Armdroid Auto Picking System using Webcam Sensor

[1]Mamoun Jassim Mohammed,[2]Loay E. George, [3]Hayder Ayad, [4]Zaid Hashim Jaber, [5]Raad Ahmed Hadi
[1,5] Department of Computer Engineering , College of Engineering Al-Iraqia University, Iraq
[2,4] Department of Computer Science, College of Science, Baghdad University, Iraq
[3] College of Business Administration , Albayan University, Iraq
[1]mamoun_87@yahoo.com , [2]loayedwar57@scbaghdad.edu.iq, [3]hayder.alsultany@gmail.com, [4]zaidhashim@gmail.com, [5]raadwap@gmail.com

Abstract

This research paper presented a method for controlling the Armdroid. The method is developed to control the Armdroid for picking an object that lay on a surface plane using webcam sensor. The optical camera is used to feed the system continuously with sequence of images which convey the needed pictorial data for allocating the target position. The system is designed to accomplish the tasks required to carry the object and move it to the desired location. The transformation of target position from pictorial coordinate system to real word coordinates was accomplished using 2nd order polynomial transformation to translate the object position from image to real world position. The computation of angles of the robot arm segments is done continually to assure short maneuver route for moving the object. The style of movement is by issuing sequentially the proper commands for moving each segment individually such that a smooth movement is achieved. The conducted results indicated a well movement performance is attained.

Keywords: ARMDroid Control, Armdroid Pic, Optical Object Tracking, Coordinates Mapping.

1. Introduction

Automation technology is one of the developmental steps in today's science and industry. Smart robots are modified automation technology. Such technologies are commonly used in different countries for nuclear rescue as is the case in Fukushima or for home care. These robots have multiple functions, axes whether fully or partially devices [3,4] They manipulated automatically and have common mobility functions [5].

The ideal solution is "performing a manipulation that maximizes robustness and minimizes cost"; which is still a hard challenging problem. The particular important considerations should be taken during manipulation are the compliance and high fidelity force control. Force sensing and compliance at each robot joint can allow robot to safely interact in unstructured and unfamiliar environments [8].

Furthermore, smart robots are called so as they combine artificial intelligence, sensory detection technology mathematical computations, high sense of control, power, integration and precisely designed instruments . They are used in different domains, such as that of [6].

Many researchers developed algorithms to control the robot arm, [1] proposed a simple robot, which has a high sense of motion and velocity control. They further achieved a hybrid control of position and force of the robot manipulator while maintaining a perpendicular attitude to the target. Moreover, [2] developed 2D-vision robust system with industrial parts that are sensitive to lighting conditions. Results have shown that such smart robot have effective and dynamically expanded video cameras.

2. A Control System for Arm Robot

In this paper, a simple method based on computer vision via optical camera is proposed to establish a simple automatic system for controlling arm robot to accomplish certain task. The arm movements of such robots are synchronized to pick the moving objects on a conveyor belt. It aims to identify the target objects according to their color; the target objects are coming on the conveyor. After identifying the object place they are picked and

placed in its respective pre-programmed place using the robot arm. The introduced method helps eliminate human monotonous work through achieving high sense of accuracy and speed.
The project is color sensitive; it works by sending signals to the microcontroller. Once the microcontroller receives the, it sends it to the circuit to generate a motor function. Accordingly, the arm will be able to grab and place any object in any location. Based upon the detected color, the robotic arm moves to the specified place in order to hold the object using its clamp, pick it up, move accordingly. Once the caught object is released, the robot will go back to its original position [9,10].

2.1 The System Platform

This developed system consists of the following modules:

- ARMDROID Robot Arm (Motors and joints)
- Vision Sensor (webcam)
- Central Processing Unit
- Microcontroller with USB Driver

