A Smart Robot Arm Design for Industrial Application

Amira Y. HAIKAL, Mostafa A. EL-HOSSEINI

Faculty of Engineering, Mansoura University,
35516, Egypt,
amirayh@gmail.com, melhosseini@gmail.com.

Abstract: The proposed paper outlines the design and implementation of smart robotic arm that is equipped with a vision system. Three main parts cooperate to perform the control of the proposed arm. Image processing, inverse kinematics and control are involved in the robot arm design. Forward and inverse kinematic are solved using homogenous transformation matrices and Denavit-Hartenberg's systematic representation of reference systems. The arm uses vision information imported from the processing of the captured image of the target object to decide how to proceed. The proposed system is implemented on Microcontroller PIC 16F877a.The code involves securing the system from power failure as well as hacking. The controller has the ability to detect power failure and fix it allowing the arm to proceed from the stopping point avoiding starting over again (retentive). Practical application of the proposed system shows that its ability to handle different objects whether it is learned before or not is superior.

Keywords: Image processing, Inverse kinematics, Denavit-Hartenberg, Microcontroller, Robotic arm.

1. Introduction

Factories and workshops which still lack the intelligence in their machines and still depend on human have bad effects either on human, due to health complications that may cause death, or on productivity. Previous problems lead to the use of intelligent robots with lower cost. The need for artificial intelligence and vision system is motivated by their capability to perceive the environment and take actions that maximize chances of success with the enterprise of automating and integrating a wide range of processes and representations for vision perception [1]. This article presents a self eyed robotic arm that can serve in many places with many applications like painting, welding, drawing and assembling. Robotic arm is used instead of human as it is more powerful, accurate, saves the materials, efficient in repetitive tasks, reduces the cost, and for all previous reasons this arm can improve productivity and quality of production.

Vision allows humans to perceive and understand the world surrounding them, while computer vision aims to duplicate the effect of human vision by electronically perceiving and understanding an image [1]. Giving computers the ability to see is not an easy task. Sometimes, equipment will deliver images that are 3D but this may be of questionable value: analyzing such datasets is clearly more complicated than 2D, and sometimes the 'three-dimensionality' is less than intuitive to us. Terahertz scans are an example of this [2]. Dynamic scenes such as those to which we are accustomed, with moving objects or a moving camera, are increasingly common and represent another way of making computer vision more complicated. Image processing involves changing the nature of an image in order to either improve its pictorial information for human interpretation or to render it more suitable for autonomous machine perception [3-7]. We are concerned with digital image processing, which involves using a computer to change the nature of a digital image. Humans like their images to be sharp, clear and detailed; however, machines prefer their images to be simple and uncluttered [8].

The inverse kinematic robotics problem has been the focus of kinematic analysis for robot manipulators. In order to determine all possible formations to place the end effector of a robot manipulator at a particular point in space, we must compute the movements associated with each joint variable [9]. Forward kinematics finds the value of the end position of the arm given the robot's current joint angles. There are several methods to resolve this problem. In the presented work it is done using the homogenous transformation matrices and Denavit−Hartenberg’s systematic representation of reference systems. This is because although you may find the final position geometrically, this method offers a response which could relate the position of the end of each link in the kinematic chain compared to the previous reference system in order to define the position of each articulation in the robot [10]. Forward kinematics contributes to perform inverse kinematics and find joint angles that put the robot's arm into a general desired position and orientation. Actually, the forward kinematic

problem uses the kinematics equations to determine the pose given the joint angles. While, the inverse kinematics problem computes the joint angles for a desired pose of the figure.

The smart arm implementation here is based on three main parts: Image processing, inverse kinematics and control. Where, image processing algorithm recognizes a new object then makes some processing and compares it with the stored ones in the database.

While inverse kinematics is responsible for obtaining the mathematical equations for converting the points representing the object's coordinates, which come from the image processing algorithm, into angles for moving the motors of the arm. Finally, the control part is responsible for teaching the robotic arm manually the correct coordinates of the object (which are recorded) as well as, the set of movements to be retrieved later during a redo process. Moreover, the arm is controlled automatically after receiving the angles of the motors from inverse kinematics process which is implemented using PIC microcontrollers.

The rest of the paper is organized as follows: the proposed framework is presented in details in section 2. Section 3 covers the hardware implementation. Experimental results are illustrated in section 4.

2. Proposed framework

Initially the objects' image (to be painted) is captured using a camera in a fixed position. Some processing is done to recognize this picture using image processing algorithm, then the picture is given an ID and both the picture and its ID are stored in a database. Next, a new process called learning begins. In this process, the object moves until it reaches a position in front of the presented robotic arm. A joystick is then used to control the robotic arm so that it can "paint" the object totally. The movements of the arm robot are recorded and stored in a database in the microcontroller memory and they are given an ID similar to the ID that was given to the picture of the object. This ends the learning process of the proposed system on the object.

When another object is presented to the proposed system, the camera captures its picture to compare it with the objects' picture stored in the database. Only then there will be two options for the system. The first option will be valid if the comparison finds a match between the newly presented object and the stored one. In this case, the ID of the stored matched object in the database is sent serially to the control system so that it can repeat the recorded movements stored in the database before by the microcontroller.

