DOI: https://doi.org/10.33103/uot.ijccce.22.4.8

A Comparative Study of Graph Search Algorithms for Planar Manipulator to Avoid Obstacle Collision

Mustafa Laith Muhammed[1], Amjad Jaleel Humaidi[2], Enass Hassan Flaieh [3]
[1,3]*Mechanical Engineering Department, University of Technology, Baghdad, Iraq*
[2]*Control and Systems Engineering Department, University of Technology, Baghdad, Iraq*
[1]*me.19.08@grad.uotechnology.edu.iq,* [2]*Amjad.j.humaidi@uotechnology.edu.iq,*
[3]*Enass.h.flaieh@uotechnology.edu.iq*

Abstract— *The search algorithms are characterized by their ability to find the optimal path in a short calculation time. In this study, a comparative analysis has been conducted to perform path planning of planar manipulator for static obstacle avoidance based on graph search algorithms. Four methods have been taken into account to establish a comparison platform; namely, conventional A*, modified A*, Chaos A*, and circulation heuristic search (CHS) algorithms. The performance of comparison is evaluated in terms of length of optimal path and consumption time of calculation. All algorithms have been coded and simulated within the MATLAB software environment. According to computer simulation, the results showed that CHS algorithms outperform the other graph search ones in terms of generated path length, while the Choas A* could give the least calculation time as compared to its counterparts.*

Index Terms*— Path planning, A-Star, Modified A-Star, Chaos A-Star, CHS, Planar manipulator.*

I. INTRODUCTION

The aim of robot path planning is to determine how a robot will move and maneuver within a specific team in order to maintain its objectives. The path planning task requires the robot to generate a collision-free path between a start and a destination point in addition to avoiding obstacles. Furthermore, the robot must meet certain requirements or improve specific performance aspects. The type of path planning is influenced by the amount of information available about the environment (totally unknown, partially known, and completely known). Most of the time, the environment is only partially known, with the robot identifying certain regions inside the workplace before moving on to path planning. The obstacle is said to be static (dynamic) when its position and orientation stay stationary (change) relative to fixed reference frame over the time [1].

When information from sensors installed is continually gathered while the manipulator is moving, local path planning occurs. In this technique of path planning, the robot manipulator reacts instantly to changes in the environment and changes its orientation accordingly. When the robot's surroundings are well-known and static, the planning is known as global path planning. In this mode, the planning algorithm will generate the full path before the movement begins [2][3].

In this study, the main contribution can be listed below:

1. A several modifications in classical A* path planning method has been proposed by extending search area of adjacent nodes around current node or changing search techniques.
2. A comparison study has been conducted between the proposed modified methods with the conventional one.

Received 14/February/2022; Accepted 30/March/2022
 ISSN (Print) 1811-9212 *ISSN (Online) 2617-3352*

DOI: https://doi.org/10.33103/uot.ijccce.22.4.8

II. RELATED WORKS

A review of the most recent studies in the field of current work in order to determine the analysis strategies, Raheem A. and Abdul-Kareem A. introduced a new approach of manipulator path planning based on probabilistic roadmap and the artificial potential field methods and then used A* method to enhanced roadmap [4]. The study has applied Non-uniform B-spline (NURBS) curve for enhancing the generated path. Huang X. et al., used modified A-star algorithm to develop novel collision-free path planning for articulated space manipulator. The neural network algorithm has been used to ensure optimal time cost of path planning [5]. T. Nayl et al., presented a new approach to find smooth path planning with obstacles avoidance in fully known environment based on modified A-Star algorithm for articulated manipulator with obstacle avoidance based on data acquisition of local range sensors, which are mounted on the manipulator arms [6]. F. Li et al., presented an improved A-star algorithm based on collision-detection algorithm for optimal and collision-free path planning. This study has been applied to take a needle, fixed at the gripper of 6 DOF articulated robot, to inject a target point accurately and smoothly [7]. AL-Qassar A. and Abdulnabi A. utilized Bezier curve and A-star algorithm to achieve collision-free path planning for 5 DOF robot manipulator [8]. J. Silva et al**,** used graph search algorithm A* to implement path planning for pick-and-place operations in configuration free space with the presence of obstacles. The proposed method could successfully perform shortest path and avoid both joint limitations and obstacles for articulated manipulator [9]. P. Tavares et al, proposed new path planning approach for multiple robotic manipulators operating in the configuration space based on double A* algorithm by using multiple universal robot arm 5 (UR5) [10]. S. Gunawan etal, proposed modified A* algorithm to perform path planning for nonholonomic mobile robots. The proposed method showed smooth and continuous path planning in virtual environment [11]. F. Duchon et al, introduced modifications in the A-star algorithm for mobile robot path planning. The estimation time and the path optimality are the main modified index in this study, which has been applied in different scenarios with high complexity in working environment [12]. X. Li et al proposed a path planning approach in unknown environment by fusing the improved A-star algorithm with improved Dynamic Window Approach (DWA) algorithm. The hybrid methodology could address the drawbacks in the classical A-star algorithm like the capability to avoid obstacles and the use of numerous turning points [13]. X. Lan et al., have developed an improved path planning algorithm method by combing ant colony algorithm (ACA) with A-star algorithm to find local optimal path in mobile robot applications with static obstacles and complex environments [14]. T. Zheng et al., have improved the A-star algorithm in complex environment by firstly utilizing the cost function of angle evaluation to reduces the number of inflection points in the search path and secondly by using the jump search strategy to reduce the number of search nodes [15].

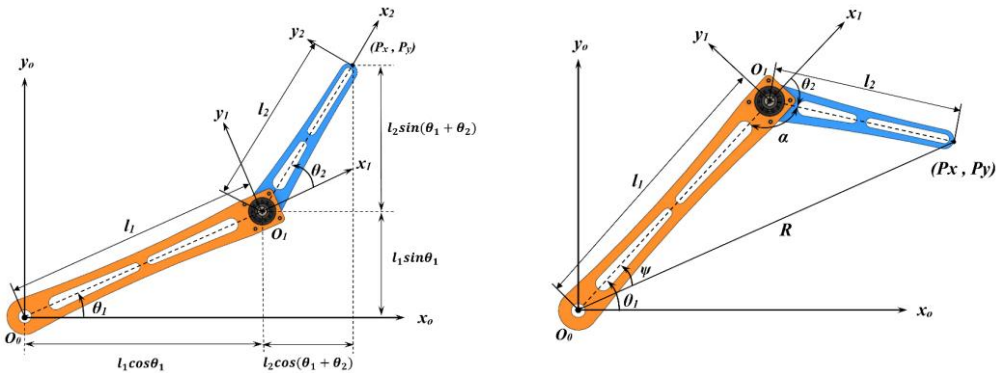
III. ROBOT MANIPULATOR MODELLING

The Kinematic analysis describes the analytical link between joint positions, position and orientation of end-effector in a Cartesian coordinates by neglecting the moments and forces producing the structural motion [16]. The Kinematic analysis can be classified into forward kinematics (F.K.) and inverse kinematics (I.K.).

The F.K. of robot manipulator is concerned with calculation of the position and orientation of end-effector frame relative to joint coordinates θ, while the I.K. focuses on finding specifies joint angles based on specified cartesian points of end-effector to obtain desired orientation and position of end-effector. *Fig. 1* shows the forward and inverse kinematics for planar manipulator [17][18].

Received 14/February/2022; Accepted 30/March/2022
 ISSN (Print) 1811-9212 ISSN (Online) 2617-3352

DOI: https://doi.org/10.33103/uot.ijccce.22.4.8



(A) (B)

FIG. 1. 2R PLANAR MANIPULATOR (A) ELBOW DOWN CONFIGURATION (B)ELBOW UP CONFIGURATION.

Based on elbow up configuration, one can deduce the cartesian positions as follows:

$$P_x = L_1.\cos\theta_1 + L_2.\cos(\theta_1+\theta_2) \quad (1)$$

$$P_y = L_1.\sin\theta_1 + L_2.\sin(\theta_1+\theta_2) \quad (2)$$

where L_1 and L_2 are the link lengths of manipulator. For elbow up configuration and based on geometric analysis, the I.K. can be obtained according to the following equations:

$$R^2 = P_x^2 + P_y^2 \quad (3)$$

Using the cosines law to have

$$R^2 = l_1^2 + l_2^2 - 2l_1l_2\cos\alpha \quad (4)$$

It easy to get α,

$$\alpha = cos^{-1}\left(\frac{l_1^2+l_2^2-R^2}{2l_1l_2}\right) \quad (5)$$

According to *Fig. 1*B, one can have

$$\theta_2 = \alpha - \pi \quad (6)$$

$$\theta_2 = cos^{-1}\left(\frac{l_1^2+l_2^2-R^2}{2l_1l_2}\right) - \pi \quad (7)$$

$$\psi = tan^{-1}\left(\frac{l_2\sin\theta_2}{l_1+l_2\cos\theta_2}\right) \quad (8)$$

$$\theta_1 = tan^{-1}\left(\frac{P_y}{P_x}\right) + tan^{-1}\left(\frac{l_2\sin\theta_2}{l_1+l_2\cos\theta_2}\right) \quad (9)$$

IV. CLASSIC A* ALGORITHM

The A-Star algorithm is heuristic method which is devoted to search feasible and optimal path in certain environment. The algorithm divides the space into several cells. The evaluation of path length is based on cost function which can be defined by:

$$f(n) = g(n) + h(n) \quad (10)$$

where g (n) denotes the actual distance from the current cell (node) to the start point, h (n) define the heuristic distance from the current cell to the destination point, and f (n) represents the total path distance from the start point to destination point via selected sequence of cells [17].