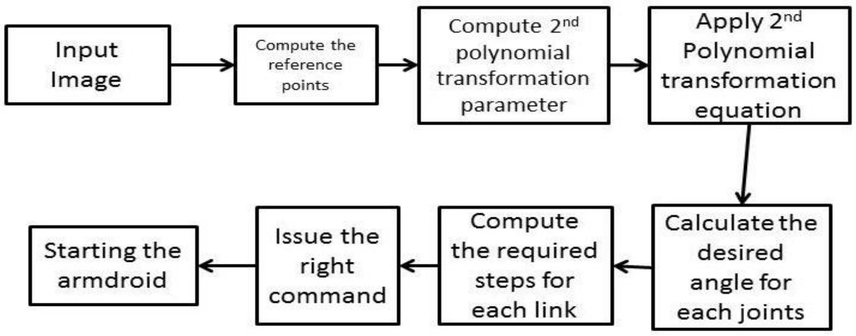
The ARMDROID I consists of five rotating axes and three gripping fingers that are all controlled and manipulated via a computer. The arms of such robots have a continuous path that allows the movements to be simultaneously controlled and operated. The unit works using a direct microcomputer keyboard control or a series of motions that can be repeated over and over are taught. The three rubber-made fingers are designed in a way that helps catching items of various sizes and shapes.
The ARMDROID feature is its parallelogram operation. The upper arm is vertical whereas the forearm is horizontal. The hand points downwards while the joint shoulder rotates in a way that the forearm and hand retain their orientation. The forearm can move upwards and downwards when the hand points downwards. This feature is highly significant when picking and placing objects. In fact, the robot has fixed number of joints which restrain the movement of the robot arm.
The topology is constructed in a way that allows manipulating the jobs of the robot. The base of the arm has to be fixed with some distance above the workspace. To be more precise, the arm will capture the objects on the target table within some distance because the shoulder lift joint has about 170 degrees of movement.
For this, camera used as input sensor, which will be projected down on the marked reference area, and connected to PC via USB. The camera will take a snap and it will be fed to PC for color processing. The system is equipped with a central processing unit to be capable of processing large amounts of data in real-time. The micro-controller (PIC18F4550) have been used to be responsible for driving the stepper motor and doing command translation after being issued from the computer via USB port, this microcontroller does not handle any of the primary processing tasks for controlling the ARMDROID. The movement of the robotic arm is fully controlled by stepper motors that in turn control the angular movement. The gripper of robotic arm depends on the size of the object and works automatically. Figure 1 shows the Armdroid.

Figure 1 ARMDROID I

2.2 Proposed System Description

The important aspect of this project is object picking and control the movement of the arm based on vision sensor. This system is designed to control the robot arm for picking object. An algorithm is developed to give the best result of the system. The Figure 2 shows block diagram of the algorithm. When the frame captured from the camera, the system detects the presence of reference mark area that has particular color. Then, it determines the coordinates of the table (in pixels) of the detected mark area appeared in the captured image plane.

Figure 2 System diagram


Input Image
Compute the reference points
Compute 2nd polynomial transformation parameter
Apply 2nd Polynomial transformation equation
Calculate the desired angle for each joints
Compute the required steps for each link
Issue the right command
Starting the armdroid

2.2.1 Simulation Functionality

The robotic arms usually perform movements looped by feedback mechanism. Often this leads to cumbersome user controls, making it difficult to operate. This problem inspired us to go for a using a virtual arm, a simple replica of robotic hand. The virtual arm operates like an actual robot arm with joint by joint replication. Also, the simulation has the same constraints of movements as like the actual robot arm. Our robot arm is of 5 degree freedom (DOF). The grabbing mechanism involves bowing and rotating the shoulder, bowing the elbow and the wrist rotating the hand, as shown in Figure 3.

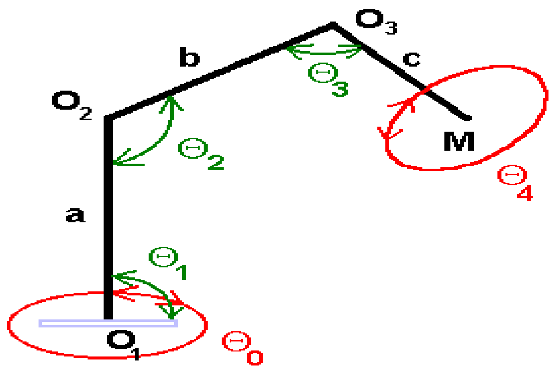
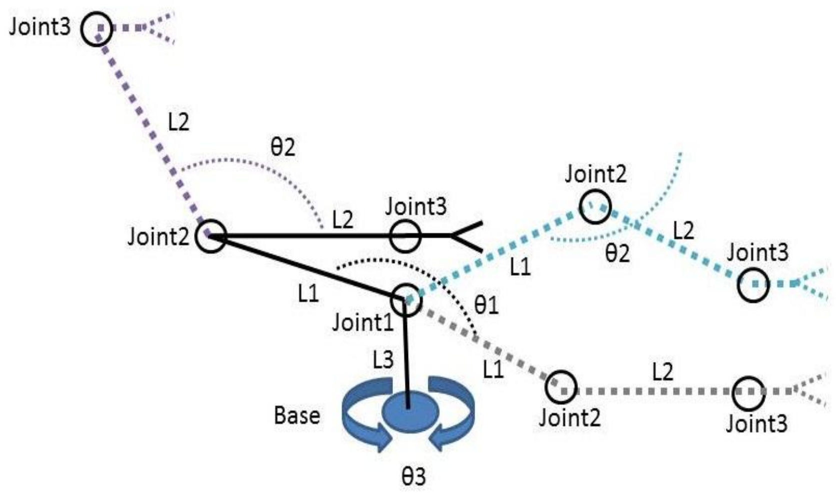


Figure 3 ARMDROID Joints.

Normally, the arm of the robot is 6 DOFs; this helps to access every point of the work-space at different incidence-angles easily. The project is limited to 5 DOFs as only 6 stepper motors are available with Armdroid. The sixth motor is set to control the grabbing-process.

