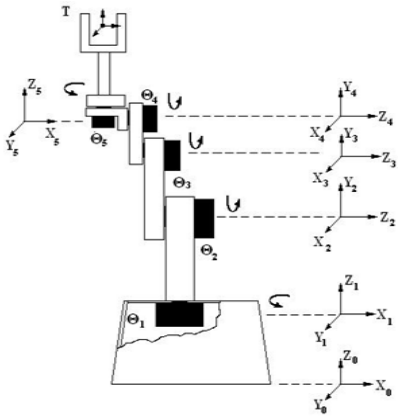


Figure 2. Rotation axes of robots

Table 1. Robot Angle Limit Values

Rotation Axes (degree)	θ_1	θ_2	θ_3	θ_4	θ_5
Maximum	100	83	115	120	80
Minimum	-95	-110	-100	-100	-80

In order to calculate the forward kinematic matrix of the robot arm, the limit values of the rotation angles, the offset and displacement values of each link were measured. Obtained D-H parameters for the robotic manipulator are listed in Table 2.

Table 2. D-H Parameters of Robot

Link (i)	1	2	3	4	5
Rotation (°) (α_{i-1})	0	-90	0	0	90
Displacement (mm) (a_{i-1})	0	0	90	70	110
Rotation (°) (θ_i)	θ_1	θ_2	θ_3	θ_4	θ_5
Offset (mm) (d_i)	200	12	12	30	0

If the D-H parameters are substituted in the translation matrix in Eq. 3, the translation matrices for each axes (Eq. 4) and overall forward kinematics matrices are obtained (Eq.5).

$$ {}^{0}_{1}T=\begin{bmatrix}\cos\theta_1 & \sin\theta_1 & 0 & 0\\ -\sin\theta_1 & \cos\theta_1 & 0 & 0\\ 0 & 0 & 1 & h_1\\ 0 & 0 & 0 & 1\end{bmatrix},{}^{1}_{2}T=\begin{bmatrix}\cos\theta_2 & 0 & \sin\theta_2 & 0\\ -\sin\theta_2 & 0 & \cos\theta_2 & 0\\ 0 & -1 & 0 & 0\\ 0 & 0 & 0 & 1\end{bmatrix},{}^{2}_{3}T=\begin{bmatrix}\cos\theta_3 & \sin\theta_3 & 0 & -h_2\\ -\sin\theta_3 & \cos\theta_3 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix}, $$
$$ {}^{3}_{4}T=\begin{bmatrix}\cos\theta_4 & \sin\theta_4 & 0 & -h_3*\cos\theta_4\\ -\sin\theta_4 & \cos\theta_4 & 0 & h_3*\sin\theta_4\\ 0 & 0 & 1 & h_5\\ 0 & 0 & 0 & 1\end{bmatrix},{}^{4}_{5}T=\begin{bmatrix}\cos\theta_5 & 0 & \sin\theta_5 & -h_4*\cos\theta_5\\ -\sin\theta_5 & 0 & \cos\theta_5 & h_4*\sin\theta_5\\ 0 & -1 & 0 & 0\\ 0 & 0 & 0 & 1\end{bmatrix} \quad (4) $$

$$ {}^{0}_{5}T={}^{0}_{1}T*{}^{1}_{2}T*{}^{2}_{3}T*{}^{3}_{4}T*{}^{4}_{5}T=\begin{bmatrix}0 & -1 & 0 & 0\\ 0 & 0 & -1 & h5\\ 0 & 0 & 1 & (h1+h2+h3+h4)\\ 0 & 0 & 0 & 1\end{bmatrix} \quad (5) $$

2.2 Inverse Kinematics

The inverse kinematics can state as to find the evaluating of the joints' variables that will realize the required position and orientation for a requisite location of the arm's limit. It can be expressed as follows:

$$ \theta_i = f(\mathrm{x},\mathrm{y},\mathrm{z}) \quad (6) $$

where $(\mathrm{x},\mathrm{y},\mathrm{z})$ represent the position at the Cartesian coordinate system and (θ_i) are joint angles.