The classic A* searches for a cell in the space grid for the optimal way to go from a start point to a destination point. In *Fig. 2*, the A-star algorithm is applied to eight neighborhoods nodes around a current node.

Received 14/February/2022; Accepted 30/March/2022

ISSN (Print) 1811-9212 ISSN (Online) 2617-3352

DOI: https://doi.org/10.33103/uot.ijccce.22.4.8

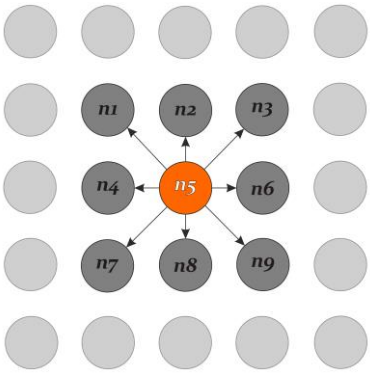


FIG. 2. A* ALGORITHM SEARCHING DIAGRAM.

In order to move the robot arm from the start point to the goal point in the presence of obstacles, a sequence of joint angles along the path have to be determined [19]. The problem of finding a feasible collision free path, from start to goal, can be solved by applying A* algorithm, a number nodes within the acceptable area of the workspace will be found by Eq. 10, and used to generate the path. The A* path planning algorithm to move the arm through the chosen nodes to reach the desired goal point is illustrated via the Pseudo Code listed in *Fig. 3*.

```
Classic_A_star_WSA ()
Begin
 //Workspace Analysis Generation:
       Set Workspace Parameter;
       Set Manipulator Parameter;
       Initiate Workspace;
       Do
          For i=1 to All Point in Workspace, do
               Calculate:Inverse_Kinematic_Elbow_Up
          End For (i)
       Read Number, Shape, Size, Position of Obstacles;
       Identify Acceptable Points;
       Identify Forbidden Points;
       While (All Points in Workspace are Analyzed)
            IF Manipulator Collide the Obstacles, then
                      Repeat Process
            ELSE
                      Write: Free Workspace Analysis
             End IF
 //Path Planning Generation:
       Set Search Space Parameter;
       Set Open_List = (), Close_List = ();
       Insert (Start_Node, Open_List);
       Do
       Current_Node = Start_Node;
       Determine Eight Neighbor_Node;
       Insert (Neighbor_Nodes, Open_List);
            For n=1 to Neighbor_Nodes, do
                 F(n) = G(n) + H(n)
            End For (n)
       Remove (min_cost_Node, Open_List);
       Insert (min_cost_Node, Close_List);
            IF the Node is the Target_Node, then
                 Write: The Path from Targrt_Node
                          to Start_Node
            ELSE
                 Find Node Successor that NOT in
                 Close_List, put in Open_List and
                 calculate the cost functions.
            End IF
       While (All Nodes are Analyzed)
End
```

FIG. 3 THE PSEUDO CODE TO EXECUTE PATH PLANNED OF 2 DOF MANIPULATOR BASED ON A* ALGORITHM.

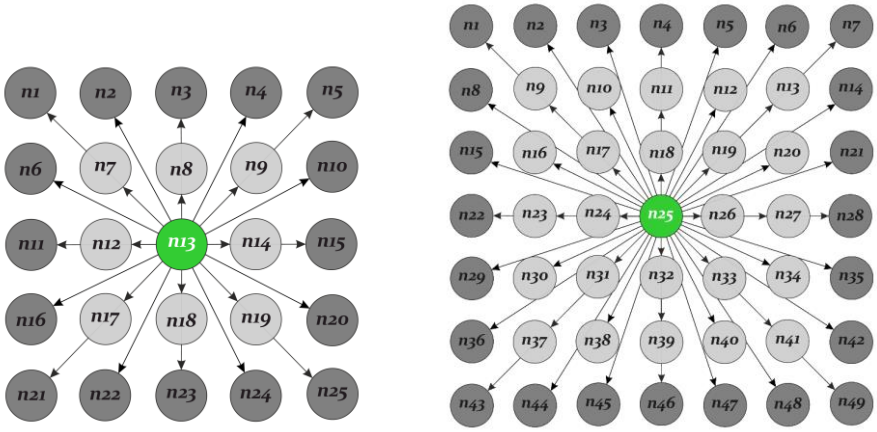
V. MODIFIED A* ALGORITHM

In the search based on standard A* algorithm, the work space grid is limited to 8 adjacent nodes surrounding the current node as illustrated in *Fig. 2*. Since the generated path is based on linking the closet possible nodes, this will lead to zigzag style path and hence this is not quite desirable due to sharp edges in finding optimal path.

Received 14/February/2022; Accepted 30/March/2022

ISSN (Print) 1811-9212 *ISSN (Online) 2617-3352*

DOI: https://doi.org/10.33103/uot.ijccce.22.4.8



(A) (B)

FIG. 4. MODIFIED A* ALGORITHM SEARCHING DIAGRAM (A) 24 NEIGHBORHOOD NODES. (B) 48 NEIGHBORHOOD NODES.

The modification of classic A* algorithm is to extend searching neighborhoods of the current node. Instead of using searching nodes of 8-nodes, an extension of 24-nodes around the current node has been proposed. Of course, this will reduce the sharp edges as indicated in *Fig. 4*. However, the evaluation function is calculated in the same principle as in Eq. 10. For path generation, the usage of the node from outer or inner sets is performed based on the presence of the obstacles or the rational calculation of each node, see *Fig. 4*A. The Pseudo code which represents the steps of modified A* algorithm taking into account the planar manipulator and obstacle avoidance is developed in *Fig. 5*:

```
Modified_A_star_WSA ()
Begin
//Workspace Analysis Generation:
    Set Workspace Parameter;
    Set Manipulator Parameter;
    Initiate Workspace;
    Do
      For i=1 to All Point in Workspace, do
          Calculate: Inverse_Kinematic_Elbow_Up
      End For (i)
    Read Number, Shape, Size, Position of Obstacles;
    Identify Acceptable Points;
    Identify Forbidden Points;
    While (All Points in Workspace are Analyzed)
      IF Manipulator Collide the Obstacles, then
            Repeat Process
      ELSE
            Write: Free Workspace Analysis
      End IF
//Path Planning Generation:
    Set Search Space Parameter;
    Set Search Space Parameter;
    Set Open_List = (), Close_List = ();
    Insert (Start_Node, Open_List);
    Do
    Current_Node = Start_Node;
    Read Neighbor_Nodes_Matrix_Size;
    Determine n^th Neighbor_Nodes;
    Insert (Neighbor_Nodes, Open_List);
      For n=1 to Neighbor_Nodes, do
          F(n) = G(n) + H(n)
      End For (n)
    Remove (min_cost_Node, Open_List);
    Insert (min_cost_Node, Close_List);
      IF the Node is the Target_Node, then
          Write: The Path from Targrt_Node to
          Start_Node
      ELSE
          Find Node Successor that NOT in
          Close_List, put in Open_List and
          calculate the cost functions.
      End IF
    While (All Nodes are Analyzed)
End
```

FIG. 5. THE PSEUDO CODE TO EXECUTE PATH PLANNED OF 2 DOF MANIPULATOR BASED ON MODIFIED A* ALGORITHM.

VI. CHAOS A* ALGORITHM

The necessity behind developing such a modification is either to increase the smoothness of the generated path or to reduce the time needed to reach the destination node, or both, as explained in the previous modification. Because of the search is based on the standard A* algorithm, the work space grid is limited to 8 adjacent nodes surrounding the current node, as shown in *Fig. 2*, according to that it

Received 14/February/2022; Accepted 30/March/2022

ISSN (Print) 1811-9212 ISSN (Online) 2617-3352

DOI: https://doi.org/10.33103/uot.ijccce.22.4.8

take much longer time to reach destination node, so it necessary to develop an algorithm reach destination node with significant less estimation time.

