Mathematical Modelling of Engineering Problems
Vol. 9, No. 2, April, 2022, pp. 379-389
Journal homepage: http://iieta.org/journals/mmep

Towards Comparison and Real Time Implementation of Path Planning Methods for 2R Planar Manipulator with Obstacles Avoidance

Mustafa Laith Muhammed[1], Amjad Jaleel Humaidi[2*], Enass Hassan Flaieh[1]

[1] Mechanical Engineering Department, University of Technology, Baghdad 10066, Iraq
[2] Control and Systems Engineering Department, University of Technology, Baghdad 10066, Iraq

Corresponding Author Email: Amjad.j.humaidi@uotechnology.edu.iq

https://doi.org/10.18280/mmep.090211

Received: 3 December 2021
Accepted: 16 March 2022

Keywords:
path planning, rational Bezier curves, NURBS, 2R planar manipulator

ABSTRACT

The main requirement of parametric path planning techniques in robot manipulators is to create continuous, smooth, and easy-to-modify path such as to move the end-effector from start point to destination point. To meet these requirements, the rational Bezier and NURBS algorithms have been proposed for path planning of 2R manipulator in environment with known and static obstacles. In this study, a comparison in terms of path length and time consumption of algorithm has been conducted to show the superior of one method to another. Based on numerical simulation, it has been shown that the rational Bezier algorithm generates shorter path and takes less time to complete the path planning task as compared to NURBS method. In addition, this study presented the design of real-time set-up based on Arduino UNO microcontroller and micro-stepping actuators. It has been shown that the experimental results could successfully verify the numerical results for both proposed path planning methods for different configuration of obstacles.

1. INTRODUCTION

Path planning is a completely geometrical task in nature, since it's defined as a geometrical path construction without regard to any specific time law [1]. Path planning is considered *offline* when it develops the plan in advance, depending on a known model of the environment (workspace), and then delivers the path to an executor (manipulator). It is defined to be *online* if the plan is gradually developed while the executor (manipulator) is working. In this situation, the planner could be sensor-based, which means it includes sensing, analysis, and execution [2].

Robot path planning is concerned with deciding how a robot will move and navigate in a workspace or environment in order to achieve its goals. Aside from avoiding obstacles, the path planning task demands the robot to compute a collision-free path between a start and a destination point. In addition, the robot must fulfill certain specifications or optimize specific performance characteristics. The amount of information available about the environment (i.e., completely known environment, partially known environment, and entirely unknown environment) influences the type of path planning. The environment is only partially known most of the time, with the robot already identifying certain locations inside the workplace prior to path planning and navigating (i.e., areas likely to pose local minimum problems). The status of an obstacle can be *static* (when its position and orientation relative to a known fixed reference system stays constant over time) or *dynamic* (when its location and orientation relative to the fixed reference system changes over time) [3].

Path planning can be either *local* or *global*. The *local* Path planning occurs, whereas the manipulator being moving and acquiring data from various sensors. In the present case, the robot possesses the capability for implementing a new course in response to the variations in environment. The *global* path planning is merely achievable when the robot's surroundings (obstacles) are static and well-recognized. In this context, the Path planning algorithm generates a complete path before the manipulator starts its movement [4, 5].