While the second option occurs when the comparison results is a no-match situation (i.e. the newly presented object is not in the stored database). In this case, some processing is done on the picture of the new object. This processing aims at dividing the picture into groups (grouping) so that we can reach the required points that we need to fully cover the object in the painting process .These points are then passed to the inverse kinematics part which is used to find the corresponding motor angles that are required to move the arm robot to each point. A flow chart describing the proposed framework is illustrated in Figure 1.

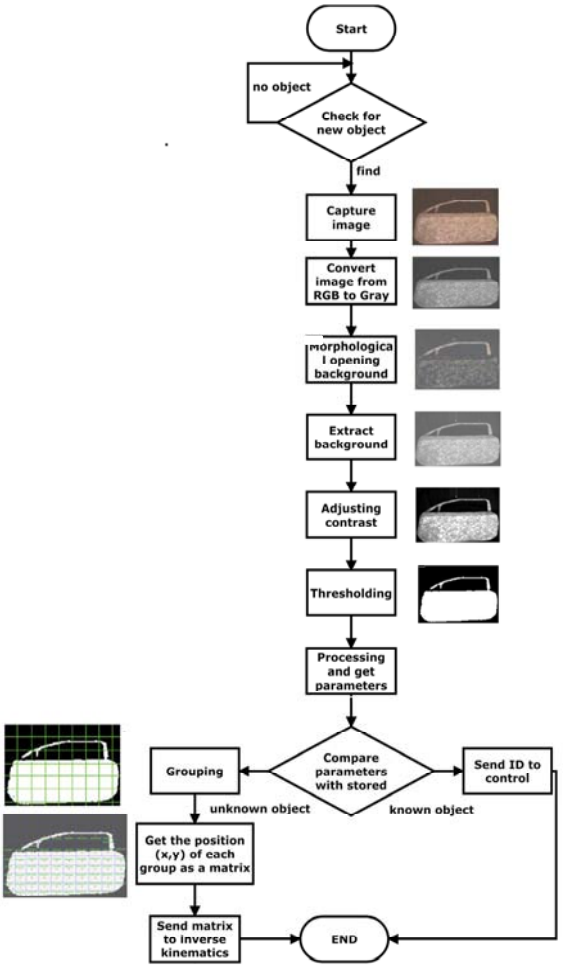


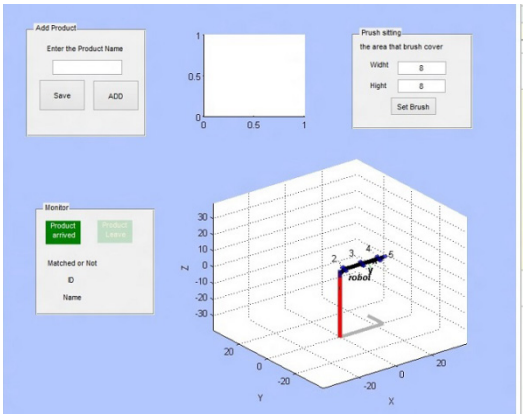
Figure 1. Flowchart of the proposed Framework.

2.1 Image Processing

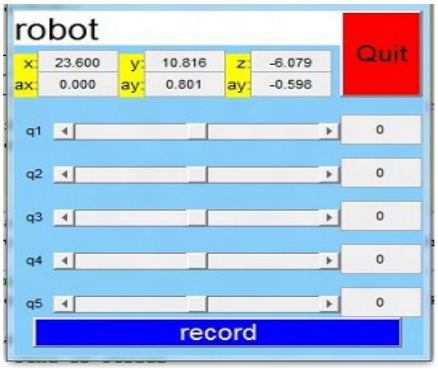
The vision system here depends on comparing features of products with known specification to decide if the arm robot is trained to deal with it or not. The component of such system is a camera (digital camera) connected to a PC through USB port. PC has software that receives the coming images then makes the processing and the decision to finally send the output to the arm robot controller.

Choosing the camera depends on three parameters: The size of the out coming image which depends on the product, resolution of the out coming image that depends on the robot application and focal length that depends on the area of the work space in which a robot performs.

Initially, the camera captures the image of the target object. A pre-processing task begins after obtaining gray scale from the colored image. The Morphological Opening is used on the gray scale or binary image with the structuring element SE [11]. The morphological opening operation is erosion followed by a dilation using the same structuring element for both operations [12]. Dilation adds pixels to the boundaries of object in an image, while erosion removes pixels on object boundaries. The number of pixels removed or added depends on the size and shape of the structuring element used to process the image. The next step subtracts the background image from the original Image, then increase the image contrast by adjusting image intensity or color map. Thresholding the image is followed to convert the image to binary. Finally, get the parameters and store them in database with the object ID to retrieve when it's needed for comparison with other arriving objects in the production line. If a match occurs between new object and the stored one, the id is sent to the control unit by the serial port. Otherwise, the arm robot is not leaned, so use grouping to send the points the path that it should follow. Where, the image is separated to group of pixels, size of each group depends on the end effector's (spray) size. The position of each group is represented by the position of the pixel in the center of it. Figure 2 shows the proposed interface of the image processing algorithm with MATLAB.



(a)



(b)

Figure 2 (a,b). Image processing interface with MATLAB GUIDE.

2.2 Forward and inverse kinematics

The assistance of an inverse kinematics algorithm gives the animator merely the desired location of certain chosen points on the body and relies on the algorithm to automatically compute a set of joint angles that satisfy the end-effector constraints which represents our focus in this paper [13-16].