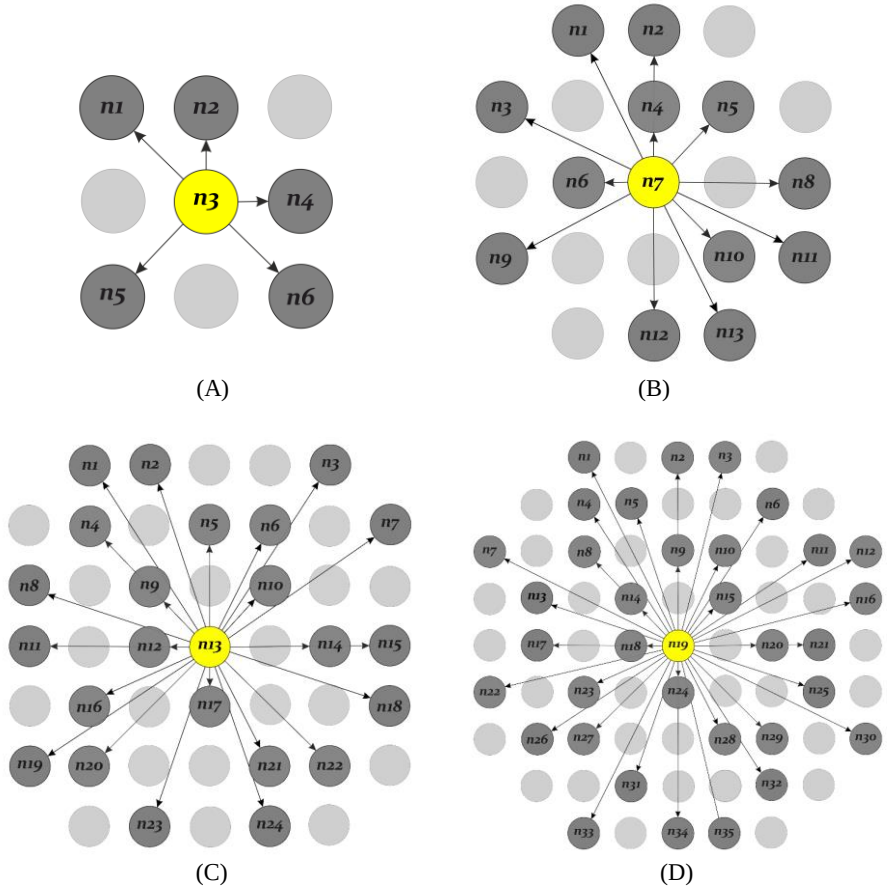


FIG. 6. CA* ALGORITHM SEARCHING DIAGRAM (A) 5 RANDOM NEIGHBORHOOD NODES. (B) 12 RANDOM NEIGHBORHOOD NODES. (C) 23 RANDOM NEIGHBORHOOD NODES. (D) 34 RANDOM NEIGHBORHOOD NODES.

The Chaos A-star (CA*) algorithm is also a modification of the classic A* algorithm which extend searching neighborhoods of the current node. The expansion strategy takes the structure of an octagon with a specific radius to distribute neighboring nodes, but in a random distribution manner; for example, if the number of adjacent nodes is 48, only 23 of them are randomly distributed as shown in *Fig. 6*C. The objective of this modification is to minimize the time necessary to search for the destination node or the transition time from the start node to the end node in the presence of obstacles by more than 50% when compared to previous modified algorithms with the same number of search nodes. However, the evaluation function is calculated in the same principle as in Eq. 10. For path generation, the selection of the node in the searching process is performed based on the presence of the obstacles or the rational calculation of each node, see *Fig. 6*B, C, D. The Pseudo code which represents the steps of Chaos A* algorithm is developed in *Fig. 7*.

Received 14/February/2022; Accepted 30/March/2022

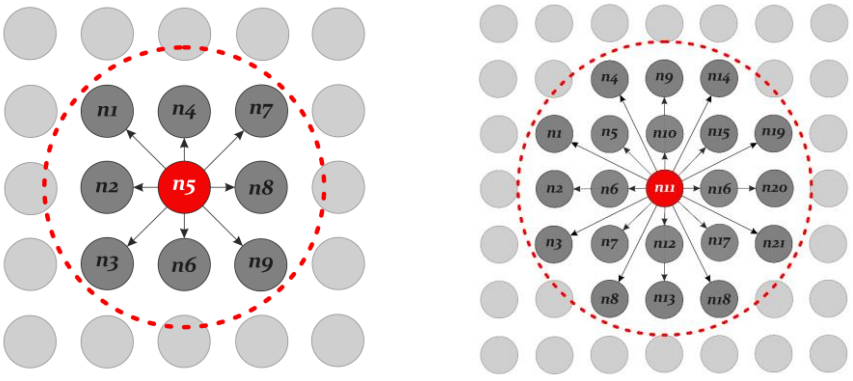
DOI: https://doi.org/10.33103/uot.ijccce.22.4.8

```
Chaos_A_star_WSA ()
Begin
//Workspace Analysis Generation:
    Set Workspace Parameter;
    Set Manipulator Parameter;
    Initiate Workspace;
    Do
      For i=1 to All Point in Workspace, do
       Calculate: Inverse_Kinematic_Elbow_Up
      End For (i)
    Read Number, Shape, Size, Position of Obstacles;
    Identify Acceptable Points;
    Identify Forbidden Points;
    While (All Points in Workspace are Analyzed)
      IF Manipulator Collide the Obstacles, then
            Repeat Process
      ELSE
            Write: Free Workspace Analysis
      End IF
//Path Planning Generation:
   Set Search Space Parameter;
    Set Search Space Parameter;
    Set Open_List = (), Close_List = ();
    Insert (Start_Node, Open_List);
    Do
    Current_Node = Start_Node;
    Read Search Radius of Octagon from
    Current_Node;
    Determine n^th Neighbor_Nodes Randomly;
    Insert (Neighbor_Nodes, Open_List);
      For n=1 to Neighbor_Nodes, do
            F(n) = G(n) + H(n)
      End For (n)
    Remove (min_cost_Node, Open_List);
    Insert (min_cost_Node, Close_List);
      IF the Node is the Target_Node, then
            Write: The Path from Targrt_Node to
            Start_Node
      ELSE
            Find Node Successor that NOT in
            Close_List, put in Open_List and
            calculate the cost functions.
      End IF
    While (All Nodes are Analyzed)
End
```

FIG. 7. CHAOS A* ALGORITHM PSEUDO CODE.

VII. CIRCULATION HEURISTIC SEARCH

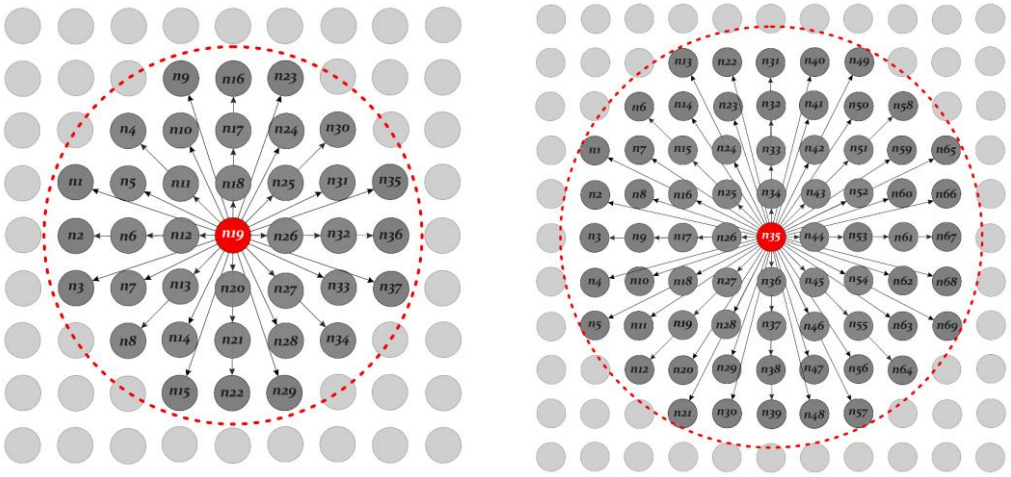
The Circulation Heuristic Search (CHS) is a modification of standard A* algorithm which extend searching neighborhoods of the current node. Instead of using searching nodes of 8-nodes, the expansion strategy uses deploying all nodes inside a circle whose center is the current node, and its diameter is determined in advance based on the unit of measurement used in the work space. Of course, this will reduce the sharp edges as indicated in *Fig. 8*.



(A) (B)

Received 14/February/2022; Accepted 30/March/2022

DOI: https://doi.org/10.33103/uot.ijccce.22.4.8



(C) (D)

FIG. 8. CHS ALGORITHM SEARCHING DIAGRAM (A) 2-UNIT RADIUS (B) 3-UNIT RADIUS (C) 4-UNIT RADIUS (D) 5-UNIT RADIUS.