Figure 4 the degree of freedom for each part of the ARMDROID.



The first step is drawing the shape of the robot arm. Then calculate the angles and the steps of movement for each part from its initial state to the limits. So, for our robot arm (θ_1, θ_2, θ_3) represents the angles and (L1, L2, L3) represents the upper arm and the forearm and the base. Mathematically to visualize the movement of these parts, we will use the equation of circle. Table 1 shows the length of the Armdroid parts.

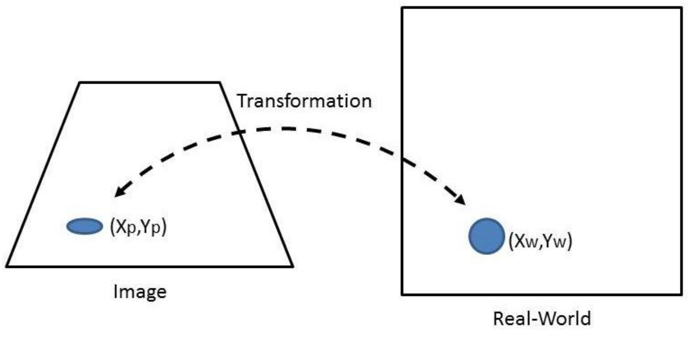
Table 1 ARMDROID main links, steps and angles.

Parts	Steps	Angle
L1	140	170
L2	80	130
L3	50	90

2.2.2 Object Location

The transformation of object location in image to its real world location needs a mathematical equations and methods to be achieved. There are many methods of transformation from image coordinates to real world coordinates. According to the transformation equations we can use polynomial transformations. The advantage of this method is to deter-mine the most probable values of the coefficients including their characteristics of accuracy. Polynomial transformations, unlike the linear transformations, use a polynomial relation between the coordinates. Generally, in practice, the first and second order of polynomials is used. The second order shows good results and it did significantly improve the transformation method. Figure 5 shows the scene of the target on surface.

Figure 5 Image-scene



The object location transformation requires the following:

- Requires computing 2nd polynomial equations between an image and scene surfaces.
- Employs **image-scene** mappings.

2.2.3 2nd Polynomial transformation

The non-linear ***polynomial*** transformation connects two 2D Cartesian coordinate systems via three processes: *translation, rotation* and a *variable scale change.* The transformation function might have an infinite number of terms. The equation is:

$$X' = x_0 + a_1X + a_2Y + a_3XY + a_4X^2 + a_5Y^2 + a_6X^2Y + a_7XY^2 + a_8X^3$$

$$Y' = y_0 + b_1X + b_2Y + b_3XY + b_4X^2 + b_5Y^2 + b_6X^2Y + b_7XY^2 + b_8X^3$$

Polynomial transformations sometimes georeference uncorrected satellite imagery or aerial photographs. They can further match improperly fitted vector data layers by stretching or rubber sheeting them over the most accurate data layer, as consider in Figure 6.