The inverse kinematic solution could be realized using three methods; complete analytical solution (closed form), semi-analytical solutions and numerical solutions. There may not be adequate closed form solutions when coupling of the orientation and position kinematics happens. Using ANN's to solve the inverse kinematics problems in this case is better [26].

3. ANN Model for Inverse Kinematics

Because of above statement, ANNs are a widespread approach used for the inverse kinematic solution of the robot arm. Given a sample vector, ANN is capable to map the connection between inputs and outputs. When the ANN is trained, it can generalize related output for an input set data. In the learning process, training algorithm iteratively modifies the connection weights.

In this study, this nonlinear modeling feature of ANN was used. The inputs values are robot joint angles which were generated between robot axes limit angles. The output values are robot Cartesian coordinate values. This data set was inversely used for the inverse kinematic solution of the robotic arm with ANN. MLP and RBF models were proposed to measure the performance of the ANN.

3.1 Multi-Layer Perceptron (MLP)

Typically, MLP network is formed in neurons layers. Here each neuron in the layer calculates the sum of its input data $\mathbf{x}=[\theta_1,\theta_2,\theta_3,\theta_4,\theta_5]$. Then, this sum is applied to an activation function (f). The output of the network $\mathbf{o}=[x,\ y,\ z]$ is defined as a matrix form;

$$\mathbf{o}=f^2(\mathbf{W}^2 f^1(\mathbf{W}^1\mathbf{x}+\mathbf{b}^1)+\mathbf{b}^2) \tag{7}$$

where (W) are weight matrices, (b) are bias vectors and (f) are the sigmoid and linear activation functions.

MLP network with one hidden layer used in this study is shown in Figure 3. Backpropagation algorithm is used to update the weights of the MLP network [16]. The following equation is used to minimize the mean square error for adjusting the weights in this algorithm;

$$e=\frac{1}{2}\sum_{\gamma=1}^{p}(t^{\gamma}-o^{\gamma})^2 \tag{8}$$

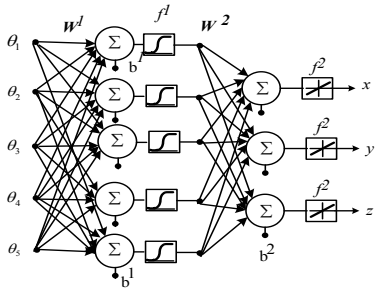


Figure 3. Realized MLP network structure

Network error is decreased iteratively by the steepest descent algorithm:

$$w^m(k+1)=w(k)-\cdot\frac{\partial e}{\partial w}$$
$$b^m(k+1)=b(k)-\cdot\frac{\partial e}{\partial b} \tag{9}$$

where; ***o*** is output, ***γ*** is the sampling instant in ***q*** size and ***t*** is a target and ***η*** is learning rate.

3.2 Radial Basis Network (RBF)

RBF is a feed-forward neural network comprising two layers. These are a nonlinear hidden layer and a linear output layer (Figure 4). The hidden layer activation function used in this study is Gaussian kernel (Ψ) function.

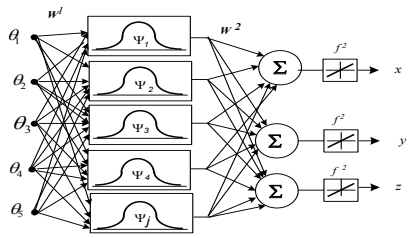


Figure 4. Realized RBF structure

The output layer performs the linear combination of the basic function responses expressed as follows:

$$\mathbf{o}=b+\sum_{j=1}^{q}w_{i,j}\Psi_j \tag{10}$$

where q is the model dimension and Ψ_j is the kernel function of the *jth* hidden neuron defined as;

$$\Psi j=\exp\left[-\frac{\|\mathbf{x}-c_j\|^2}{2\sigma_j{}^2}\right] \tag{11}$$

RBF training procedure has two phases. In the first phase, the input data set is used to define the center positions (c_j). For this purpose, unsupervised clustering method such as the K-means algorithm is used. The radius (σ_j) is determined by the k-nearest neighbor precept. Output layer's weights (W) is updated in the second phase while keeping the (c_j) and (σ_j) which are fixed [15].