However, the evaluation function is calculated in the same principle as in Eq. 10. For path generation, the selection of the node inside the searching circle is performed based on the presence of the obstacles or the rational calculation of each node, see *Fig. 8*B, C, D. The Pseudo code which represents the steps of CHS algorithm taking into account the planar manipulator and obstacle avoidance is developed in *Fig. 9*.

```
Circulation_Heuristic_Search_WSA ( )
Begin
//Workspace Analysis Generation:
      Set Workspace Parameter;
      Set Manipulator Parameter;
      Initiate Workspace;
      Do
         For i=1 to All Point in Workspace, do
          Calculate: Inverse_Kinematic_Elbow_Up
         End For (i)
      Read Number, Shape, Size, Position of Obstacles;
      Identify Acceptable Points;
      Identify Forbidden Points;
      While (All Points in Workspace are Analyzed)
         IF Manipulator Collide the Obstacles, then
                  Repeat Process
         ELSE
                  Write: Free Workspace Analysis
         End IF
//Path Planning Generation:
     Set Search Space Parameter;
      Set Search Space Parameter;
      Set Open_List = (), Close_List = ();
      Insert (Start_Node, Open_List);
      Do
      Current_Node = Start_Node;
      Read Search Circle Size from Current_Node;
      Find all the Nodes inside the search Circle and
         Consider them as Neighbor_Nodes of
         Current_Node;
     Insert (Neighbor_Nodes, Open_List);
        For n=1 to Neighbor_Nodes, do
                F(n) = G(n) + H(n)
        End For (n)
     Remove (min_cost_Node, Open_List);
     Insert (min_cost_Node, Close_List);
        IF the Node is the Target_Node, then
                Write: The Path from Targrt_Node to
                Start_Node
        ELSE
                 Find Node Successor that NOT in
                 Close_List, put in Open_List and
                 calculate the cost functions.
        End IF
       While (All Nodes are Analyzed)
End
```

FIG. 9. THE PSEUDO CODE TO EXECUTE PATH PLANNED OF 2 DOF MANIPULATOR BASED ON CHS ALGORITHM.

VIII. SIMULATION RESUTLS

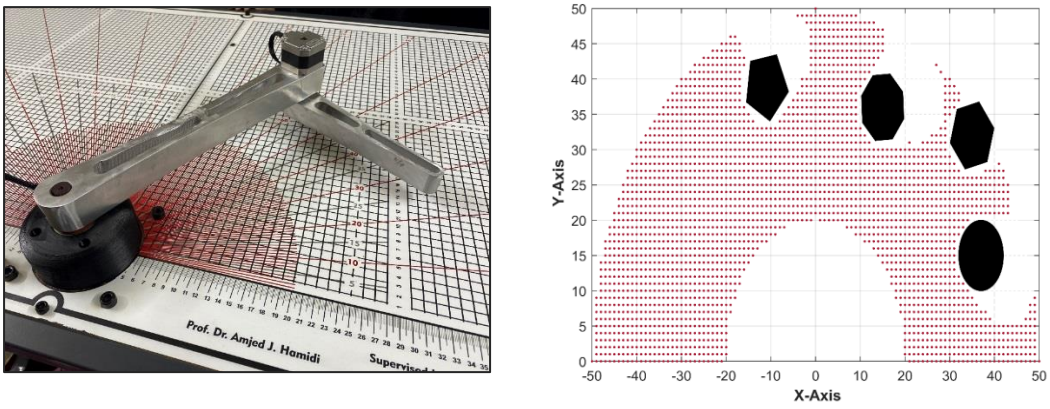
In this study, the up-elbow configuration is addressed and the generated path planning algorithms consists of three parts. The first part is the workspace in Cartesian space. The workspace is defined as a space made of all points that can only be reached by a specified end-effector configuration. Inverse kinematics has been used to obtain these points which are related with joint angles. As illustrated in

Received 14/February/2022; Accepted 30/March/2022

ISSN (Print) 1811-9212 *ISSN (Online) 2617-3352*

DOI: https://doi.org/10.33103/uot.ijccce.22.4.8

Fig. 10, the generation of free cartesian space is limited by mechanical and geometric constraints, which affect and limit the motion of the robotic manipulator as well as split the workspace into *acceptable* and *forbidden* zones. The generation of free cartesian space can be achieved by analyzing all possible solutions for *acceptable* points in the environment, which are dependent on the obstacles collision checking function as shown in *Fig. 10*B [20].



(A) (B)

FIG. 10. (A) MANIPULATOR LAYOUT (B) WORK SPACE ANALYSIS WITH DIFFERENT OBSTACLE SHAPE BASED ON ELBOW UP MANIPULATOR.

The workspace has a dimension of (50 to 50) cm in the x-axis and (0 to 50) cm in the y-axis. The length of the manipulator links is 30 cm for link 1 and 20 cm for link 2. The obstacle type is static and its arrangement in the workspace has two cases for each proposed method. The cases are composed of four obstacles, each one of various shapes with a diameter of 10 cm and coordinates as shown in Table I.

TABLE I. OBSTACLES COORDINATES FOR THE PROPOSED PATH PLANNING ALGORITHMS

Configuration	1st-Obstacle		2nd-Obstacle		3rd-Obstacle		4th-Obstacle	
	X (cm)	Y (cm)	X (cm)	Y (cm)	X (cm)	Y (cm)	X (cm)	Y (cm)
Case 1	35	18	7	35	-13	35	-30	20
Case 2	41	10	7	40	-16	33	-	-

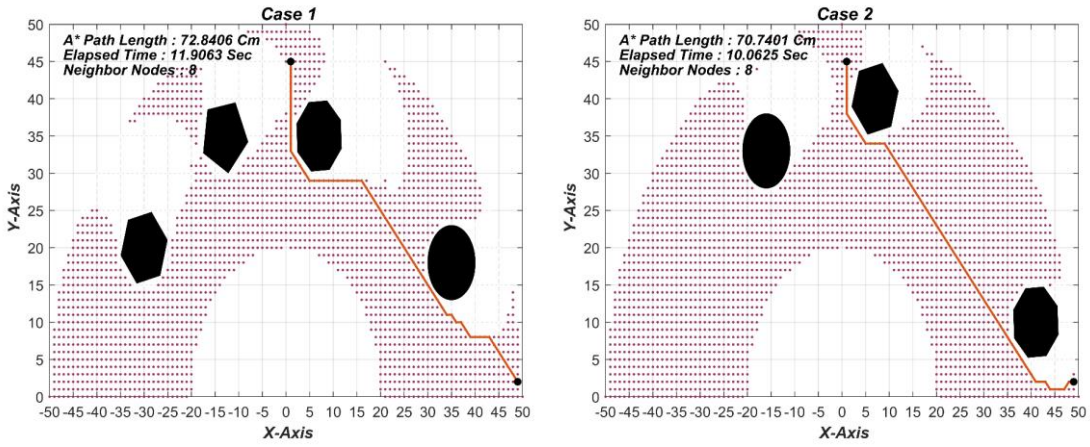
The second part of the proposed path planning algorithms is the workspace analysis for obstacle avoidance. This part is the same as the obstacle-free space excluding all the points which make contact with obstacle area. In other words, this part includes all points of workspace where there is no collision of manipulator's links with resident obstacles during the path planning from start to destination points. Accordingly, one can detect three forbidden regions; one is due to allowable lengths of arms (outer region), the second area is due to presence of obstacle, while the third area (inner region) is due to singularity and mismatch in length of the first and second arms. The latter area has a radius equal to the length of second link as shown in *Fig. 10*B.

The third part of path planning is the path planning algorithm. This process is restricted only to an acceptable area. The algorithms have to find the shortest path from start to destination point within the acceptable area in a shortest time. *Fig. 11*, shows the implementation of two cases of obstacle configurations based on A* algorithm.