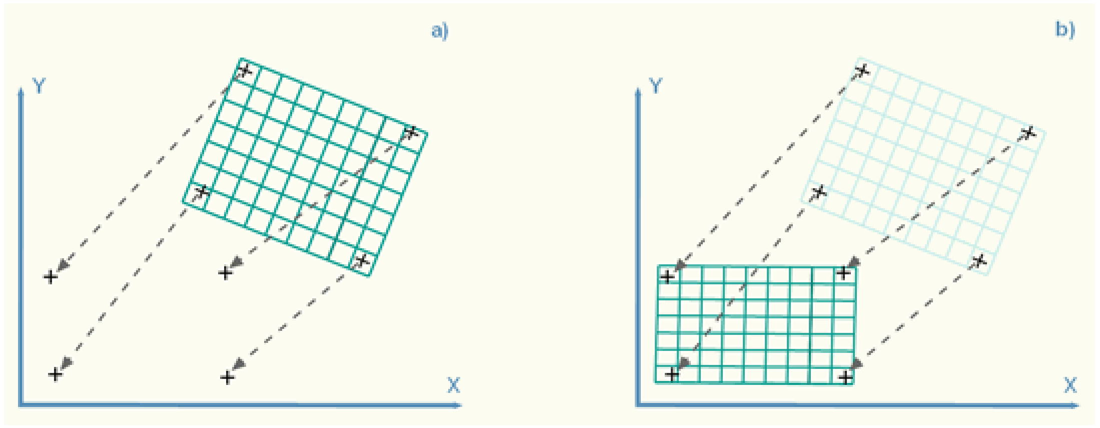


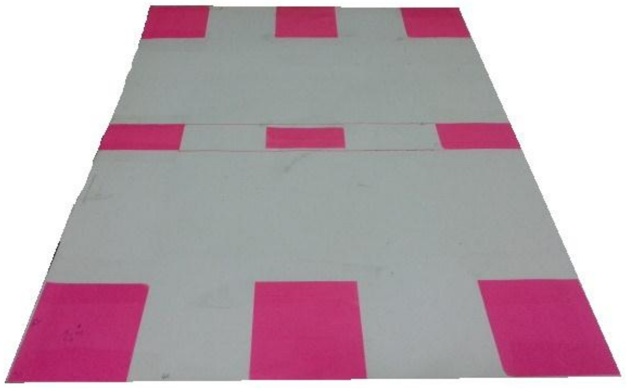
Figure 6 **a-** Geo-referenced Raster Image Coordinates of a 2D transformation with control points.; **b-** Adjusting georeferenced image to match the plane coordinate system.

During the polynomial transformation, the parameters are regularly calculated with Ground Control Points (GCPs). Such a step can be done through applying common points that are called tie points. Cases in points are the corners of houses or road intersections. Besides, two control points that help define the four parameters (a, b, x_0, y_0) of the ideal transformation are needed. The affine transformation requires three control points to determine the six parameters (a, b, c, d, x_0, y_0). Lastly, the 12 parameters (x_0, a_1 to a_5, y_0, b_1 to b_5) of a simple second-order polynomial transformation require six control points in order to be determined.

In the field of computer vision, any two points of the same planar surface in space are related by a mathematical relation. To find this relation we implement the following steps:

Step 1. Extraction of the featured points from the image and the surface (object points, image points): the featured points as shown in Figure 7 are recognized by specific color. Also, the location of each point should be known and find location of points in the image. In the new interface, object points are a vector of calibration pattern points vectors in the calibration pattern coordinate space. The outer vector has a similar number of elements to that of the pattern views. When in each view a similar calibration pattern is fully shown, , all the vectors will be the same. Moreover, partially or differently occluded patterns cab be used; accordingly, different vectors will result. The 3D points work in a patterned coordinate system. Accordingly, if the rig is planar, the model will be set to an XY coordinate plane; consequently, the Z-coordinate of each input object point will be 0. The image points in the new interface represent a vector of vectors of the projections of calibration pattern points. The image points represent the object points on the image as shown in Figure 2. The 3D object point's coordinates with their corresponding 2D projections in each view must be specified using an object of a known geometry and of easily detectable feature points.

Figure 7 sample of input image shows the reference points



Step 2. Computation of the coefficients parameter: the inputs for this step are (object points, image points). The computation module shown in Algorithm 1

Algorithm 1 computing the coefficients parameter

Input: Xp, Yp arrays of Image points and Xw,YW arrays of surface points
Output: CofX, CofY Coefficients parameter

Process:

```
For I = 0 to NoPoints   // NoPoints is the count number of points
  F[0] = 1;
  F[1] = Xp[I]
  F[2] = Yp[I]
  F[3] = Xp[I] x Xp[I]
  F[4] = Yp[I] x Yp[I]
  F[5] = Xp[I] x Yp[I]
      For J = 0 to 6
        For K = 0 to 6
```

```
          Ff[J, K] = Ff[J, K] + F[J] x F[K]
          U[J] = U[J] + F[J] x Xw[I]
          V[J] = V[J] + F[J] x Yw[I]
        End For K
      End For J
End For I
For J = 0 to 6
   For K = 0 to 6
      D[J, K] = Ff[J, K]
   End For K
End For J
Dd = Calculate Determinant of the array 6x6 (D)
For I = 0 to 6
    For J = 0 to 6
        For K = 0 to 6
        If Not  K equal  I
            D[J, K] = Ff[J, K]
        else
            D[J, K] = U[J]
        End For K
    End For J
  CofX[I] = Calculate Determinant of the array 6x6 (D) ÷ Dd
  End For I
For I = 0 to 6
    For J = 0 to 6
        For K = 0 to 6
        If Not  K equal  I
            D[J, K] = Ff[J, K]
        else
            D[J, K] = V[J]
        End For K
    End For J
  CofY[I] = Calculate Determinant of the array 6x6 (D) ÷ Dd
 End For I
Output: CofX[ ], CofY[ ]
```

Step 3. involves calculating the location transformation of an object: The polynomial transformation consists of two formulas: the first is used for computing the output x-coordinate for an input (x, y) location whereas the second is calculating the y-coordinate for an input (x, y) location.

$X' = CofX_0 + CofX_1 \times X + CofX_2 \times Y + CofX_3 \times X^2 + CofX_4 \times Y^2 + CofX_5 \times X \times Y$

$Y' = CofY_0 + CofY_1 \times X + CofY_2 \times Y + CofY_3 \times X^2 + CofY_4 \times Y^2 + CofY_5 \times X \times Y$

Where X' and Y' are the surface point of the target, X and Y are the image point of the target, CofX and CofY are the coefficient parameter for both coordinates.