A review of the most recent studies in the field of current work has been conducted. Chen et al. [6] proposed an effective strategy for generating a Path planning with an obstacle avoidance based on Bezier Curve and Particle Swarm Optimization (PSO) for a super redundant manipulator. Tharwat et al. [7] introduced a unique Chaotic Particle Swarm Optimization (CPSO) approach for optimizing Bezier curve control points to generate optimal and smooth path between the start and destination points. Wu and Snasel [8] introduced an effectual Bezier Curve-based path planning method for a robot soccer that incorporates position modification features, path smoothing, obstacle avoidance, and path planning, in the static obstacles presence. AL-Qassar and Abdulnabi [9] presented an optimum path planning method for 5-DOF Lab-Volt 5250 robot manipulator, where the strategy was provided in the joint space utilizing the Bezier curve method with obstacles avoidance, and the PSO approach was employed for finding an optimum path. Li et al. [10] proposed a novel path planning method founded based on Fire-fly Optimization (FAO) as well as Bezier Curve that generates optimal and collision-free path. Jalel et al. [11] introduced a unique roadmap technique to generate an optimal path based on NURBS method. The efficiency of proposed technique has been proven via numerical simulation. Marthon [12] proposed a new technique for optimal path planning in complex 2D environments with static obstacles. The study utilized the graph theory to find shortest path which can avoid the obstacles. Lai et al. [13] developed a novel obstacle avoidance technique for a robot manipulator based on NURBS. Raheem and Abdulkareem [14] introduced a new approach of manipulator path planning using probabilistic roadmap as well as the artificial prospective field methods. and then used a A* method to the enhanced roadmap after that used a NURBS curve for improving the generated path. Shi et al. [15] proposed a path planning method for six-degree of freedom articulated robot arm founded upon quintic NURBS, then a genetic algorithm was adopted for optimizing the path of manipulators aims to get an optimal path length. Chen et al. [16] presented a path planning model for industrial 4-D of articulated painting manipulator based on T-Bezier curve (trigonometric Bezier curve).

Other than other path planning methods, the rational Bezier and the NURBS methods have been proposed due to their stability and ease of calculation. The presence of weight parameters in their algorithms grants them high flexibility in shaping and forming the generating path. As such, these methods have suggested for synthesizing the path of 2R manipulator in the presence of static and known obstacles.

The following points highlight the main contribution of this study:

1. Conducting a comparison study based on numerical simulation in terms of path length and time consumption of algorithm.
2. Embedded Design and implementation of the path planning algorithms for 2R manipulator in real-time environment with different obstacle configurations.
3. Experimental verification of numerical results.

2. ROBOT MANIPULATOR MODELLING

The kinematic analysis of a mechanical structure of robot manipulator is concerned with the motion depiction regarding a fixed reference. It illustrates the analytical link between the positions of joint and the position of end-effector and orientation in a Cartesian frame via neglecting the forces as well as moments that produce a structural motion [17].

The kinematic analysis is divided into (2) approaches; the forward kinematics, and the inverse kinematics. The robot manipulator's forward kinematics relates to the position and orientation computation of its end-effector frame from its joint coordinates (θ), as shown in Figure 1(a), where L1 and L2 are the link lengths of manipulator [18]:

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

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

The inverse kinematics concern with the difficulty of finding the specified angles of joints for getting a particular wanted end-effector's position and orientation. In this work, the elbow-up configuration as indicated in Figure 1(b) [19].

The simplest method to solve the inverse kinematics problem based on geometrical method is to pursue the following steps (see Figure 1):

Step 1: Determination of the length between the end-effector and manipulator base.

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

Step 2: Determination of angle between the two links of manipulator.

$$\alpha = \cos^{-1}\left(\frac{l_1^2+l_2^2-R^2}{2l_1l_2}\right) \quad (4)$$

where, α has been derived based on cosine law.

$$R^2 = l_1^2 + l_2^2 - 2l_1l_2\cos\alpha \quad (5)$$

Step 3: Obtaining the angle of elbow joint θ_2 using:

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

Using Eq. (4), one can write above equation as follows:

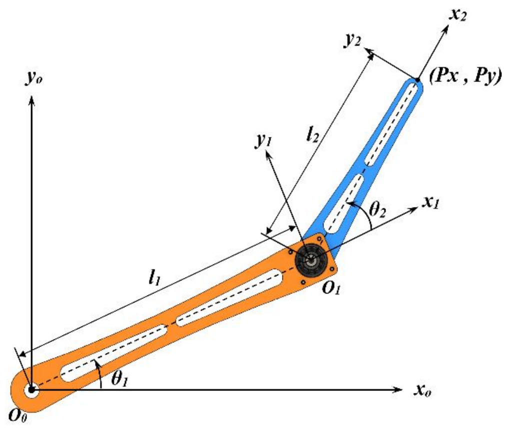
$$\theta_2 = \cos^{-1}\left(\frac{l_1^2+l_2^2-R^2}{2l_1l_2}\right) - \pi \quad (7)$$

Step 4: The joint angle θ_1 of shoulder (base) can be determined using:

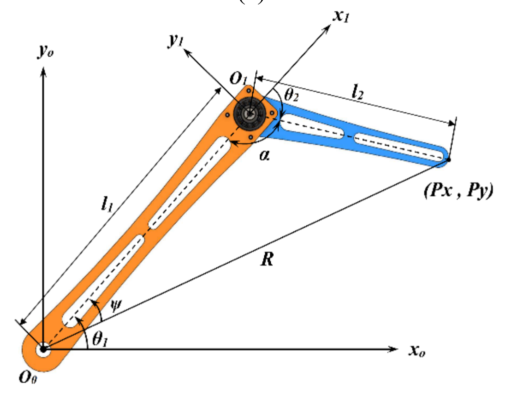
$$\psi = \tan^{-1}\left(\frac{l_2\sin\theta_2}{l_1+l_2\cos\theta_2}\right) \quad (8)$$

where, ψ is the angle between the line R and the first link.

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



(a)



(b)

Figure 1. Forward kinematics and inverse kinematics of a 2R planar manipulator

3. RATIONAL BEZIER CURVE

The Bezier curve is a parametric curve P(t). The polynomial (n) degree relies upon the control points (n+1) number used for defining the approximating curve. An nth order Bezier curve with control input (P_n) being characterized into Eq. (10) [20].

$$P(t) = \sum_{i=0}^{n} B_i^n(t) P_{i,} \qquad t \in [0,1] \tag{10}$$

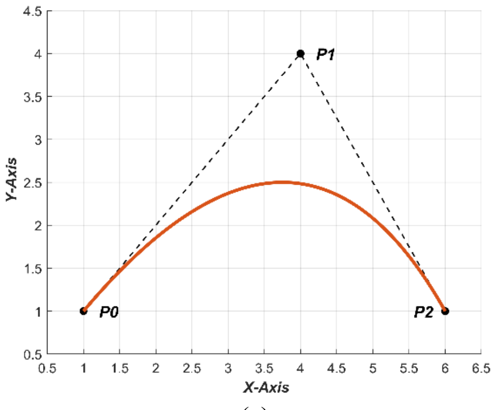
where, P_i represent the coordinates of ith control points governing the shape of the curve and $B_i^n(t)$ defines the Bernstein polynomial which given by [21]:

$$B_i^n(t) = \binom{n}{i} (i - t)^{n-i}\, t^i, \qquad i \in [0,1,\dots,n] \tag{11}$$

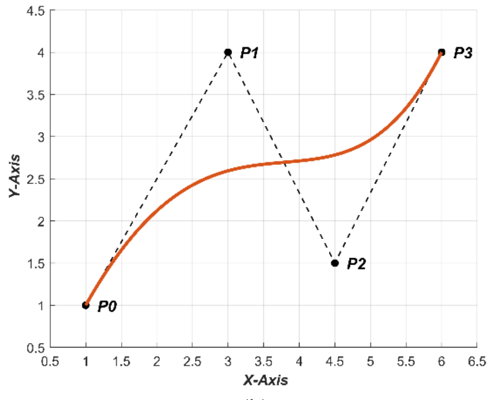
And $\binom{n}{i}$ binomial coefficient is specified via [20]:

$$\binom{n}{i} = \left(\frac{n!}{i!\,(n-i)!}\right) \tag{12}$$

The control polygon of Bezier curve is the polygon determined if the control points are being linked. The curve's geometrical features are determined from the control polygon merely. In addition to that, the curve initiates with (P0) and ends with (P_n), and the other control points just control the shape of carve and not intersect with it. Various examples of Bezier curve for several nth degree are illustrated in Figure 2 [22].



(a)



(b)

Figure 2. (a) 2nd degree. (b) 3rd degree. (c) 3rd degree. (d) 4th degree Bezier curve

The rational Bezier curve is an extension of the standard Bezier curve Eq. (10) to:

$$P(t) = \sum_{i=0}^{n} R_i^n(t) P_{i,} \qquad t \in [0,1] \tag{13}$$

where, $R_i^n(t)$ is the weight functions, which can be expressed by:

$$R_i^n(t) = \left(\frac{w_i B_i^n(t)}{\sum_{j=0}^{n} w_j B_j^n(t)}\right) \tag{14}$$

The weight functions $R_i^n(t)$ being the polynomials ratios (which is the cause for the word rational), and also, they rely upon the weights (w_i).