Received 14/February/2022; Accepted 30/March/2022

ISSN (Print) 1811-9212 ISSN (Online) 2617-3352

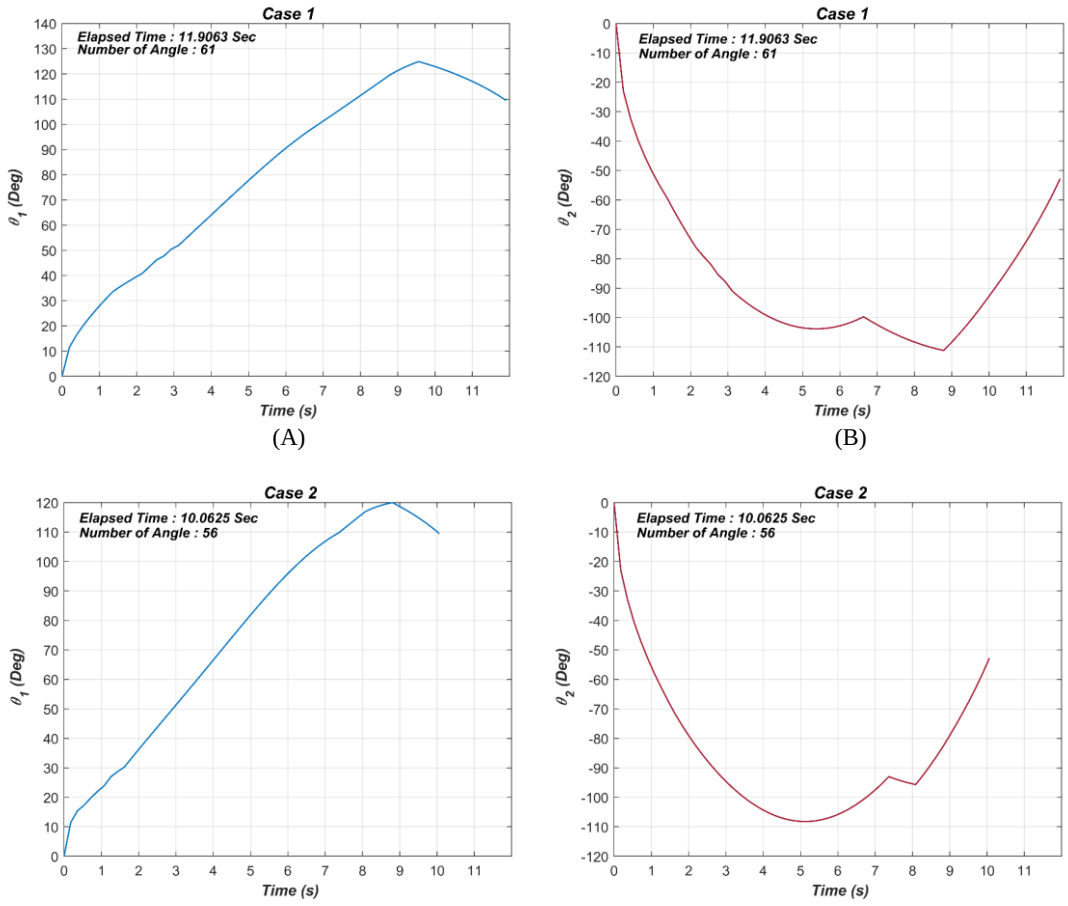
DOI: https://doi.org/10.33103/uot.ijccce.22.4.8



(A) (B)

FIG. 11. A* PATH PLANNING WITH TWO OBSTACLE CONFIGURATIONS: (A) CASE 1, (B) CASE 2.

As shown in *Fig. 11*, the generated paths are due to searching in 8 neighbor nodes around the current node, where in case 1 the total length of the path from start to destination point is 72.8406 cm, in total estimated time 11.9063 sec, whereas in case 2, the total length of the path is 70.7401 cm, in total estimated time 10.0625 sec. The variation in length between the two cases is due to the location change and shape of the obstacles. *Fig. 12*, shows the variation of the joint's angles with respect to time of manipulator movement from start to destination for both cases.



(A) (B)

(C) (D)

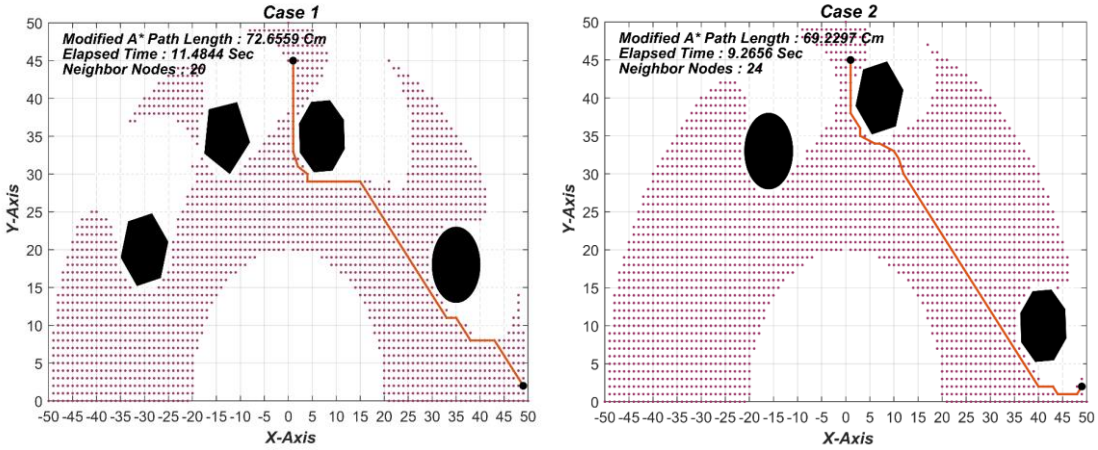
FIG. 12. A* PATH PLANNING JOINT VARIATION $\boldsymbol{\theta_1}$ (A , C) AND $\boldsymbol{\theta_2}$ (B , D) FOR BOTH CASES.

Received 14/February/2022; Accepted 30/March/2022

ISSN (Print) 1811-9212 *ISSN (Online) 2617-3352*

DOI: https://doi.org/10.33103/uot.ijccce.22.4.8

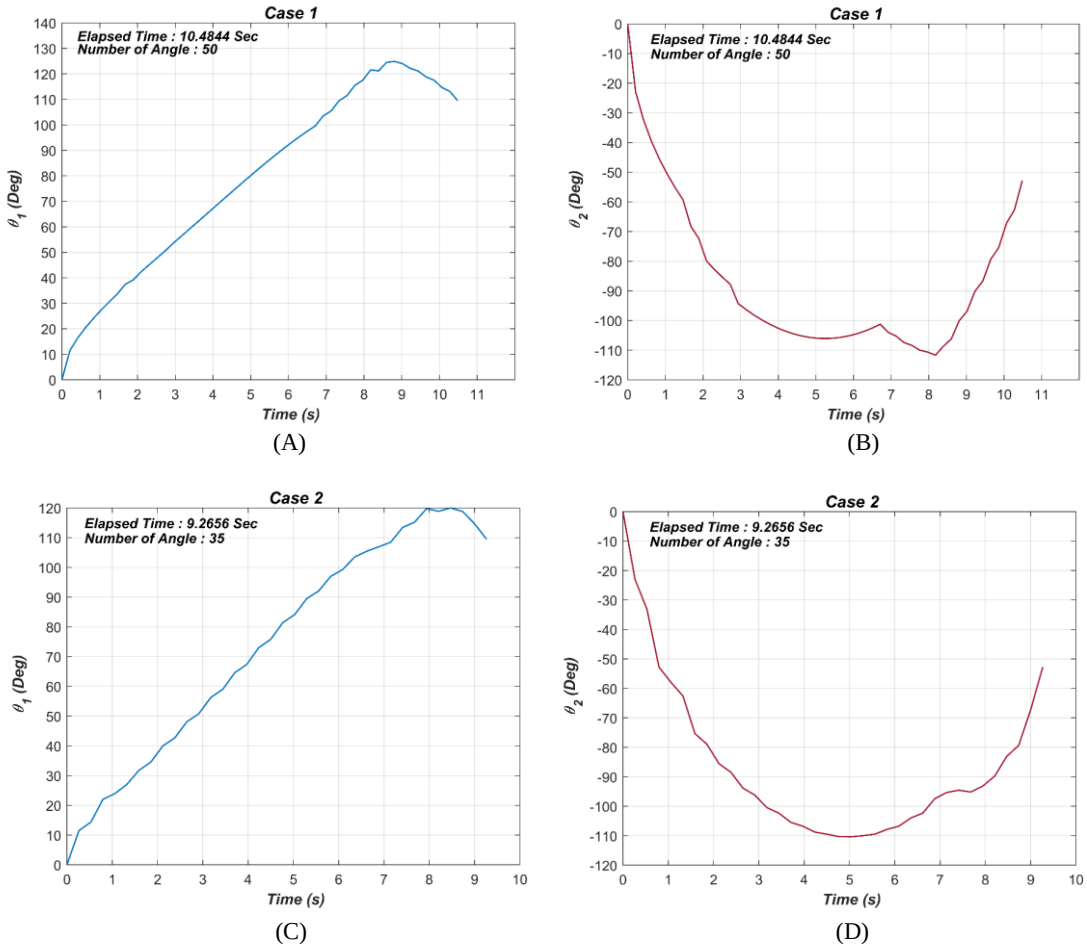
In the next scenario, the Modified A* path planning is considered as shown in *Fig. 13*. for two cases. In this study, 24 neighbor nodes are used for searching purposes. In case 1, the total length of path, from start to destination point, is equal to 72.6559 cm, which is done in estimated time 11.4844 sec. In case 2, the total length of the path is 69.2297 cm with estimated time equal to 9.2656 sec. The variation in length and time between the two cases is due to the change in obstacle locations and shapes.



(A) (B)

FIG. 13. MODIFIED A* PATH PLANNING: (A) CASE 1, (B) CASE 2.

Fig. 14 shows the variation of the joint's angles for both cases.