Figure 3 illustrates the presented robotic arm with the joint angles appearing on it. We represent each joint using Denavit-Hartenberg transformation matrix and to find position and orientation of end-effector we multiply all the transformation matrices. Denavit-Hartenberg method is discussed later in details in section 3.1.

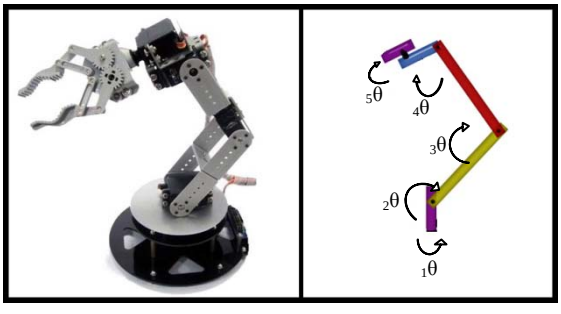


Figure 3. Robotic Arm with joint angles.

2.3 Control

Controlling the real-time movements of arm's servo motors depends on the software program's results from the previous stage (Inverse Kinematics and Modeling). After converting points and coordinates into angles they are passed to motors and used by microcontroller control code to perform control tasks of the designed arm. Microcontroller records and saves stopping positions when record button is pressed. When redo button is pressed, motors return to their initial positions and begin to repeat the motion recorded which provides arm with the ability of learning manually. To make this option more reliable, we provide it with a security system to enable only the authorized persons to enter the system. We also provide it with a recovery system when power failures occurred and continue working from the last position before failure.

2.3.1 Denavit Hartenberg method

The Denavit-Hartenberg parameters (DH parameters) are the four parameters associated with a particular convention for attaching reference frames to the links of a spatial kinematic chain or robot manipulator [17].

A homogeneous matrix A_i^{i-1} is a 4x4 matrix that contains information related to the position and orientation of the reference system attached to the link i of the manipulator compared with the reference system of the link i-1. In that way the matrix A_1^0, represents the position and orientation of the coordinate system S_1 compared with the coordinate system S_0. If S_0 is placed in the frame of the manipulator and S_1 in the end of link one, the matrix A_1^0 represents the position of S_1 compared with the fixed coordinate system of the robot, Figure 4. So that the representation of the end position of the manipulator is A_n^0 where n is the degrees of freedom DOF of this matrix. This matrix is usually named T and is given by equation (1):

$$T = A_n^0|_{n=5} = A_1^0 A_2^1 A_3^2 A_4^3 A_5^4 \quad (1)$$

For the calculus of A_i^{i-1} matrices, DH parameters must be defined, these parameters rely exclusively on the geometric characteristics of each link and allow placing the coordinate systems in each. The coordinate systems must be placed like in Figure 4, to comply with the characteristics of the DH parameters which are defined next.

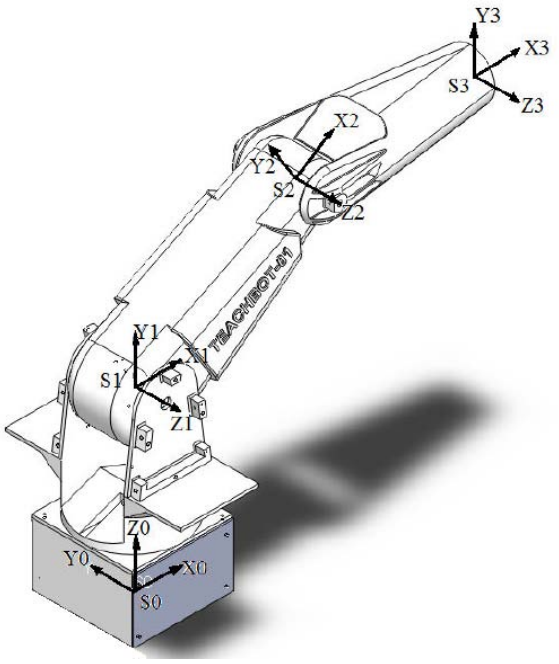


Figure 4. Location of coordinate systems.

The DH parameter characteristics are [18]:

- a_i is the link length, i.e., translation along Xi axis.
- α_i is the link twist angle, i.e., Rotation around Xi axis.
- d_i is the Joint distance, i.e., Translation along Zi-1 axis.
- θ_i is the Joint angle , i.e., Rotation around Zi-1 axis. Which are the required angles, giving it initial value =0

The calculation of DH parameters is illustrated in Figure 5.

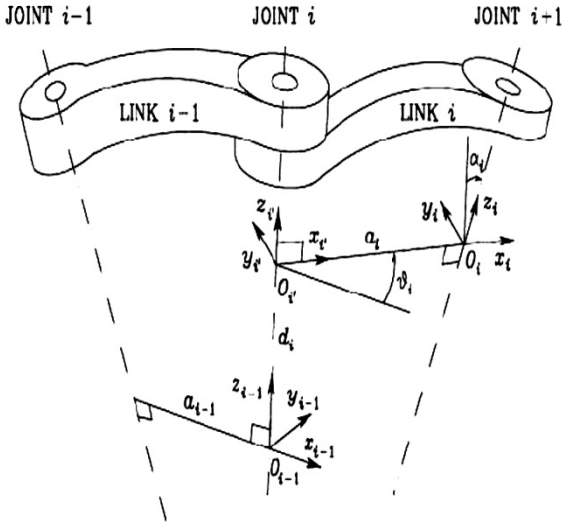


Figure 5. Denavit-Hartenberg parameters calculation.