3.1 Weights parameter influence on Bezier curve

The weights (w_i) work as further parameters controlling the curve shape [23], and it adds more accurate control and even greater flexibility to the curve shape, where the weights varying at a particular control point are shown in Figure 3 revealing (5) curves, where the weight (w_1) rose from (0) to (5). This curve is dragged to the (P1) in a manner that the distinct points upon the curve converge at the (P1). For $w_1 = 0$, point P1 has no effect. As w_1 raises to (5), this curve gets further and further appealed to the (P1) [24].

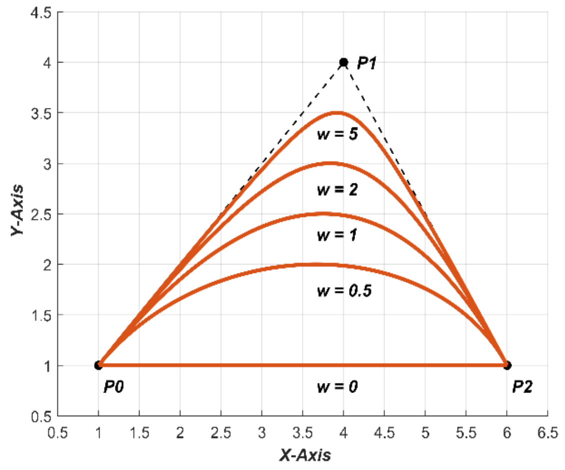


Figure 3. Weight effect on the 2nd degree Bezier curve

4. NON-UNIFORM RATIONAL B-SPLINE CURVE (NURBS)

NURBS curves are defined as a function of the control points as well as the basis functions. B-spline curve is a curve consisting of several polynomial pieces (the segments). It defined by the degree of each of its polynomial pieces (the segments), and by the number n of segments [25, 26].

The rational B-spline curves has a new set of (n + 1) parameter, named *weights* w_i, for providing more flexibility to the curve, the (non-rational) B-spline curve is [23, 27]:

$$P_{nr}(t) = \sum_{i=0}^{n} Q_i N_{ik}(t) \tag{15}$$

where, $Q_i = (x_i, y_i, w_i)$ being the 3D control points. Thus, one gets the rational B-spline $P_r(t)$ via segregating that part of $P_{nr}(t)$ that relies upon the 3rd coordinates (wi) and dividing via this part.

$$P_r(t) = \frac{\sum_{i=0}^{n} P_i\, w_i N_{ik}}{\sum_{i=0}^{n} w_i\, N_{ik}} = \sum_{i=0}^{n} P_i\, R_{ik}(t) \tag{16}$$

where, $P_i = (x_i, y_i)$ being the 2D control points, and $R_{ik}(t)$ being the fresh, rational blending functions described via

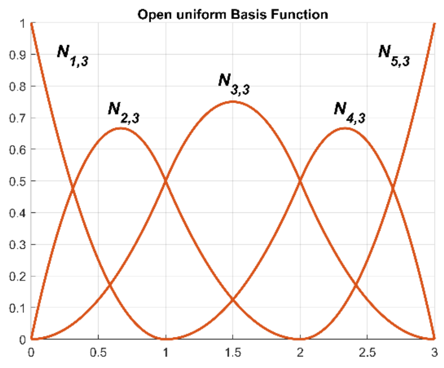
$$R_{ik}(t) = \frac{w_i N_{ik}(t)}{\sum_{i=0}^{n} w_i N_{ik}(t)} \tag{17}$$

And, non-rational basis functions $N_{ik}(t)$ calculation depends on knot vector T, it is defined recursively by:

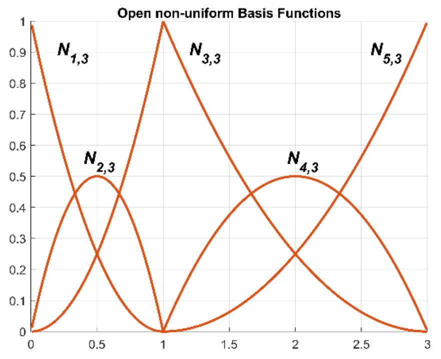
$$N_{ik}(t) = \frac{t - t_i}{t_{i+k-1} - t_i} N_{i,k-1}(t) + \frac{t_{i+k} - t}{t_{i+k} - t_{i+1}} N_{i+1,k-1}(t) \tag{18}$$

Now, for Knot Vector, it has a significant influence upon the B-spline basis functions $N_{ik}(t)$ and therefore upon the obtained B-spline curve. Basically, (2) kinds of Knot vector being utilized, Open and Periodic, in (2) kinds, Uniform and Non-uniform, as shown in Figure 4 [24].

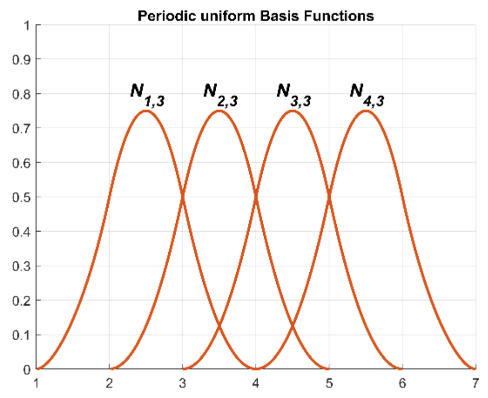
The uniform basis functions symmetry is noticed in Figures (4-a) and (4-c), and the way that the symmetry being lost in the non-uniform basis functions in Figures (4-b) and (4-d).



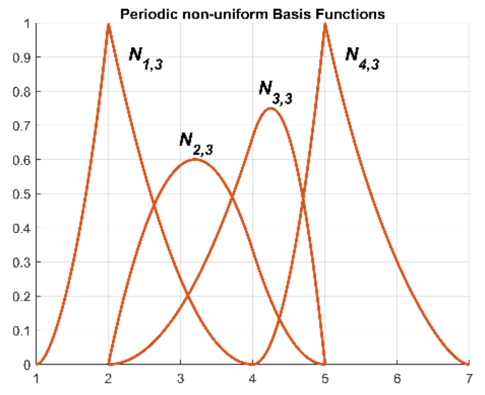
(a) T = [0 0 0 1 2 3 3 3]



(b) T = [0 0 0 1 1 3 3 3]



(c) T = [0 1 2 3 4 5 6]



(d) T = [0 1 1 2 3 4 4]

Figure 4. Uniform and Non-uniform basis functions for (n + 1 = 5) and (k = 3) with Open and periodic Knot vector

4.1 Weights parameter influence on NURBS curve

As being said earlier, the weights (w_i) that work as further parameters govern the curve shape. It is easy to see in Figure 5 the weight (w_2) effect upon the curve shape via governing the quantity of pulling that the point (P2) applies upon the curve. For weight (w_2) equals to zero, the point (P2) possesses no influence. As the weight (w_2) rises to (5), the curve gets further and further appealed to the (P2) [23].

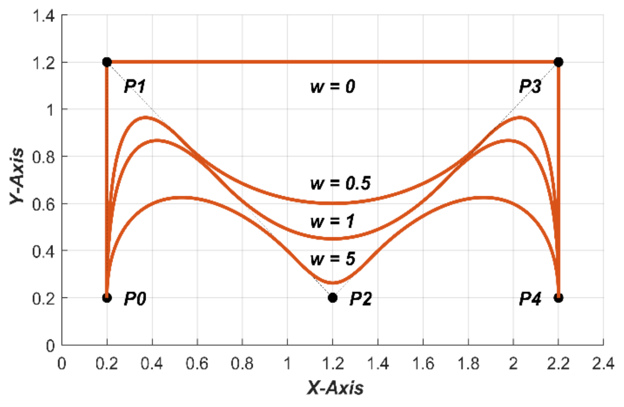


Figure 5. Weight effect on 2nd degree (n + 1 = 5 and k = 3) NURBS curve

5. IMPLEMENTATION THE PATH PLANNING OF BEZIER CURVE

To move the robot arm from a start point to a goal point in the obstacles existence, a series of joint angles alongside the path must be obtained [28].