2.2.4 Command Issues

Gripping and picking the objects is the primary task for many of the robotic arms. When we consider these tasks, it is very much necessary to achieve the relative motions smoothly without much vibration. Also these must be completely controlled in closed loop to get complete information of the position of arm at any moment.

To consider all the aspects, information from sensor like camera is collected and actuators are synchronized accordingly in the model.

A. Actuation scheme

Figure 3 displays how the arm of the robot moves in all direction as a five DOF. The movements are generated using stepper motors whereas the joints provide the ability to move at different angles. In addition, the timing belts, and the cable circuits are used to transfer the motion from the actuators to other parts of the arm. Timing belts and cable circuits provide a minimal contact between the geared wheels which has result zero backlashes. By using the coupling mentioned above, the arm is allowed to make little movements (less than 0.5 mm); consequently, no gearing will be damaged by any external forces.

When the scheme is connected to high torque of low speed stepper motors, low-cost with a relatively high performance actuation scheme will result. to the weak point of this scheme is that the reduction mechanisms occupy a relatively large volume; leading the proximal portion of the arm to be somewhat large.

The two-stage reduction scheme couples between the motions of joints 1 and 2, and joints 2, 3, and 4 in a linear way that can easily be calculated as a feed forward term in software. The paths of the timing belts and cables are shown in Figure 4.

B. Computing Steps

The process of counting the number of steps for each motor is the most important. The movement of the arm must take into account the degree of freedom of every joint and the length of each link in the arm. Therefore, when you calculate the angle required for each joint, the mathematical equations will be used to find the number of steps that required for each motor then issues the right command.

Generally speaking, the manipulator of the robot openly chained. Such a mechanism does not require 1 input for controlling the output (position, velocity, etc.) like that of a 3-bar linkage. As a result, the joints must be individually controlled. The number of degrees-of-freedom (DOF) of an arm depends on the type and number of joints. Generally, the joints of such manipulators permit 1 DOF and thus. Accordingly, the values of these degrees are equal to the number of the displacing links which are in return similar to the number of joints. A unique manipulator configuration can be determined by a set of unique joint values.

The angle computation of each joints depends on the length and the end position of the bar linkage. So each joint angle will be calculated individually using the following formula:

$$D = \sqrt{(X - X')^2 + (Y - Y')^2}$$
$$Z = \sqrt{(D' - D)^2}$$

Where D is the distance from the current position with reference to the end of the bar linkage to the target position, X and Y are the current point coordinates, X' and Y' are the target point coordinates. D' is the required distance to the target, Z is the azimuth.

However, the azimuth computation for each joint shall be done in sequence and it will apply repeatedly until the armdroid take the right position to reach the target.

Do

$X = L1 \times \cos(\theta 1 + stp1) + L2 \times \cos(\theta 2)$

$Y = L1 \times \sin(\theta 1 + stp1) + L2 \times \sin(\theta 2)$

$D_{11} = \sqrt{(X_2 - X')^2 + (Y_2 - Y')^2}$

$Z_{11} = \sqrt{(D' - D_2)^2}$

$X = L1 \times \cos(\theta 1 - stp1) + L2 \times \cos(\theta 2)$

$Y = L1 \times \sin(\theta 1 - stp1) + L2 \times \sin(\theta 2)$

$D_{12} = \sqrt{(X_2 - X')^2 + (Y_2 - Y')^2}$

$Z_{12} = \sqrt{(D' - D_2)^2}$

$X = L1 \times \cos(\theta 1) + L2 \times \cos(\theta 2 + stp2)$

$Y = L1 \times \sin(\theta 1) + L2 \times \sin(\theta 2 + stp2)$

$D_{21} = \sqrt{(X - X')^2 + (Y - Y')^2}$

$Z_{21} = \sqrt{(D' - D)^2}$

X = L1 × Cos (θ1) + L2 × Cos (θ2-stp2)

Y = L1 × Sin (θ1) + L2 × Sin (θ2- stp2)

$D_{22} = \sqrt{(X - X')^2 + (Y - Y')^2}$

$Z_{22} = \sqrt{(D' - D)^2}$

Where Z_{11} and Z_{12} are the azimuth with considering the L_1 ± step, Z_{21} and Z_{22} are the azimuth with considering the L_2 ± step.

After calculate the azimuth we apply the following steps to reach the final results:

```
min = Minimum ($Z_{11}$, $Z_{12}$, $Z_{21}$, $Z_{22}$)
if min = $Z_{11}$
increment θ1 by stp1
else if min = $Z_{12}$
Decrement θ1 by stp1
else if min = $Z_{21}$
 increment θ2 by stp2
else if min = $Z_{22}$
decrement θ2 by stp2
while (Min > target distance)
```

Above steps will repeat steps until reach its goal the issue the command to make the robot enter in a closed loop to perform the issued commands.