Initially we define previous constants to each link of the arm robot as in Table 1.

Table 1. Initial values for DH parameters for each link of the arm robot.

Link	a_i	α_i	d_i	θ_i
1	2.5	90	2	0
2	9.5	0	0	0
3	9.1	0	0	0
4	2.5	90	0	0
5	0	0	13.5	0

A matrix DH which is defined from table 1:

$$DH = \begin{bmatrix} 0 & 2 & 2.5 & 90 \\ 0 & 0 & 9.5 & 0 \\ 0 & 0 & 9.1 & 0 \\ 0 & 0 & 2.5 & 90 \\ 0 & 13.5 & 0 & 0 \end{bmatrix} \quad (2)$$

Then define the arm robot to the Matlab through "SerialLink" command:

Arm =SerialLink(dh);

The general form of orientation matrices is given by:

$$A_i^{i-1} = \begin{bmatrix} Cos\theta_i & -Sin\theta_i Cos\alpha_i & Sin\theta_i Sin\alpha_i & a_i Cos\theta_i \\ Sin\theta_i & Cos\alpha_i Cos\theta_i & -Sin\alpha_i Cos\theta_i & a_i Sin\theta_i \\ 0 & Sin\alpha_i & Cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3)$$

Replacing the values of Table 1 in equation 3:

For Link 1

$$A_1^0 = \begin{bmatrix} Cos\theta_1 & 0 & Sin\theta_1 & a_1 Cos\theta_1 \\ Sin\theta_1 & 0 & -Cos\theta_1 & a_1 Sin\theta_1 \\ 0 & 1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4)$$

For Link 2

$$A_2^1 = \begin{bmatrix} Cos\theta_2 & -Sin\theta_2 & 0 & a_2 Cos\theta_2 \\ Sin\theta_2 & -Cos\theta_1 & 0 & a_2 Sin\theta_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (5)$$

For Link 3

$$A_3^2 = \begin{bmatrix} Cos\theta_3 & -Sin\theta_3 & 0 & a_3 Cos\theta_3 \\ Sin\theta_3 & Cos\theta_3 & 0 & a_3 Sin\theta_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (6)$$

For Link 4

$$A_4^3 = \begin{bmatrix} Cos\theta_4 & 0 & Sin\theta_4 & a_4 Cos\theta_4 \\ Sin\theta_4 & 0 & -Cos\theta_4 & a_4 Sin\theta_4 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (7)$$

For Link 5

$$A_5^4 = \begin{bmatrix} Cos\theta_5 & -Sin\theta_5 & 0 & 0 \\ Sin\theta_5 & Cos\theta_5 & 0 & 0 \\ 0 & 0 & 1 & d_5 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (8)$$

For getting transformation matrix that represents the relation between link 1 and link 5 we multiply the 5 matrices as in equation (1).

After simplifying, the resulting matrix is:

$$T = \begin{bmatrix} R_{11} & R_{12} & R_{13} & P_x \\ R_{21} & R_{22} & R_{23} & P_y \\ R_{31} & R_{32} & R_{33} & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (9)$$

As θ_5 is the angle of end effector and we ignore it, so let θ_5=0;

After simplifying,

$$R_{11} = \cos(\theta_1)*\cos(\theta_2+\theta_3+\theta_4) \quad (10)$$

$$R_{12} = \sin(\theta_1) \quad (11)$$

$$R_{13} = \cos(\theta_1)*\sin(\theta_2+\theta_3+\theta_4) \quad (12)$$

$$\begin{aligned} P_x = {} & \cos(\theta_1)*(d_5*\sin(\theta_2+\theta 3+\theta 4) \\ & + a_4*\cos(\theta_2+\theta 3+\theta 4) \\ & + a_3*\cos(\theta_2+\theta_3)+a_2*\cos(\theta_2)+a_1) \end{aligned} \quad (13)$$

$$R_{21} = \sin(\theta_1)*\cos(\theta_2+\theta_3+\theta_4) \quad (14)$$

$$R_{22} = -\cos(\theta_1) \quad (15)$$

$$R_{23} = \sin(\theta_1)*\sin(\theta_2+\theta_3+\theta_4) \quad (16)$$

$$\begin{aligned} P_y = {} & \sin(\theta_1)*(d_5*\sin(\theta_2+\theta_3+\theta 4) \\ & + a_4*\cos(\theta_2+\theta_3+\theta 4)) \\ & + a_3*\cos(\theta_2+\theta_3)+a_2*\cos(\theta_2)+a_1) \end{aligned} \quad (17)$$

$$R_{31} = \sin(\theta_2+\theta_3+\theta_4) \quad (18)$$

$$R_{32} = 0 \quad (19)$$

$$R_{33} = -\cos(\theta_2+\theta_3+\theta_4) \quad (20)$$

$$\begin{aligned} P_z = {} & -d_5*\cos(\theta_2+\theta_3+\theta 4) \\ & + a4*\sin(\theta_2+\theta_3+\theta_4) \\ & + a_3*\sin(\theta_2+\theta_3)+a_2*\sin(\theta_2)+d_1 \end{aligned} \quad (21)$$

By dividing eq. (11)/eq. (15):

$$\theta_1=\tan^{-1}\left(-\frac{R12}{R22}\right) \quad (22)$$

By dividing eq. (12)/eq. (10):

$$\therefore \theta_2+\theta_3+\theta4=\tan^{-1}\left(\frac{R13}{R_{11}}\right) \quad (23)$$