3.3 Inverse Kinematic Solution with ANN

For each axis of the robot arm, 800 inputs $(\theta_1,\theta_2,\theta_3,\theta_4,\theta_5)$ were generated randomly within the rotation axis limit values given in Table 1. The dataset output values $(x,\ y,\ z)$ are calculated by using D-H parameters of the robotic arm as a given in Eq. 5. Among this created datasets, 600 of them were randomly selected for training and the rest of them used for the testing procedure. This dataset was inversely used the inverse kinematic solution of the robot arm using ANN. That means ANN input values are the Cartesian coordinate matrix $(x,\ y,\ z)$ and the targets are rotation angles $(\theta_1,\theta_2,\theta_3,\theta_4,\theta_5)$. This configuration is shown in Figure 5. After ANN training, test procedures were carried out and to show the MLP and RBF networks performance.

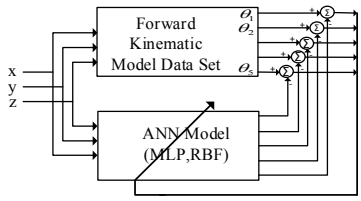


Figure 5. The inverse kinematic solution with ANN

4. Image Processing

To create a digital image, the perceived continuous data must be converted to a digital format. The digitization process includes two processes: sampling and quantization [14]. The resolution of the digital image is determined through the sampling process (i.e.,1024 × 768) and the color depth of the image is determined through quantization. Initially, camera calibration process must be used to correct the convex view obtained from the camera. Then, color images must be converted to black and white image so that the object searched is clearly visible. Finally, desired object Cartesian coordinate values must be obtained. In this study, it is not necessary to find the z coordinate value (height) because the white table tennis ball was used as the desired object to be found.

4.1 Camera Calibration

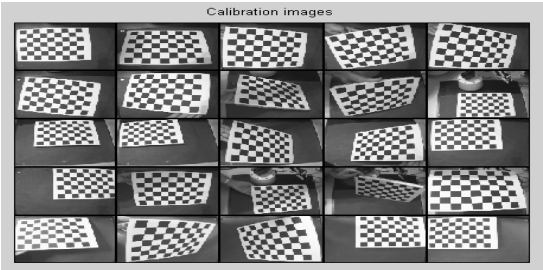


Figure 6a. Calibration panel images in the different distances and angles

In this study, the obtained images are dished (convex) because the camera uses a wide-angle lens. Elimination of this convex corruption and converting the picture to plane form are necessary to avoid wrong measurements of the object coordinates within the workspace. For this procedure, MATLAB Camera Calibration Toolbox software is used [6]. The calibration page in view of a checkerboard consists of 2.8 × 2.8 cm squares, with 7 pieces on the vertical and 9 pieces on the horizontal axis. Using twenty-five images captured at different angles and distances in front of the camera pictures and their obtained direction and position are given in Figure 6a and 6b respectively.

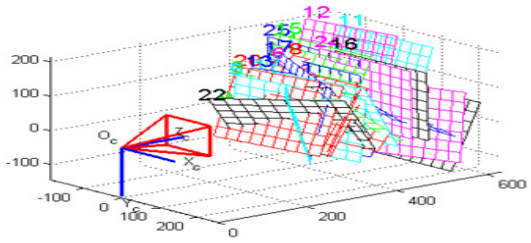
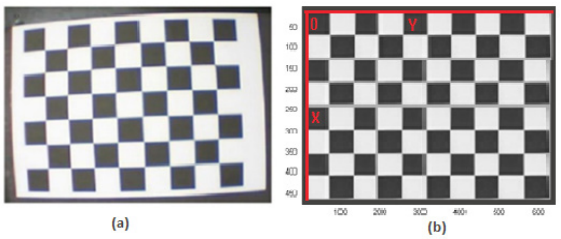


Figure 6b. Calculated directions and camera-centered positions of the calibrated images

After the calibration process, all images obtained from the camera can be easily corrected with the software. After the calibration process, the raw image of the calibration worksheet is brought into the image plane to enable common and metric readings. An image of the calibration board before and after the calibration process is given in Figure 7(a) and 7(b), respectively.



