VOL. 16, NO. 2, 2017, 44-46
www.fke.utm.my/elektrika
ISSN 0128-4428

ELEKTRIKA
Journal of Electrical Engineering

Intelligent Robot Arm Pick and Place using Hough transform vision algorithm

Kumeresan A. Danapalasingam* and **Mohamed Abbas Elamin**

Faculty of Electrical Engineering, Universiti Teknologi Malaysia, 81310 UTM Skudai, Johor, Malaysia.

*Corresponding author: kumeresan@fke.utm.my

Abstract: This paper proposes an improved algorithm for 4-degree of freedom robot arm forward/inverse kinematics with vision sensor fusion utilizing Hough transform for a pick and place application. As a result, an optimized approach for path planning and end-effector velocity generation is enabled. The main advantage of the algorithm is the simplicity of deriving the manipulator kinematic equations and generating solutions. This is achieved by deriving a common set of simultaneous equations that characterize the system for both forward/inverse kinematics, and subsequently solving numerically using Matlab symbolic toolbox. The solution incorporates input from vision sensor for object detection and tracking by implementing OpenCV vision libraries under Matlab, interfaced with a low cost microcontroller that controls the manipulator servos. The path planning approach involves segmentation of identified optimal path and stabilizing end-effector velocity by calculating corresponding joint angular velocity of each segment. To test the effectiveness of the proposed algorithm, Matlab simulations are conducted and preliminary results are produced for three components namely the kinematics algorithm, vision Hough Transform besides the optimal path planning approach. The superiority of this proposal is in the simplicity, compared to the widely used homogeneous matrix transformation that implements the cumbersome Denavit-Hartenberg convention for setting up the frame coordinate, also the proposed algorithm removes singularities and redundancies from the solution and confine the solution to the servos operating limits. It is envisioned that this algorithm will be efficient in the design of practical high performance, low cost robot arm in preparation for the ongoing industrial revolution.

Keywords: Robot Arm, DOF, Pick and Place , Kinematic, Path Planning, Vision , Hough Transform, OpenCV.

1. INTRODUCTION

Robot Arm kinematics and dynamics algorithms are diverse and of variant degree of complexity, in literature the inverse kinematic for instance could be approached in more than six methods namely (inverse transform, screw algebra, dual matrices, dual quaternion, iterative, Geometrics , … etc) on the other hand there isn't a method to combine the forward and inverse calculations in a single step to ease the incorporation of vision sensor.

To overcome this obstacle a universal description of the system is identified for both the inverse and forward kinematics in a set of simultaneous equations solved utilizing the powerful Matlab symbolic toolbox by passing the set of equations then assigning to solve for the angle variables in case of inverse kinematics or solving the same set of equations by returning the end-effector final Cartesian coordinates in case of forward kinematics.

The Hough transform vision algorithm that identify circular objects is implemented to identify and track the picked target using the OpenCV vision libraries that returns the radius and the center coordinates of the identified circles which is used to determine the initial end-effector coordinates, from which the path to the final place coordinate is planned through segmentation and solving for each segment using the same set of universal equations in an iteration that stores the path coordinates and respective angles in matrices.

2. SYSTEM CHARACTERIZATION

2.1 Forward/Inverse Kinematics computation

The layout of the system used to derive the characteristics equation is depicted in Figure 1. The length of each link is L,L1 & L3 respectively, the angles for the base and the two joints are θ, θ1 and θ2 respectively. θ1 and θ2 are measured from a reference line parallel to the x-y plane beside θ reference is taken from the initial base frame servo position.

The algorithm is summarized in the following Matlab command that solves for two variables either theta1 & theta2 for inverse kinematic or x,y & z for forward kinematics. the solution is limited for the angle values to eliminate redundancy and singularities.

solve([equation1,equation2,theta1>0,theta1<pi,theta2>0 ,theta2<pi],[theta1,theta2]);