By dividing eq. (16)/eq. (14):

$$\therefore \theta_2+\theta_3+\theta4=\tan^{-1}\left(\frac{R23}{R_{21}}\right) \quad (24)$$

By dividing eq. (18)/eq. (20):

$$\therefore \theta_2+\theta_3+\theta4=\tan^{-1}\left(-\frac{R31}{R_{33}}\right) \quad (25)$$

From eq. (21)

$$\begin{aligned} P_z &= d_5*R_{33}+a4*R_{31} \\ &\quad + a_3*\sin(\theta_2+\theta_3)+a_2*\sin(\theta_2)+d_1 \\ \therefore a_3*\sin(\theta_2+\theta_3)&+a_2*\sin(\theta_2) \\ &= P_z - d_5*R_{33} - a4*R_{31} - d_1 \end{aligned} \quad (26)$$

By adding eq. (13) + eq. (17)

$$\begin{aligned} \therefore a_3*\cos(\theta_2+\theta_3)&+a_2*\cos(\theta_2)= \\ &=\sqrt{Px^2+Py^2}-d_5*R_{31}+ \\ &+a4*R_{33}-a_1 \end{aligned} \quad (27)$$

For simplification let θ_2 = constant = 0

And substituting in eqs. (26) & (27):

$$\therefore a_3*\sin(\theta_3) = P_z - d_5*R_{33} - a4*R_{31} - d_1 \quad (28)$$

$$\begin{aligned} \therefore a_3*cos(\theta_3) &= \sqrt{Px^2+Py^2}- \\ &-d_5*R_{31}+a4*R_{33}-a_1-a_2 \end{aligned} \quad (29)$$

By dividing equ (28)/equ (29)

$$\theta_3=\tan^{-1}\left(\frac{Pz-d5*R33-a4*R31-d1}{\sqrt{Px^2+Py^2}-d5*R31+a4*R33-a1-a2}\right) \quad (30)$$

By substituting in equ (23) by θ_2=0& θ_3

$$\theta_4=\tan^{-1}\left(\frac{R13}{R11}\right)-\theta_3 \quad (31)$$

Equations 22, 30, 31 are the solutions of inverse kinematics of the proposed robot arm.

3. Hardware Implementation

The embedded control code is produced to control the project manually as well as automatically according to the state of PIC 16F877a involved. Where, in dealing with the manual control using joystick, each servo motor can be moved to any position using PWM with variable duty cycle and two switches to increase and decrease that duty. Therefore, the servo motor moves right and left (±90 angle). Moving the 5 servo motors permit the whole gripper of the arm to reach any point (X,Y) within its dimension space. For monitoring purposes some information need to be displayed on an LCD. This information include the recorded position of each servo motor in non-volatile memory (EEPROM), the recorded track which the arm will redo, as well as the recorded capacity of the memory.

The code involves securing the system from power failure as well as hacking. This is done using PIC 18F4550, where password is required to move the arm manually, record new object, delete exciting object or erase the whole memory. The password is variable and can be changed knowing the old one, also it can be reset to the default one.

The controller has the ability to detect power failure and fix it allowing the arm to proceed from the stopping point avoiding starting over again. Figure 6 shows hardware implementation of the proposed system.

This information during processing is displayed on LCD. For the purpose of receiving image processing action a serial PIC 16F877a is used to communicate with PC and a stepper controller. Also, PIC 16F877a is used to move the object across the production line synchronously with the system and stop the object using limit switch which exists in front of the camera.

After validating the inverse kinematic model, it is found that the proposed robotic arm shows an acceptable behavior.

4. Experimental Results

When the presented object produce a comparison match with the stored object, the ID of the stored matched object in database is sent serially to controller to repeat the recorded movement stored before by microcontroller. We measured the average time for the redo process which the arm takes to redo 6 prerecorded points. For that purpose, we recorded "**15**" random times in sec.: 11.36, 10.80, 11.00, 12.75, 11.80, 12.61, 10.90, 11.80, 10.87, 12.97, 13.44, 12.00, 12.93, 12.03, and 13.31.

The redo process average time is 12.038 Sec. When the process is tested on a 6 points