(a) Before (b**)** after the calibration process

Figure 7. Calibration board image

4.2 Image Processing

Conversion of the received image to a binary image and thresholding operations are carried out in this part. Initially, the global threshold value is determined (Figure 8a) after obtaining the threshold value of the image. Then, a new threshold value is manually selected as shown in (Figure 8b).

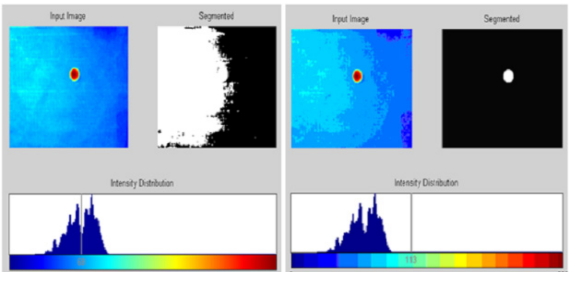


Figure 8. Threshold. (a) with the global threshold value and (b) with a new threshold value

4.3 Object Cartesian Coordinate Values

The object recognition process consists of two phases; recognition of object shape and center point of recognized object. The total number of pixels is used as the basis for defining the object in the work done. Since a fixed camera system is used, the total pixel number of the object will not change for each case [4]. Total pixel of white areas is found as follows;

$$A_k = \sum_{x=1}^{n}\sum_{y}^{m} B(\mathrm{x},\mathrm{y}) \text{ , where } B(x,y) = \begin{cases} 1 & if\ B(x,y) \in k \\ 0 & if\ B(x,y) \notin k \end{cases} \quad (12)$$

where A_k is the field of the object and $B(x, y)$ represents the x column and the y line value of a labeled picture that has m rows and n columns.

The position of the object is determined by computing the center of mass. The x and y coordinates of the center of mass are detected with Eq. (14).

$$x = \frac{\sum_{i=1}^{n}\sum_{j=1}^{m} i * B(i,j)}{A_k},\ y = \frac{\sum_{i=1}^{n}\sum_{j=1}^{m} j * B(i,j)}{A_k} \quad (13)$$

After determining the center of mass, it is determined whether the coordinate values are inside the workspace area. Values obtained in the previous section are presented in the coordinate plane in this part of the interface given in Figure 2.

4.4 User Interface

The interface and sub functions were developed under Matlab. The main interface of the application designed for visual servoing is shown in Figure 9. In image acquiring section, the image is taken from the selected camera and the camera can be calibrated. In image processing section, the color picture is converted to the black and white picture at the desired threshold value. The coordinate finding of the object evaluates whether the object on the screen is the wanted object, and when the object sought is found, the coordinate values of the object's center point in cm are calculated. In the image acquiring process, the camera calibration toolbox, the robot forward kinematic, and the robotic toolbox [10] for the simulation functions and the ANN toolbox for the inverse kinematic operation were used.

5. Experimental Results

A well-trained MLP and RBF networks structure could predict the output for any input data from the input space. The correlation coefficients (R) are used to compare the performance of the networks trained after the test procedure. The correlation coefficient is a dimensionless measure of the degree of a linear association of two random variables, with value in the interval [−1, 1]. 0 means no linear association, (1) means linear association and (−1) is a linear association with opposite directions.