The plot in Figure 2 gives an overview of the algorithm output

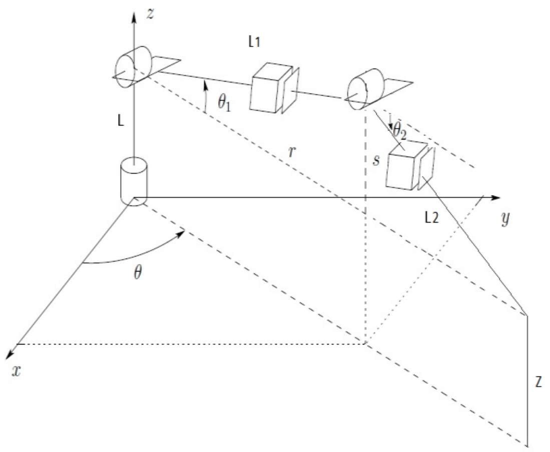

Figure 1. Robot Arm with 4-DOF description

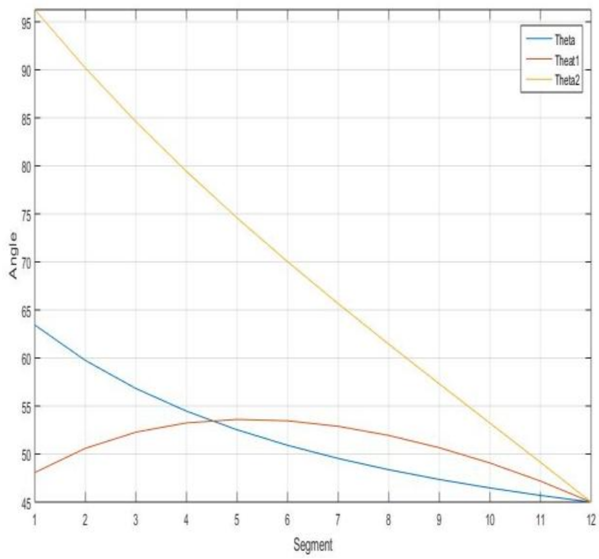

Figure 2. Angles generation

The parameters for the above scenario was as follows:

x1 = 0.25;
y1 = 0.5;
z1 = 0.75;
x2 = 1;
y2 = 1;
z2 = 1;
L = 1;
L1= 1;
L2= 1;

As seen all angles converge to 45º since all links length and target coordinate is set to unity.

2.2 Hough Transform vision algorithm

The Hough transform vision algorithm was implemented using the OpenCV 2.14 vision libraries that has built-in functions that could be called under Matlab utilizing the mex feature for external routines, which requires the installation and compilation of the libraries in Matlab to setup the operating environment. The algorithm is as follows:

circles = cv.HoughCircles (img,'DP',1.8, 'MinDist' ,550, 'MinRadius' ,60, 'MaxRadius',350)

The range of desired detected circles is input in the Min & Max Radius parameter and the separating distance between each circle is also defined in MinDist to avoid congestion, the DP parameter is a sensitivity gauge for the algorithm. Figure 3 shows an output for a random semi-circle object, as seen some smaller circles are excluded from the identification based on the specified radius range. The displayed coordinates are retrieved from the *circles* parameter which is a matrix that hold the radius and center Cartesian coordinate for every detected circle.

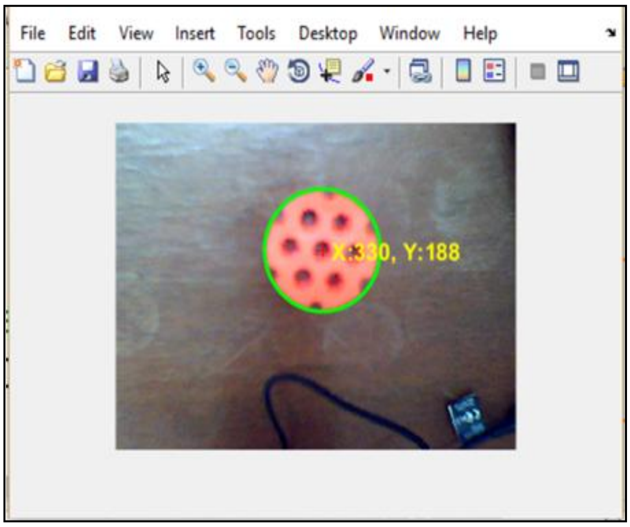

Figure 3. Hough Transform vision algorithm output

The initial pick coordinates could be retrieved from the vision algorithm without manually assigning, which provide higher accuracy in the picking operation to overcome the various parameters affecting including items misplacement, arm vibration, servos response …etc.