(A) (B)

(C) (D)

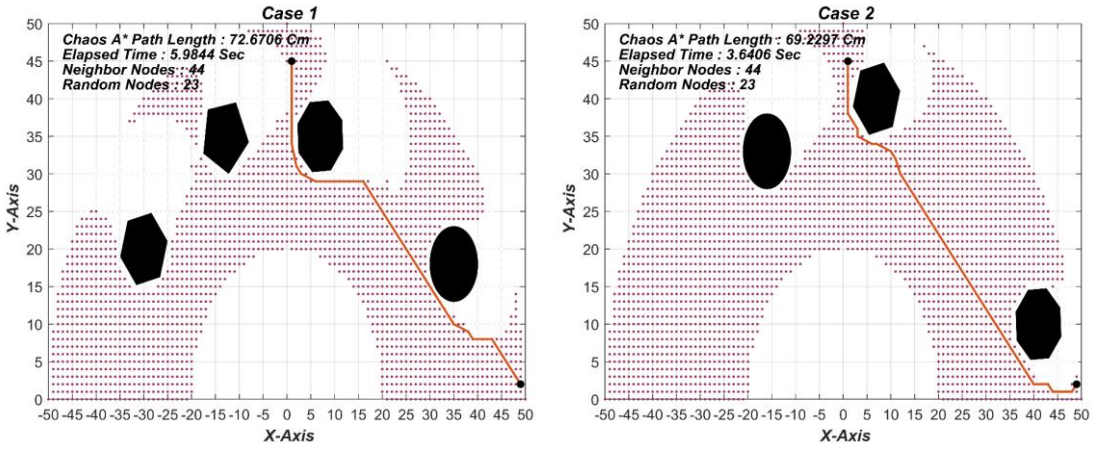
FIG. 14. MODIFIED A* PATH PLANNING JOINT VARIATION $\boldsymbol{\theta_1}$ (A , C) AND $\boldsymbol{\theta_2}$ (B , D) FOR BOTH CASES.

Received 14/February/2022; Accepted 30/March/2022

ISSN (Print) 1811-9212 ISSN (Online) 2617-3352

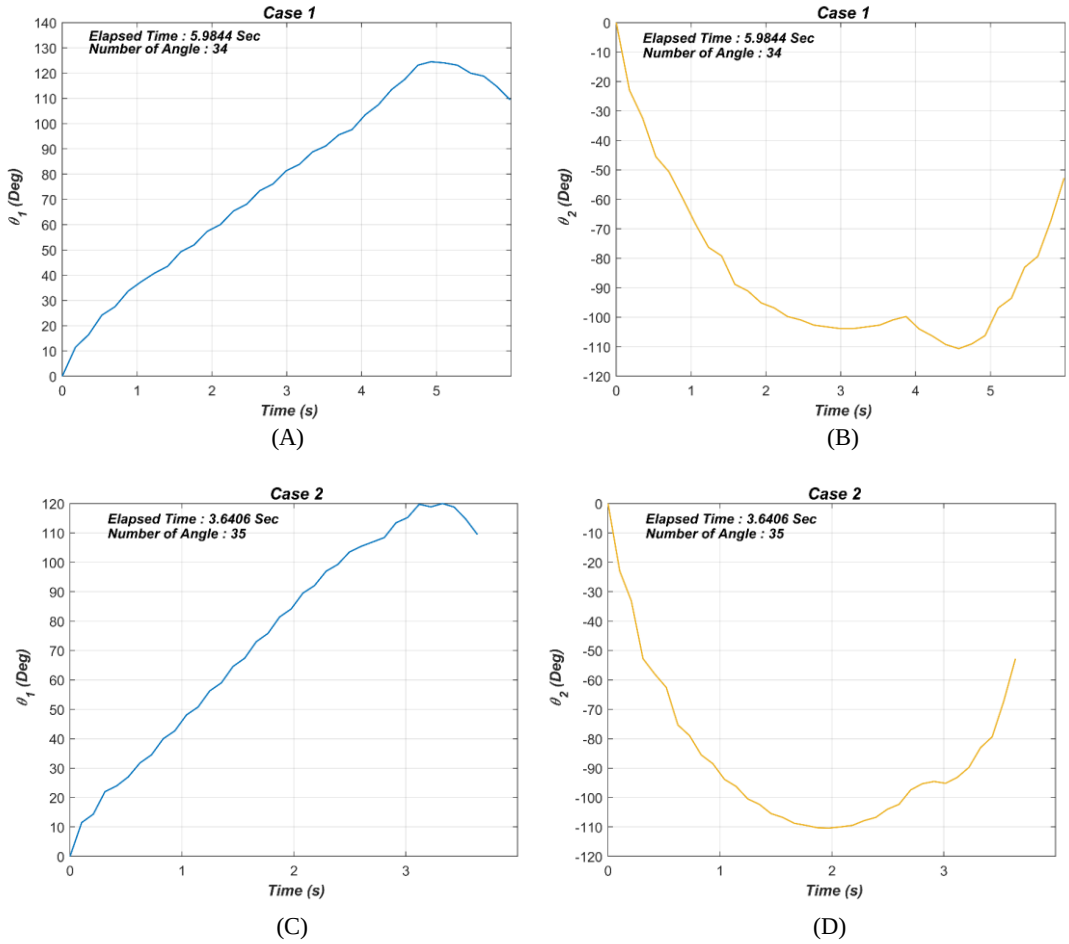
DOI: https://doi.org/10.33103/uot.ijccce.22.4.8

Now, the chaos A* path planning is applied as shown in *Fig. 15*, for two cases. In this study, 23 random nodes of 44 neighbor nodes are used for searching purposes. In case 1, the total length of path, is equal to 72.6706 cm, which is done in estimated time 5.9844 sec. In case 2, the total length of the path is 69.2297 cm with estimated time equal to 3.6406 sec. *Fig. 16*, shows the variation of the joint's angles for both cases.



(A) (B)

FIG. 15. CHAOS A* PATH PLANNING: (A) CASE 1, (B) CASE 2.



(A) (B)

(C) (D)

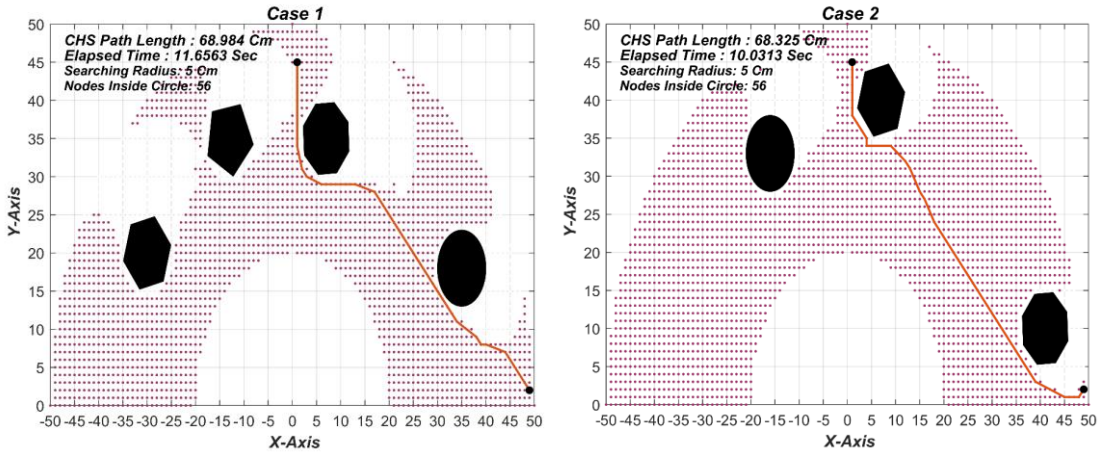
FIG. 16. CHAOS A* PATH PLANNING JOINT VARIATION $\boldsymbol{\theta_1}$ (A , C) AND $\boldsymbol{\theta_2}$ (B , D) FOR BOTH CASES.