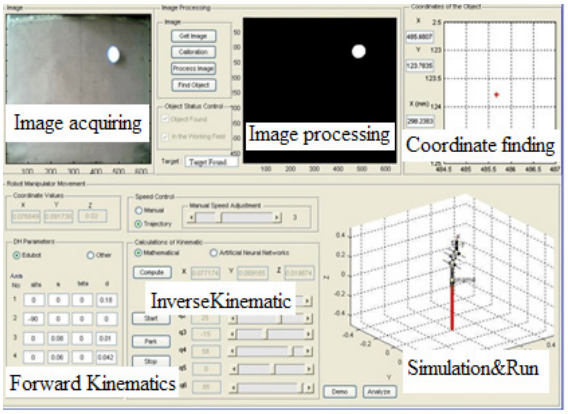


Figure 9. The user interfaces main screen

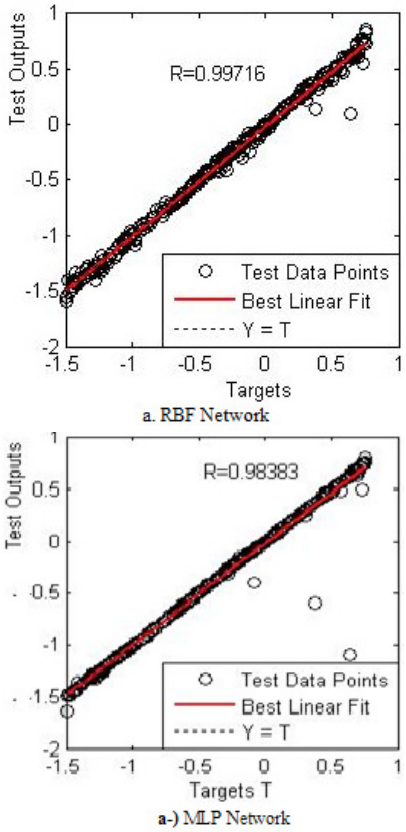


Figure 10. MLP and RBF networks scattering diagrams and correlation coefficients (R)

Scattering diagrams showing the performances of MLP and RBF networks are given in Figure 10. As can be seen, the RBF network's correlation coefficient (R = 0.997) is slightly better than the MLP network's (R = 0.983). This means that both structures can be used to solve the inverse

kinematic problems and there is no obvious difference between them.

To examine this proposal in detail, the results obtained from the MLP and RBF network tests are given in Figures 11 and 12, respectively, in comparison with the forward kinematic input values of the robot. Only 80 datasets were given to show the figure clearly and. In these figures, rotation angles are given in radian.

In order to test the obtained results in the real-time, 10 test points were randomly produced in the x, y Cartesian coordinate plane within the robot working area. The work area is in the size of 200 x 300 mm and the axis origin point is determined as the upper left corner.

The rotation angles calculated by the ANN models for these test points and the locations of the robot arm end points for these angles are given in Table 3. As seen in the table, the error value in the results obtained from both ANN models is within the limits of ±4 mm. Also, the error performance of the RBF network structure is better than the MLP structure.

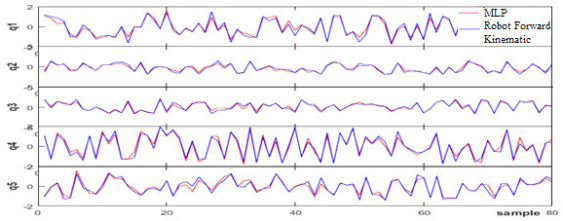


Figure 11. Robot rotation angles for forward kinematic model and MLP network

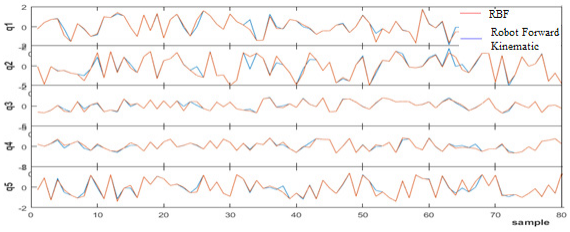


Figure 12. Robot rotation angles for forward kinematic model and RBF Network

6. Conclusion

In this study, visual servoing application of a robot arm whose inverse kinematic solution realized with ANN was performed.