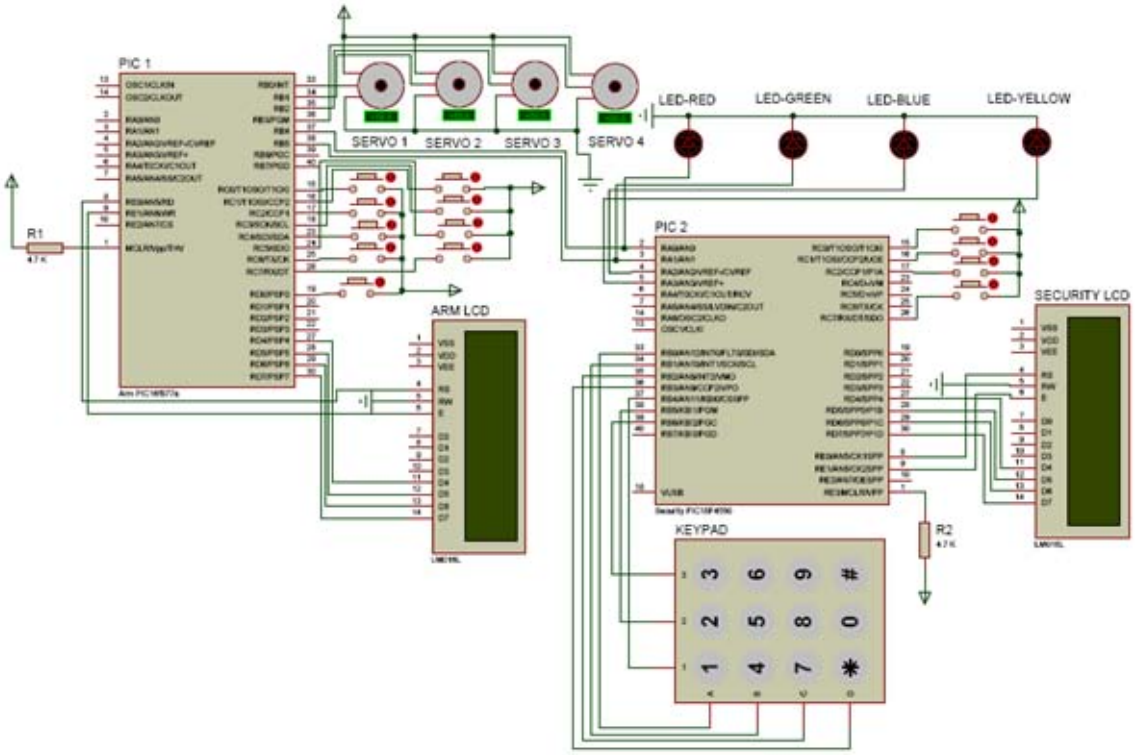


Figure 6. Hardware Implementation of the proposed project.

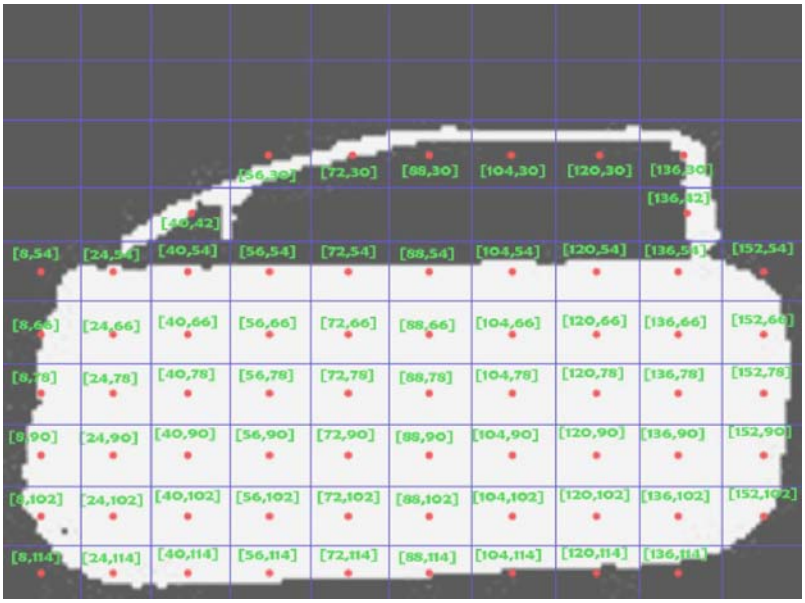


Figure 7. The position of each group as a matrix for the chosen object (car door).

prerecorded, Target Object "car door" is placed at an initial fixed place, and the arm fail to reach only 1 point correctly. The accuracy is 83.333 %. After more trials and replacing the servo motors by better torque, the arm reaches the 6 prerecorded points correctly. However, when the presented object is new where its comparison with the stored objects produces no match, a grouping process starts. When image processing complete grouping process, it calls inverse function and sends the coordinates to find the corresponding angles to the arm movement then convert them to PWM to control servo motors. Results of the grouping process are shown in Figure 7, illustrating the position of

Table 2. The resulted angles from the inverse function for each point.