Received 14/February/2022; Accepted 30/March/2022

ISSN (Print) 1811-9212 *ISSN (Online) 2617-3352*

DOI: https://doi.org/10.33103/uot.ijccce.22.4.8

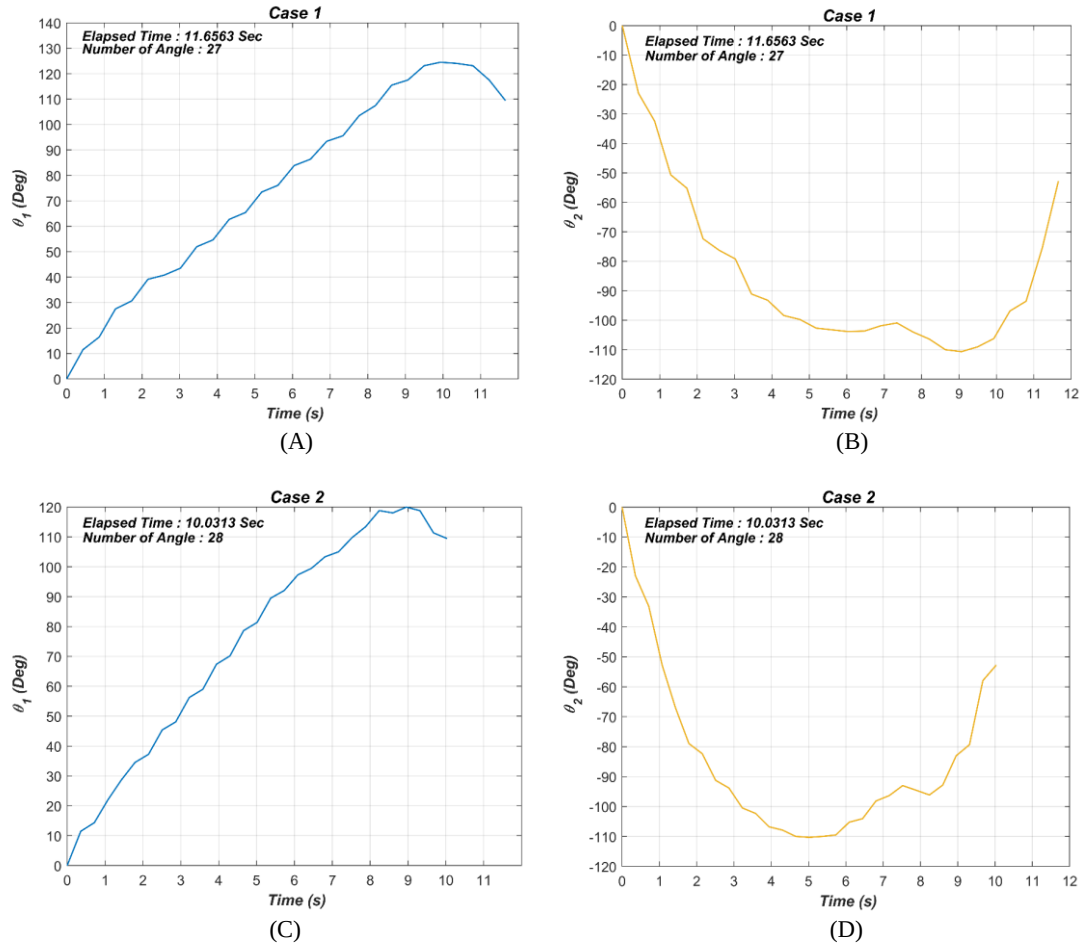
As illustrated in *Fig. 17*, the CHS path planning is evaluated for two scenarios. In this work, a search circle with a radius of 5 units and 56 neighbor nodes inside it is employed. In Case 1, the entire length of the path from start to target node is 68.984 cm, which is accomplished in an estimated time of 11.6563 sec. In Case 2, the entire length of the path is 68.325 cm, with an estimated time of 10.0313 sec.



(A) (B)

FIG. 17. CHS PATH PLANNING: (A) CASE 1, (B) CASE 2.

Fig. 18, depicts the fluctuation of the joint angles in both circumstances.



(A) (B)

(C) (D)

FIG. 18. CHS PATH PLANNING JOINT VARIATION $\boldsymbol{\theta_1}$ (A , C) AND $\boldsymbol{\theta_2}$ (B , D) FOR BOTH CASES.

Received 14/February/2022; Accepted 30/March/2022

ISSN (Print) 1811-9212 ISSN (Online) 2617-3352

DOI: https://doi.org/10.33103/uot.ijccce.22.4.8

Based on simulated results, the length and the time taken by proposed path planning algorithms are listed in Table II. The table indicates that the length of path based on CHS algorithm is shorter than that based on A* algorithm and other modifications for both cases. But, in term of estimation time, the chaos A* algorithm beats the other modified algorithms including the classic A* algorithm.

TABLE II. COMPARISON OF THE PROPOSED PATH PLANNING ALGORITHMS

Algorithms	Case 1				Case 2			
	Length (cm)	Improvement (%)	Time (sec)	Improvement (%)	Length (cm)	Improvement (%)	Time (sec)	Improvement (%)
A* Path Planning	72.8406	0	11.9063	0	70.7401	0	10.0625	0
Modified A* Path Planning	72.6559	0.25	11.4844	3.67	69.2297	2.18	9.2656	8.60
Chaos A* Path Planning	72.6706	0.23	5.9844	98.95	69.2297	2.18	3.6406	176.39
CHS Path Planning	68.984	5.59	11.6563	2.14	68.325	3.53	10.0313	0.31

The block diagram describing the signal flow between PC and the robot manipulator is shown in *Fig. 19*. The computer hosts the path which is generated by A* algorithm. The path is in cartesian space and it converted joint steps according to samples required movements. The samples of angular positions are fed to micro-stepping motors via Arduino-Uno microcontroller. Due to low power of output signals for microcontroller, the drive is necessary to actuate the motor for desired step angular position.

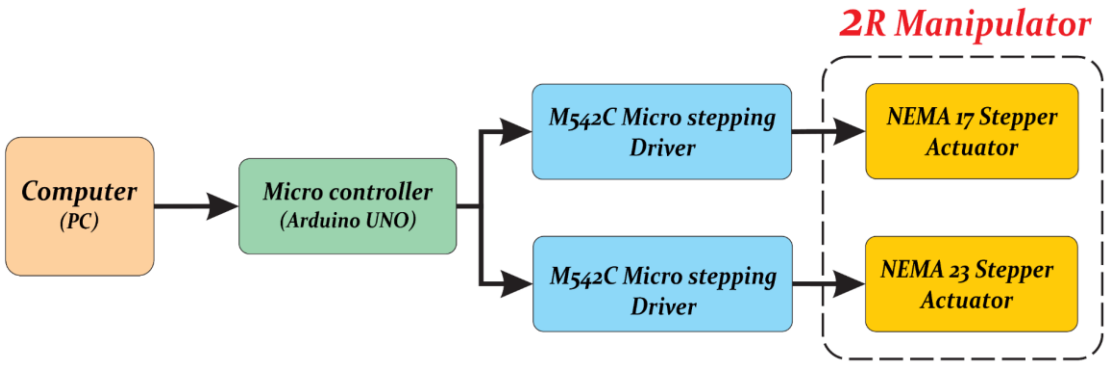


FIG. 19. BLOCK DIAGRAM OF REAL-TIME IMPLEMENTATION FOR PATH PLANNING ALGORITHMS.

Other hardware embedded design can be applied to implement the optimization algorithms in real-time environment. The Arduino microcontroller used in this work can be replaced by other single-board computers like Raspberry-Pi, or semiconductor devices like Field Programming Gate Array (FPGA) to implement the scenarios of proposed path planning techniques for 2R manipulator. It has been shown that has new hardware technologies can enhance the time cost to large extent [21]–[24].

This study can be extended for future work to include the dynamic model of the manipulator and to apply advanced control techniques in implementation of path planning methods[25]–[31]. Another update of this work is to in-cooperate modern optimization techniques in finding the optimal path in the presence of obstacles. One may propose recent optimization algorithms like particle swarm optimization (PSO), social spider optimization (SSO), Whale optimization algorithm (WOA), Butterfly optimization algorithm (BOA), Grey-wolf optimization (GWO) [32]–[36].

Received 14/February/2022; Accepted 30/March/2022

ISSN (Print) 1811-9212 ISSN (Online) 2617-3352

DOI: https://doi.org/10.33103/uot.ijccce.22.4.8

IX. CONCLUSIONS

This work analyzes path planning techniques for 2R planar manipulators in the presence of static obstacles in a known environment as well as their implementation. The classic A*, modified A*, chaos A*, and CHS algorithms have been proposed as path-planning algorithms. All techniques have been evaluated in a MATLAB simulation environment. The results reveal that the path generated by the CHS algorithm is shorter than the paths generated by the other approaches. In addition, the chaos A* algorithm could execute the robot gripper in significantly less time than its counterparts.

REFERENCE

[1] S. G. Tzafestas, *Introduction to Mobile Robot Control*, 1st. Elsevier Inc., 2014.

[2] M. Elbanhawi and M. Simic, "Sampling-Based Robot Motion Planning : A Review," *IEEE Access*, vol. 2, pp. 56–77, 2014, doi: 10.1109/ACCESS.2014.2302442.

[3] V. Kunchev, L. Jain, and V. Ivancevic, "Path Planning and Obstacle Avoidance for Autonomous Mobile Robots : A Review," in *International Conference on Knowledge-Based and Intelligent Information and Engineering Systems*, 2006, pp. 537–544, doi: https://doi.org/10.1007/11893004_70.

[4] F. A. Raheem and M. I. Abdulkareem, "Development of A* algorithm for robot path planning based on modified probabilistic roadmap and artificial potential field," *J. Eng. Sci. Technol.*, vol. 15, no. 5, pp. 3034–3054, 2020.

[5] X. Huang, Q. Jia, and G. Chen, "Collision-free Path Planning Method with Learning Ability for Space Manipulator," *12th IEEE Conf. Ind. Electron. Appl.*, 2017.