Firstly, distortion caused by the convex camera lens was eliminated. Thus, it is provided to give real distance values on the two-dimensional Cartesian coordinate system of the image taken from the camera. The identity and midpoint of the object were determined by developing an object identification algorithm based on the total number of pixels on the image. Object-detection algorithm was also tested on different types of objects. The algorithm does not allow the robot arm to move when an unrecognized object enters the workspace.

The MLP and RBF models were used to obtain inverse kinematic solution based on the forward kinematic solution. According to the results obtained from the testing of RBF and MLP network structures, both network structures can be used in robot inverse kinematics calculation even if the RBF network structure performs slightly better. According to the results of RBF and MLP network structures, both network structures can be used in robot inverse kinematics calculation even if the RBF network structure performs slightly better. The designed system was tested in real time and similar results were achieved.

According to all these results, it was concluded that the system would be an alternative to a real-time visual servoing application.

Table 3. Real time robot rotational angles and obtained cartesian coordinate values

ANN Model	Obtained Rotation Angles (degree)				Test Points (mm)		Robot end-effector point (mm)		Error	
	θ_1	θ_2	θ_3	θ_4	X	Y	X	Y	X	Y
MLP	-1	20	-10	67	87	51	90	48	-3	3
RBF	0	22	-13	68			85	52	2	-1
MLP	-62	12	-14	74	42	127	38	130	4	-3
RBF	-80	17	-6	79			39	123	3	4
MLP	-55	19	-16	76	75	88	77	99	-2	-11
RBF	-55	23	-15	75			74	93	1	-5
MLP	24	21	5	40	80	125	81	104	-1	21
RBF	27	24	-5	36			80	126	0	-1
MLP	11	15	-1	77	55	75	68	79	-13	-4
RBF	9	19	-8	75			55	74	0	1
MLP	-9	15	-17	90	32	32	73	29	-41	3
RBF	-15	11	11	81			4	23	28	9
MLP	-12	12	-12	84	59	3	85	4	-26	-1
RBF	-20	17	-3	80			62	4	-26	-1
MLP	-12	12	-12	84	59	3	85	4	-26	-1
RBF	-20	17	-3	80			62	4	-3	-1
MLP	-12	12	-12	84	59	3	85	4	-26	-1
RBF	-20	17	-3	80			62	4	-26	-1
MLP	-34	10	-17	90	36	43	87	42	-51	1
RBF	-52	14	10	81			48	47	-12	-4

REFERENCES

1. Ahmed, S., Wang, H. & Tian Y. (2018). Fault Tolerant Control Using Fractional-order Terminal Sliding Mode Control for Robotic Manipulators, *Studies in Informatics and Control*, *27*(1), 55-64. ISSN 1220-1766.

2. Al-Junaid, H. (2015). ANN Based Robotic Arm Visual Servoing Nonlinear System, *Procedia Computer Science*, *62*, 23-30.

3. Almusawi, A. R. J., Dülger, L. C. & Kapucu, A. (2016). New Artificial Neural Network Approach in Solving Inverse Kinematics of Robotic Arm (Denso VP6242), *Computational Intelligence and Neuroscience*, 2016.

4. Benjamin, B. C. & Charles, L. (1992). Inverse Kinematics Problem in Robotics Using Neural Networks, *NASA Technical Memorandum*, *105869*.

5. Bingul, Z., Ertunc, H. M & Oysu, C. (2005). Comparison of Inverse Kinematics Solutions Using Neural Network for 6R Robot Manipulator with Offset. In *Computational Intelligence Methods and Applications Congress*.

6. Bustamante, L. & Gu, J. (2007). Localization of Electrical Outlet for a Mobile Robot Using Visual Servoing. In *Canadian Conference on Electrical and Computer Engineering.*

7. Camera Calibration Toolbox for Matlab, <http://www.vision.caltech.edu/bouguetj/calib_doc/>.

8. Choi, W., Ryu, C. & Kim, H. (1999). Navigation of a Mobile Robot Using Mono-Vision and Mono-Audition. In *Proceedings*. *1999 IEEE International Conference on Systems, Man, and Cybernetics*, *vol. 4*.