Performance: The main goal of this system is to grab the target object based on webcam sensing. The system should take the input image and analyze the target object then computes the right movement for each parts of the armdroid in a smooth way.

1. Initialization:

The initialization need a scene surface prepared before starting the system to detect the control point to compute the parameters of the 2nd order polynomial transformation. As shown in Figure 8.

Figure (8) a- the surface scene, b- the image scene.

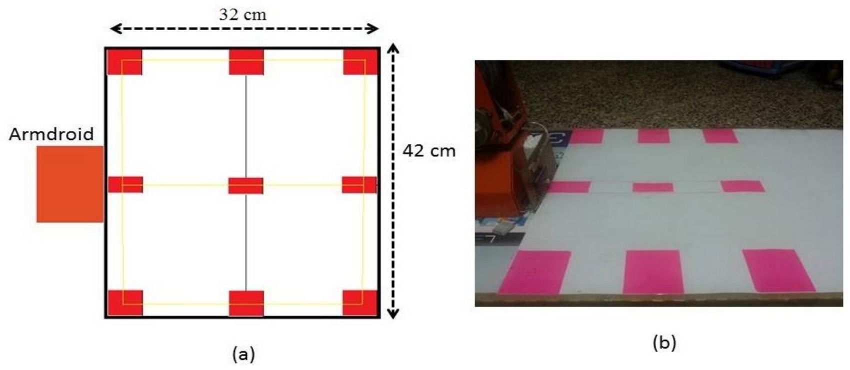
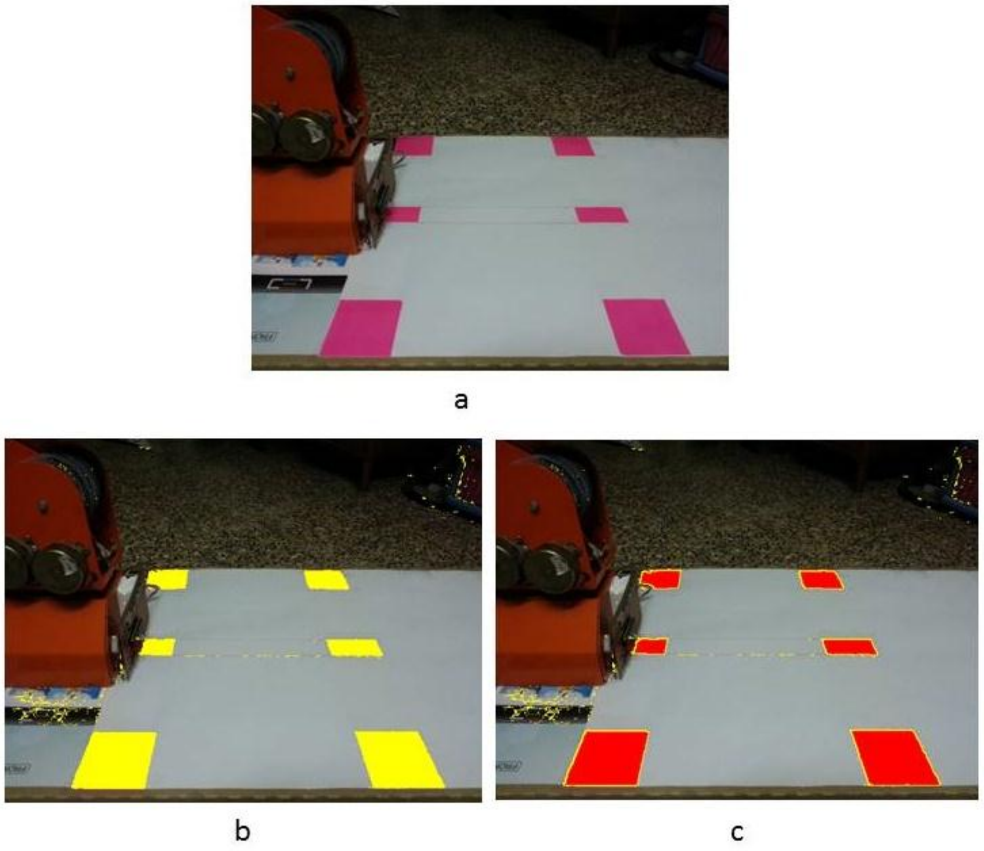


Figure 9 **a-** the reference image, **b-** detect the target color of the control point, **c-** the control points after



removing the noise color

Table 2 the coordinates of control points and image points based on Figure 9.

Points no.	X pixel	Y pixel	X world	Y world
1	191	139	0	0
2	311	138	16	0
3	245	139	32	0
4	160	243	0	21
5	310	244	16	21
6	468	245	32	21
7	105	420	0	42
8	311	419	16	42
9	530	418	32	42

Table 3 The coefficient Coordinates based on the points that shown in Table 2.