Points	Angles (PWM)					Points	Angles (PWM)					Points	Angles (PWM)				
X,Y	Θ_1	Θ_2	Θ_3	Θ_4	Θ_5	**X,Y**	Θ_1	Θ_2	Θ_3	Θ_4	Θ_5	**X,Y**	Θ_1	Θ_2	Θ_3	Θ_4	Θ_5
56,30	66	0	146	52	0	**88,66**	71	0	148	52	0	**136,90**	69	0	149	52	0
7,30	63	0	147	52	0	**104,66**	69	0	149	52	0	**152,90**	68	0	149	52	0
88,30	61	0	148	52	0	**120,66**	67	0	149	52	0	**8,102**	98	0	148	52	0
104,30	60	0	148	52	0	**136.66**	65	0	149	52	0	**24,102**	93	0	148	52	0
120,30	59	0	149	52	0	**152,66**	64	0	149	52	0	**40,102**	89	0	148	52	0
136,30	58	0	149	52	0	**8,78**	97	0	147	52	0	**56,102**	85	0	148	52	0
40,42	77	0	146	52	0	**24,78**	91	0	147	52	0	**72,102**	81	0	149	52	0
136,42	60	0	149	52	0	**40,78**	86	0	148	52	0	**88,102**	78	0	149	52	0
8,54	96	0	145	52	0	**56,78**	81	0	148	52	0	**104,102**	75	0	149	52	0
24,54	87	0	146	52	0	**72,78**	77	0	148	52	0	**120,102**	73	0	149	52	0
40,54	80	0	147	52	0	**88,78**	74	0	148	52	0	**136,102**	71	0	149	52	0
56,54	75	0	147	52	0	**104,78**	71	0	149	52	0	**152,102**	70	0	149	52	0
72,54	71	0	148	52	0	**120,78**	69	0	149	52	0	**8,114**	98	0	148	52	0
88,54	68	0	148	52	0	**136,78**	67	0	149	52	0	**24,114**	94	0	148	52	0
104,54	66	0	148	52	0	**152,78**	66	0	149	52	0	**40,114**	90	0	149	52	0
120,54	64	0	149	52	0	**8,90**	98	0	148	52	0	**56,114**	86	0	149	52	0
136,54	63	0	149	52	0	**24,90**	92	0	148	52	0	**72,114**	83	0	149	52	0
152,54	62	0	149	52	0	**40,90**	87	0	148	52	0	**88,114**	80	0	149	52	0
8,66	97	0	147	52	0	**56,90**	83	0	148	52	0	**104,114**	77	0	149	52	0
24,66	90	0	147	52	0	**72,90**	79	0	148	52	0	**120,114**	75	0	149	52	0
40,66	83	0	147	52	0	**88,90**	76	0	149	52	0	**136,114**	73	0	149	52	0
56,66	78	0	148	52	0	**104,90**	73	0	149	52	0	**152,114**	71	0	149	52	0
72,66	74	0	148	52	0	**120,90**	71	0	149	52	0	-	-	-	-	-	-

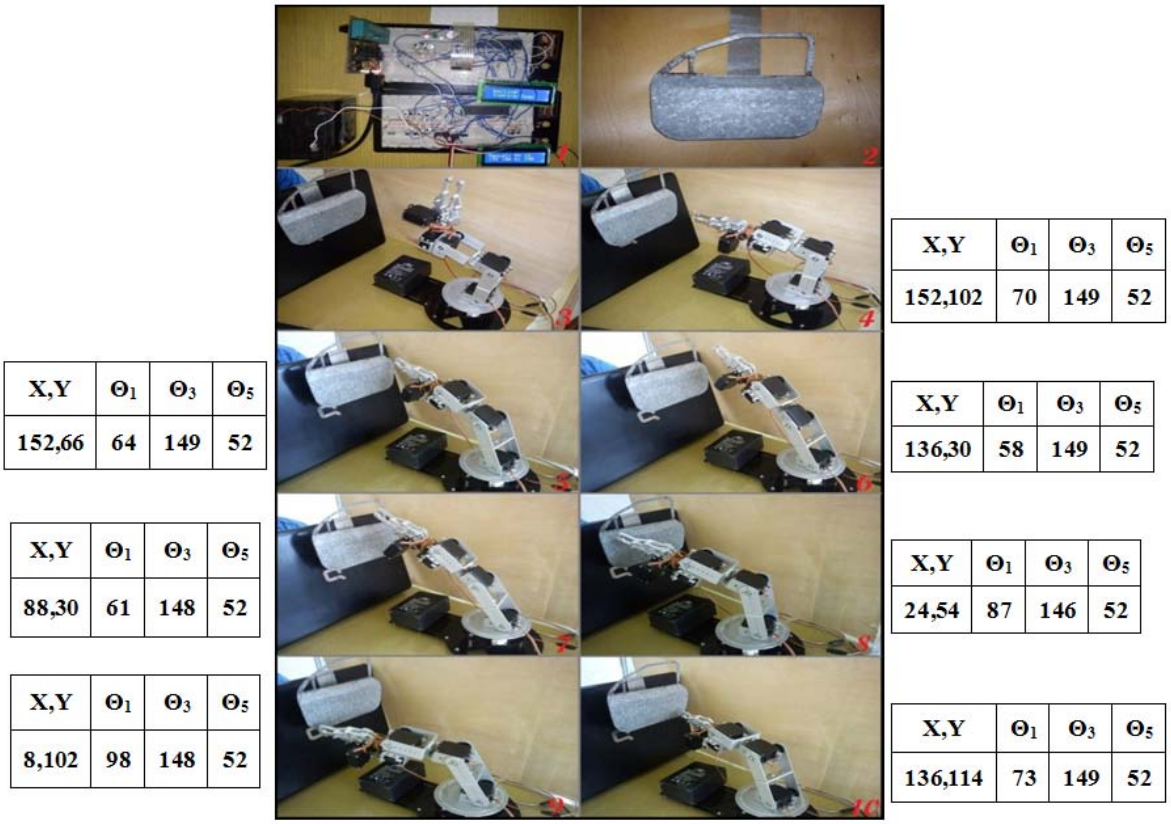
each group as a matrix for the chosen object (car door). This matrix is sent to inverse function returning the needed angles for the arm movement. Table 2 shows the returned angles. Figure 8 illustrates the control circuits, the chosen object to be dealt with the arm (car door), as well as snapshots illustrating a real time arm movement to locate some of the door coordinates.

Figure 8 shows different subsequent steps for the arm trying to locate the correct coordinates for the car door with the help of the calculated angles from Table 2. Each image representing a move of the arm is associated with its corresponding parameters as shown in figure 8.

After validating the inverse kinematic model, it is found that the proposed robotic arm shows an acceptable behavior.