[6] T. Nayl, M. Q. Mohammed, and S. Q. Muhamed, "Obstacles Avoidance for an Articulated Robot Using Modified Smooth Path Planning," in *2017 International Conference on Computer and Applications, ICCA 2017*, 2017, pp. 1–5, doi: 10.1109/COMAPP.2017.8079732.

[7] F. Li, Z. Huang, and L. Xu, "Path planning of 6-DOF venipuncture robot arm based on improved a-star and collision detection algorithms," *IEEE Int. Conf. Robot. Biomimetics, ROBIO 2019*, 2019.

[8] A. N. A. Arif A. AL-Qassar, "Obstacle Avoidance Techniques for Robot Path Planning," *J. Eng. Sci.*, 2019.

[9] J. Sousa e Silva, P. Costa, and J. Lima, "Manipulator Path Planning for Pick-and-Place Operations with Obstacles Avoidance: An A* Algorithm Approach," in *Communications in Computer and Information Science*, 2013, vol. 371, pp. 213–224, doi: 10.1007/978-3-642-39223-8_20.

[10] A. P. M. Pedro Tavares, Jose Lima, Pedro Costa, "Multiple Manipulators Path Planning using Double A*," *Ind. Robot An Int. J.*, 2016.

[11] S. A. Gunawan, G. N. P. Pratama, A. I. Cahyadi, and B. Winduratna, "Smoothed A-star Algorithm for Nonholonomic Mobile Robot Path Planning," in *2019 International Conference on Information and Communications Technology (ICOIACT)*, 2019, pp. 654–658, doi: 10.1109/ICOIACT46704.2019.8938467.

[12] T. Frantisek Duchon, Andrej Babinec, Martin Kajan, Peter Beno, Martin Florek and L. J. Fico, "Path planning with modified A star algorithm for a mobile robot," *Procedia Eng. Ltd.*, 2014.

[13] X. Li, X. Hu, Z. Wang, and Z. Du, "Path planning based on combinaion of improved A-STAR Algorithm and DWA algorithm," in *Proceedings - 2020 2nd International Conference on Artificial Intelligence and Advanced Manufacture, AIAM 2020*, 2020, no. 1, pp. 99–103, doi: 10.1109/AIAM50918.2020.00025.

[14] X. Z. Xing Lan , Xiafu Lv , Wang Liu , Yi He, "Research on Robot Path Planning Based on Improved Ant Colony Algorithm under Computer Background," in *2021 IEEE 5th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, 2021, pp. 613–617, doi: 10.1088/1742-6596/1992/3/032050.

[15] T. Zheng, Y. Xu, and D. Zheng, "AGV Path Planning based on Improved A-star Algorithm," in *IEEE 3rd Advanced Information Management,Communicates,Electronic and Automation Control Conference (IMCEC)*, 2019, pp. 1534–1538, doi: 10.1109/IMCEC46724.2019.8983841.

[16] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics : Modelling, Planning and Control*, 1st. Springer-Verlag London Limited, 2009.

[17] K. M. Lynch and F. C. Park, *"Modern Robotics Mechanics , Planning , and Control,"* 1st. Cambridge University Press, 2017.

[18] A. B. Rehiara, *Kinematics of AdeptThree Robot Arm*. Saga University, Japan: intechopen, 2011.

[19] V. Hlaváč, "Manipulator trajectory planning," *Czech Tech. Univ. Pragu*, p. 39, 2008.

[20] F. A. Raheem, A. T. Sadiq, and N. A. F. Abbas, "Robot Arm Free Cartesian Space Analysis for Heuristic Path Planning Enhancement," *Int. J. Mech. Mechatronics Eng.*, vol. 19, no. 1, pp. 29–42, 2019.

[21] A. J. Humaidi and T. M. Kadhim, "Spiking versus traditional neural networks for character recognition on FPGA platform," *J. Telecommun. Electron. Comput. Eng.*, vol. 10, no. 3, pp. 109–115, 2018.

[22] T. M. Kadhim, S. Hasan, I. Kasim Ibraheem, and A. Taher Azar, "A Generic Izhikevich-Modelled FPGA-Realized Architecture: A Case Study of Printed English Letter Recognition," in *2020 24th*

Received 14/February/2022; Accepted 30/March/2022
 ISSN (Print) 1811-9212 *ISSN (Online) 2617-3352*

DOI: https://doi.org/10.33103/uot.ijccce.22.4.8

International Conference on System Theory, Control and Computing, ICSTCC 2020 - Proceedings, 2020, no. November, pp. 825–830, doi: 10.1109/ICSTCC50638.2020.9259707.

[23] A. S. M. Al-Obaidi, A. Al-Qassar, A. R. Nasser, A. Alkhayyat, A. J. Humaidi, and I. K. Ibraheem, "Embedded design and implementation of mobile robot for surveillance applications," *Indones. J. Sci. Technol.*, vol. 6, no. 2, pp. 427–440, 2021, doi: 10.17509/ijost.v2i2.

[24] M. Q. Kasim, R. F. Hassan, A. I. Abdulkareem, A. R. Nasser, and A. Alkhayyat, "Control Algorithm of Five-Level Asymmetric Stacked Converter Based on Xilinx System Generator," in *2021 IEEE 9th Conference on Systems, Process and Control (ICSPC 2021)*, 2021, pp. 174–179, doi: 10.1109/icspc53359.2021.9689173.

[25] H. J. Amjad and A. H. Hameed, "Design and comparative study of advanced adaptive control schemes for position control of electronic throttle valve," *Information, MDPI*, vol. 10, no. 2, pp. 1–14, 2019, doi: 10.3390/info10020065.

[26] E. N. Talaat, M. R. Hameed, and A. H. Hameed, "Design of Adaptive Observer-Based Backstepping Control of Cart-Pole Pendulum System," in *Proceedings of 2019 3rd IEEE International Conference on Electrical, Computer and Communication Technologies, ICECCT 2019*, 2019, pp. 1–5, doi: 10.1109/ICECCT.2019.8869179.

[27] A. H. Hameed, A. Q. Al-Dujaili, A. J. Humaidi, and H. A. Hussein, "Design of terminal sliding position control for electronic throttle valve system: A performance comparative study," *Int. Rev. Autom. Control*, vol. 12, no. 5, pp. 251–260, 2019, doi: 10.15866/ireaco.v12i5.16556.

[28] M. R. Hameed, "Design and Performance Investigation of Block- Backstepping Algorithms for Ball and Arc S ystem," in *2017 IEEE International Conference on Power, Control, Signals and Instrumentation Engineering (ICPCSI)*, 2017, pp. 325–332, doi: 10.1109/ICPCSI.2017.8392309.

[29] H. Jaleel, A. I. Abdulkareem, and W. Zhang, "Design of augmented nonlinear PD controller of Delta/Par4-like robot," *J. Control Sci. Eng.*, vol. 2019, pp. 11–22, 2019, doi: 10.1155/2019/7689673.

[30] A. J. and A. H. Hameed, "Robustness enhancement of MRAC using modification techniques for speed control of three phase induction motor," *J. Electr. Syst.*, vol. 13, no. 4, pp. 723–741, 2017.

[31] A. H. Jaleel and H. A. Hussein, "Adaptive Control of Parallel Manipulator in Cartesian Space," in *Proceedings of 2019 3rd IEEE International Conference on Electrical, Computer and Communication Technologies, ICECCT 2019*, 2019, pp. 1–8, doi: 10.1109/ICECCT.2019.8869257.

[32] T. Ghanim, A. R. Ajel, and A. J. Humaidi, "Optimal Fuzzy Logic Control for Temperature Control Based on Social Spider Optimization," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 745, no. 1, 2020, doi: 10.1088/1757-899X/745/1/012099.

[33] A. A. Al-Qassar *et al.*, "Finite-time control of wing-rock motion for delta wing aircraft based on whale-optimization algorithm," *Indones. J. Sci. Technol.*, vol. 6, no. 3, pp. 441–456, 2021, doi: 10.17509/ijost.v6i3.37922.

[34] A. A. Al-Qassar *et al.*, "Grey-Wolf optimization better enhances the dynamic performance of roll motion for tail-sitter VTOL aircraft guided and controlled by STSMC," *J. Eng. Sci. Technol.*, vol. 16, no. 3, pp. 1932–1950, 2021.

[35] S. K. Kadhim and A. S. Gataa, "Optimal Adaptive Magnetic Suspension Control of Rotary Impeller for Artificial Heart Pump," *Cybern. Syst.*, vol. 53, no. 1, pp. 141–167, 2022, doi: 10.1080/01969722.2021.2008686.

[36] A. J. Humaidi, H. M. Badr, and A. R. Ajil, "Design of active disturbance rejection control for single-link flexible joint robot manipulator," *22nd Int. Conf. Syst. Theory, Control Comput. ICSTCC 2018 - Proc.*, pp. 452–457, 2018, doi: 10.1109/ICSTCC.2018.8540652.

Received 14/February/2022; Accepted 30/March/2022
 ISSN (Print) 1811-9212 *ISSN (Online) 2617-3352*