Points no.	Coefficient X Coordinate	Coefficient Y Coordinate
1	-33.153866132064	-36.9570059406925
2	0.165200726479013	-0.000481364505643916
3	0.0549760690886475	0.30669109766771
4	-1.06370815881277E-05	-1.20257600481543E-06
5	1.10391346697095E-05	-0.000283052608054303
6	-0.000201066475300002	3.4396504804095E-06

Table 4 error accuracy of the transformation from image points to the real world points in centimeter.

Points no.	Error X Coordinate	Error Y Coordinate
1	0.52	0.15
2	0.36	-0.14
3	-0.8	-0.01
4	-0.8	-0.11
5	-0.1	0.01
6	0.9	0.09
7	0.24	0.01
8	-0.03	0.03
9	-0.22	-0.04

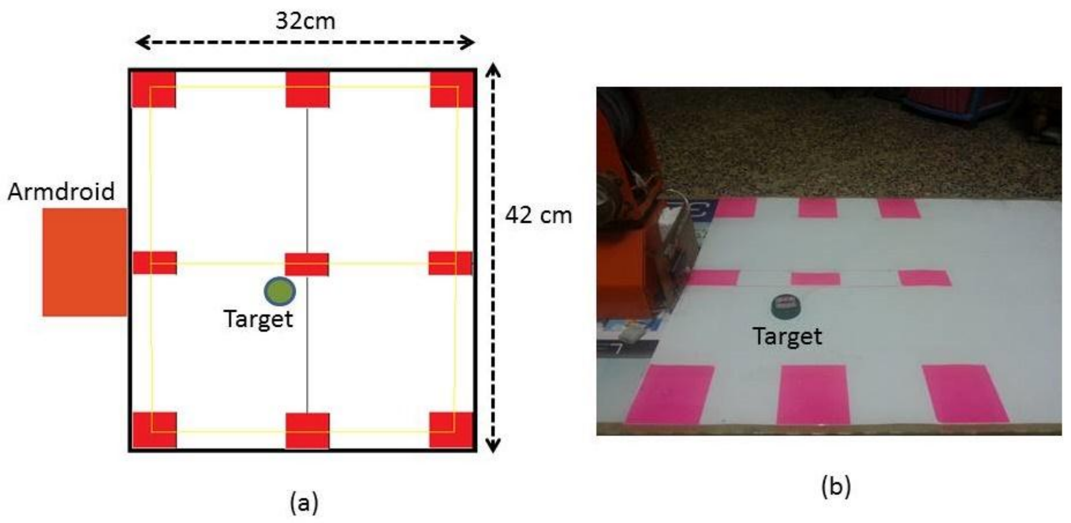


Figure 10 **a-** the target location in real world, **b-** the target location in image.

Table 11 The final result of the target.

X pixel	Y pixel	X world	Y world	distance	Angle
260	305	10.9	30.3	15cm	26

After calculate the distance and angle for the horizon of the target object. The next step is calculating the desired angle for each joint to compute the required steps for each link. The following Figure 11 is the demonstration of the Armdroid movement.