2.3 Path Planning generation

The shortest path between the initial point and final point is calculated using line segmentation technique to confine the desired path to the specific generated micro-points, an initial parameter for the number of segments is defined then a simple step incremental algorithm generate the path matrix, the inverse kinematic algorithm described in 2.1 is applied to each element in the matrix to acquire the respective angle to be applied to the servos through a low cost micro-controller. below is a the code snippet.

```
n = 11
for c = 2:n
   path{1,c} = [x1+(c-1)*((x2-x1)/n),y1+(c-1)*((y2-y1)/n),z1+(c-1)*((z2-z1)/n)];
end
```

The output of the above is given in Figure 4.

3. CONCLUSION

In this paper, a simplified and computationally economic approach is proposed for implementation in low cost microcontroller. it could easily accommodate higher degree of freedoms manipulators with diverse links and Joints arrangements. the path planning algorithm flexibility is applicable to any desired geometric path not necessarily being the shortest. On the other hand the vision algorithm could evolve to detect objects based on color or features giving a wide scope of applications.

REFERENCE

[1] Corke, Peter. *Robotics, vision and control: fundamental algorithms in MATLAB*. Vol. 73. Springer, 2011.

[2] Craig, John J. *Introduction to robotics: mechanics and control*. Vol. 3. Upper Saddle River: Pearson Prentice Hall, 2005.

[3] Spong, Mark W., Seth Hutchinson, and Mathukumalli Vidyasagar. *Robot modeling and control*. Vol. 3. New York: wiley, 2006.

[4] Siciliano, Bruno, and Oussama Khatib, eds. *Springer handbook of robotics*. Springer, 2016.

[5] Reddy, Ravitej, and S. R. Nagaraja. "Integration of robotic arm with vision system." *Computational Intelligence and Computing Research (ICCIC), 2014 IEEE International Conference on*. IEEE, 2014.

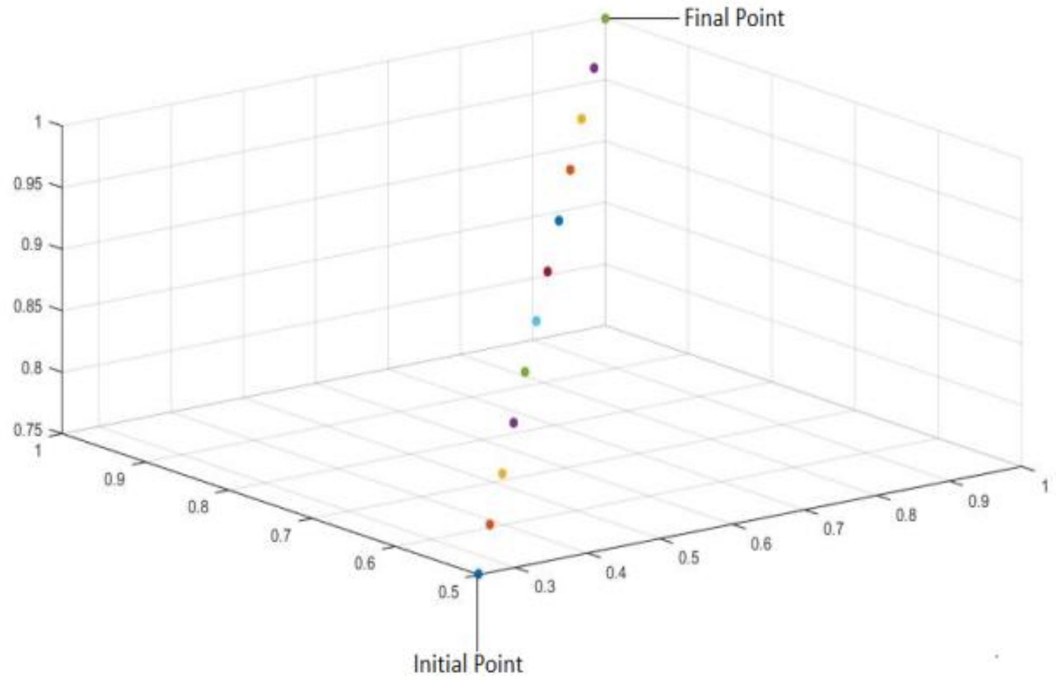

Figure 4. Path Planning generation