5. Conclusion

A smart robot arm with a vision system is presented in this paper. The presented arm can be controlled with the help of vision information to locate coordinates of a target object for several purposes such as painting, welding, drawing, and assembling. The arm can manipulate the target object either manually using a joystick or automatically. Manual operation teaches the arm the correct movement to locate the coordinates of the target object and record them for the redo process (real time movement). On the other



X,Y	Θ1	Θ3	Θ5
152,66	64	149	52

X,Y	Θ1	Θ3	Θ5
88,30	61	148	52

X,Y	Θ1	Θ3	Θ5
8,102	98	148	52

X,Y	Θ1	Θ3	Θ5
152,102	70	149	52

X,Y	Θ1	Θ3	Θ5
136,30	58	149	52

X,Y	Θ1	Θ3	Θ5
24,54	87	146	52

X,Y	Θ1	Θ3	Θ5
136,114	73	149	52

Figure 8. Real time Movements of the arm to locate the door coordinates.

hand the arm has the ability to automatically deal with a new unseen object by importing its information from previous stage (image processing and inverse kinematics) which define object coordinates and convert them to angles passed to motors for performing the specified control task. The proposed system is implemented on Microcontroller PIC 16F877a. The code involves securing the system from power failure as well as hacking. The controller has the ability to detect power failure and fix it allowing the arm to proceed from the stopping point, avoiding starting over again (retentive). The proposed robotic arm is able to locate the correct coordinates of the target object for the redo process as well as for the inverse case.

Acknowledgement

The authors would like to express their sincere gratitude to Eng. Maram Gamal, Eng. Alaa Eldeen Magdy, Eng. Ayman Galal and Eng. Zainab Ibrahim for their valuable efforts and devoted times to the proposed work

REFERENCES

1. BALLARD, D. H., C. M. BROWN, **Computer Vision**, Prentice Hall, 1982. http://homepages.inf.ed.ac.uk/rbf/BOOKS/BANDB/bandb.htm.
2. SONKA, M., V. HLAVAC, R. BOYLE, **Image Processing Analysis, and Machine Vision**, International Student Edition (3rd) by Thomson Learning, 2008.
3. SINGHAL, A., M. SINGH, **Speckle Noise Removal and Edge Detection Using Mathematical Morphology**, International Journal of Soft Computing and Engineering (IJSC) , ISSN: 2231-2307, Vol. 1, Issue 5, November 2011.
4. GONZALEZ, R. C., R. E. WOODES, **Digital Image Processing**, Pearson Education (Singapore) Pvt. Ltd, 2009.
5. KAPADIA, H., A. PATEL, **Application of Hough Transform and Sub-pixel Edge Detection in 1-D Barcode Scanning**, Journal of Advanced Research in Electrical, Electronics and Instrumentation

Engineering, Vol. 2, Issue 6, June 2013, ISSN (Print): pp. 2320-3765, ISSN (Online): pp. 2278-8875.

6. **Basic Image Acquisition Procedure**, Matlab help.
7. ACHARYA, T., A. K. RAY, **Image Processing Principles and Applications**, ISBN-10: 0471719986, ISBN-13: 978-0471719984, Edition: 1, Wiley, Sep 8, 2005.
8. MCANDREW, A., **An Introduction to Digital Image Processing with Matlab**, Notes for SCM2511 Image Processing 1, Semester 1, 2004.
9. KENDRICKS, K., **Solving the Inverse Kinematic Robotics Problem: A Comparison of the Denavit-Hartenberg Matrix and Groebner Basis Theory**, PhD Thesis at Auburn University, Auburn, Alabama, 2007.
10. LEE, H. S., S. L. CHANG, **Development of a CAD/CAE/CAM System for a Robot Manipulator**, Journal of Materials Processing Technology, ISSN: 0924-0136, 140, 2003, pp. 100-104.
11. SREEDHAR, K., B. PANLAL, **Enhancement of Images using Morphological Transformation**, International Journal of Computer Science & Information Technology (IJCSIT), Vol. 4, No 1, Feb 2012, pp. 33-50.
12. GIL, J., R. KIMMEL, **Efficient Dilation, Erosion, Opening, and Closing Algorithms**, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, Issue: 12, Dec 2002, pp. 1606-1617.
13. CUI, B.-Y., Z.-L. JIN, **Kinematics Analysis and Design of a Novel Robot Shoulder Joint**, Advanced Materials Research, vol. 646; 2013, pp. 139-143.
14. FANG, T., L. G. MING, T. KE, **Kinematics Analysis for a PRRR Manipulator**, Applied Mechanics and Materials, vol. 271, (PART 1), 2013, pp. 1578-1581.
15. UICKER, J. J., G. R. PENNOCK, J. E. SHIGLEY, **Theory of Machines and Mechanisms**, Oxford University Press, New York, 2003.
16. MCCARTHY, J. M., G. S. SOH, **Geometric Design of Linkages**, Springer, New York, 2010.
17. COLOME, A., **Smooth Inverse Kinematics Algorithms for Serial Redundant Robots**, Master Thesis, Institut de Robotica i Informatica Industrial (IRI), Universitat Politecnica de Catalunya (UPC), Barcelona, Spain, Sep. 2011.
18. ANDUEZA, L., I. AGUIRRE, **Three Degree of Freedom Robotic Manipulator Design for Educational Purpose**, Revista Ciencia e Ingenieria. Vol. 30, No. 1, diciembre-marzo, ISSN 1316-7081, 2009, pp. 3-14.