9. Claudio, X. G., Agravante, D. J., Spindler, F. & Chaumette, F. (2016). Dual Arm Manipulation and Whole Body Control with the Humanoid Robot Romeo by Visual Servoing, *Journées Nationales de la Recherche Humanoïde.* Toulouse, France.

10. Corke, P. I. (2017). *Robotics, Vision & Control*. Springer. ISBN 978-3-319-54413-7.

11. Craig, J. (2005). *Introduction to Robotics*. Prentice Hall.

12. Ezio, M. (2002). Survey of Vision-based Robot Control. In *European Naval Ship Design, Captain Computer IV Forum, ENSIETA*. Brest, France.

13. Feng, Y., Yao-nan, W. & Yi-min, Y. (2012). Inverse Kinematics Solution for Robot Manipulator based on Neural Network under Joint Subspace, *Int. J. of Computer Communications & Control*, *17*(3).

14. Gonzalez, R. C. & Woods, R. E. (2008). *Digital Image Processing* (*3rd Edition*). Prentice Hall.

15. Hagan, M. T., Demuth, H. B., Beale, M. H. & De Jesús, O. (2002). *Neural Network Design* (*2nd Edition*). ISBN-10: 0-9717321-1-6.

16. Haykin, S. (1999). *Neural Networks and Learning Machines* (*3rd Edition*). ISBN-10: 0-13-147139-2.

17. Jha, P. & Biswal, B. B. (2015). Inverse Kinematic Solution of 5R Manipulator Using ANN and ANFIS, *IAES Int. J. of Robotics and Automation (IJRA)*, *4*(2).

18. Kara, T. & Mary, A. H. (2017). Adaptive PD-SMC for Nonlinear Robotic Manipulator Tracking Control, *Studies in Informatics and Control*, *26*(1), 49-58. ISSN 1220-1766.

19. Knoeppel, C., Schanz, A. & Michaelis, B. (2000). Robust Vehicle Detection at Large Distance Using Low Resolution Cameras. In *Proceedings of the IEEE Intelligent Vehicles Symposium.* Dearborn.

20. Lou, Y. F. & Brunn, P. (1999). A Hybrid Artificial Neural Network Inverse Kinematic Solution for Accurate Robot Path Control. In *Proc. of the Inst. of Mech. Eng., Part I: Journal of Systems and Control Eng.*, *213*(1) (pp. 23-32).

21. Mondi, K. P., Sahin, F. & Saber, E. (2005). An Application of Human Robot Interaction: Development of a Ping-Pong Playing Robotic Arm. In *IEEE Int. Conference on System, Man and Cybernetics*, *vol. 2*.

22. Mundhra, K., Suluh, A., Sugar, T. & McBeath, M. (2002). Intercepting a Falling Object: Digital Video Robot. In *Proceeding of the IEEE Int.Conf. on Robotics and Aut.*

23. Raptis, S. N., Tzafestas, E. S. & Tzafestas, S. G. (2007). Robot Inverse Kinematics via Neural and Neurofuzzy Networks: Architectural and Computational Aspects for Improved Performance, *Journal of Information and Optimization Sciences*, *28*(6).

24. Sahin. T. & Zergeroglu, E. (2006). Adaptive 3D Visual Servo Control of Robot Manipulators via Composite Camera Input, *Turkish J. of Elec. Eng*, *14*(2).

25. Srisuk, P., Sento, A. & Kitjaidure, Y. (2017). Inverse Kinematics Solution using Neural Networks from Forward Kinematics Equations. In *9th International Conference on Knowledge and Smart Technology*.

26. Yang S. S., Moghavvemi, M. & Tolman J. D. (2000). Modeling of Robot Inverse Kinematics Using Two ANN Paradigms. In *TENCON Proceedings. Int. Sys. and Tec. for the New Millennium*, *vol. 3* (pp. 173-177).