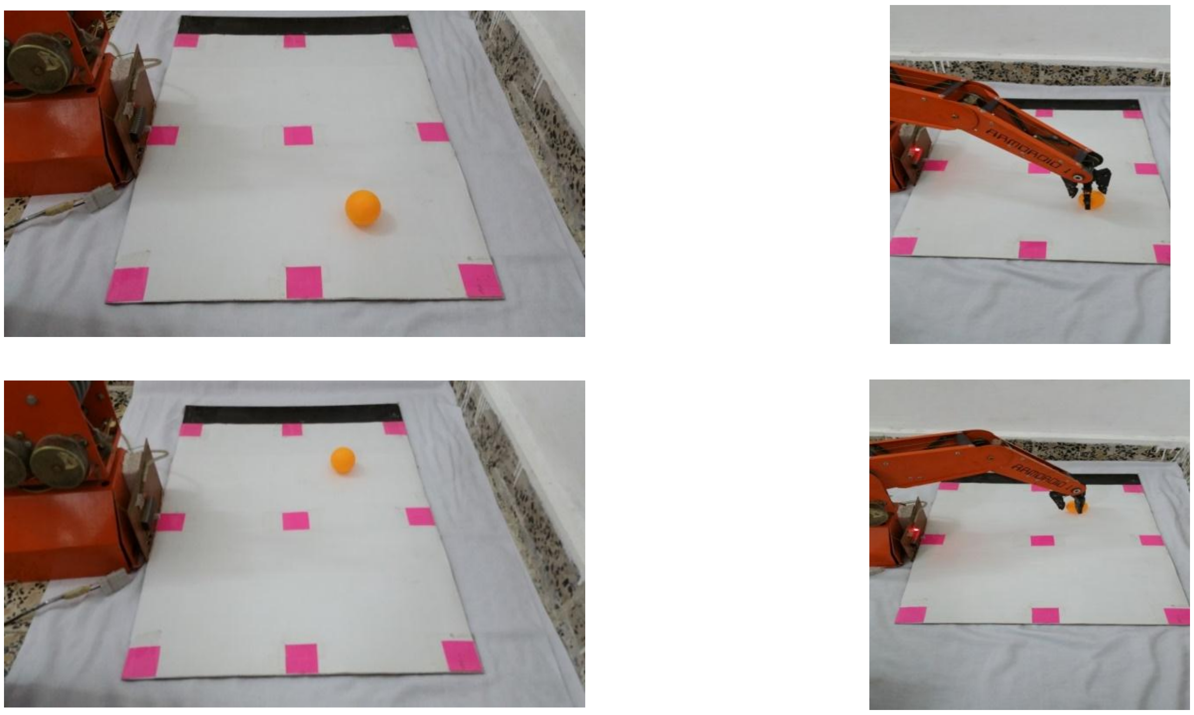
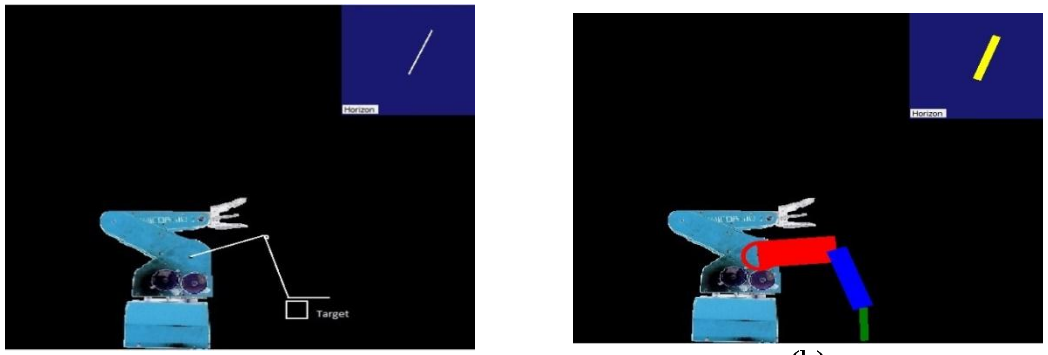


Figure 11 Demonstration of Armdroid movement



(a) (b)

Figure 12 **a-** the simulation shows the predicted move of the Armdroid. **B-** shows each step of the Armdoid in action (real-time).

3. Conclusion

In this paper, the design and implementation of an imitative robotic arm control system has all been presented in one program. Though the system works surprisingly well, yet nothing is perfect. The system detects target position, without the need of model training before using the system. The transformation of the target position can be approached from image to real world coordinates. Such a transformation helps achieve a high precision and recall rate with a blank background. It further shows acceptable robustness under the change of cluttered backgrounds and target-to-camera distances. The picking task based on the method used shows a smooth movement of the Armdroid to reach the target position wherever it is. Speaking about its enhancements, it can best be achieved by extending controlling the number of 3 to 5 joints together. Though the implementing solutions are endless, the basic idea involves using one or more cameras to detect the target position and dimension, in addition to knowing it using “blind” calculations. There are plenty of research opportunities for future work. Our approach can be extended to increase the degree of freedom in controlling two robotic arms simultaneously to automatically detect and pick a target based on camera sensor.

REFERENCES

[1] Kiyoshi Ohishi, M. O., T. M., "Motion Control of a Multijoint Robot Based on a robust Velocity Controller", Journal of Electrical Engineering in Japan, Vol. 118, No. 4, 1997.

[2] Keiichi Yamada, T. N., S. Y., "Robot Vision Robust to Change of Lighting Conditions Using a Wide-Dynamic Range Visual Sensor", Journal of Electrical Engineering in Japan, Vol. 120, No. 2, 1997.

[3] PMC, Smart Robots Industrial Development Promotion Plan, Precision Machinery Research Development Center, 2009.

[4] Chung, C.R., Introduction to Robotic Control-With BASIC Commander MCU as an Example, Taipei: Blue Ocean Cultural Publishing, 2011.

[5] Department of Mechanical Engineering, Mie University, Japan, Robotics—Definition of Robots (ISO), Advanced Course in Robotics, 2008.

[6] Chen, Y.R., Industrial Development and Promotion Plan for Digital Archives and Learning –Current Development and the Future of Learning Applications for Smart Robots, 2009.

[7] Alana L., "Robust Inexpensive Multi-Purpose Robotic Arm", Northeastern University, 2005.

[8] Alessandro G., Rossano C. and Marco T.,"A Self-Learning Multi Sensing Selection Process: Measuring Objects One by One by ARCES", LYRAS LAB University of Bologna, IEEE SENSORS Conference, 2007.

[10] Sahu S., Lenka P.; Kumari S.; Sahu K.B.; Mallick B., "Design a color sensor: Application to robot handling radiation work", Journal of Industrial. Engineering, Vol. 56, No.10, pp. 365-368, 2007.