The difficulty of obtaining a possible collision free path, from the start to the goal, can be resolved via employing Bezier method; a no. of middle points within the acceptable area of the workspace shall be determined as well as utilized for the path planning. The Bezier path planning technique is illustrated in the Pseudo Code in Figure 6.

```
Bezier_Path_Planning_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
     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:
     Read Number of Control Point;
     Read Coordinate of Control Points;
     Read Value of Control Points Weight;
     Do
     Apply $(n)^{th}$ Bernstein_polynomial;
       For i=1 to Number of Output Point, do
          Inverse_Kinematics_Elbow_Up
       End For
     While (All Output Points are Analyzed);
     Generate the Path;
End
```

Figure 6. Bezier path planning procedure

6. IMPLEMENTATION THE NURBS CURVE PATH PLANNING

NURBS were suggested to find a possible collision free path from the start to the goal, in the presence of obstacles, chosen due to their flexibility as well as geometrical properties, and joined with the truth that the algorithms of the Non-Uniform Rational B-Splines being rapid and numerically steady. The NURBS permit the of geometric forms depiction in a compact shape and they're skillful at depicting the location of a point in the workspace [11].

The NURBS path planning technique for moving the arm throughout a no. of middle points within the acceptable area of the workspace for reaching the wanted goal point being displayed in the Pseudo code in Figure 7.

```
NURBS_Path_Planning_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
     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 Workspace Parameter;
     Set Order of the Polynomial;
     Set Knot vector (T);
     Apply (n) No._Curve_Segment;
          $n = t_{n+1} - t_{k-1}$
     Read Number and Coordinate of Control Point;
     Read Value of Control Points Weight;
     Do
     Apply $N_{ik}(t)$ Cox-de Boor recursion formulas;
     Apply $P_{nr}(t)$ Position vector along the curve;
       For i=1 to Number of Output Point, do
          Inverse_Kinematics_Elbow_Up
       End For
     While (All Output Point is Analyzed);
     Generate the Path;
End
```

Figure 7. NURBS path planning procedure

7. SIMULATION RESULTS

In the present research, the suggested path planning algorithms are comprised three parts. The first one is the workspace; robot cartesian workspace is described as a space made up of points that can only be attained via a specified end-effector configuration. Inverse kinematics was used to obtain these points that are associated with joint angles. However, as illustrated in Figure 8, 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 revealed in the Figure 8 [29].

The workspace has a dimension of (50 to 50) cm in x-axis and (0 to 50) cm in 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 1:

Table 1. Obstacles coordinates for the PP algorithms

Sequence	coordinates	Case 1	Case 2
1st Obstacle	X (cm)	37	34
	Y (cm)	15	17
2nd Obstacle	X (cm)	35	18
	Y (cm)	36	31
3rd Obstacle	X (cm)	15	8
	Y (cm)	36	42
4th Obstacle	X (cm)	-11	-12
	Y (cm)	39	36

Table 2. Control points coordinates for the PP algorithms

Variables	Symbol	Coordinates		Weight
		X (cm)	Y (cm)	
1st Control Point	P0	49	2	1
2nd Control Point	P1	29	6	1.5
3rd Control Point	P2	23	23	1
4th Control Point	P3	3	31	1
5th Control Point	P4	1	45	1

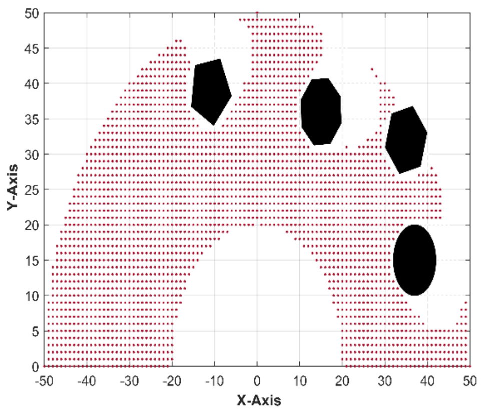


Figure 8. Work space analysis with different obstacle shape based on elbow up manipulator

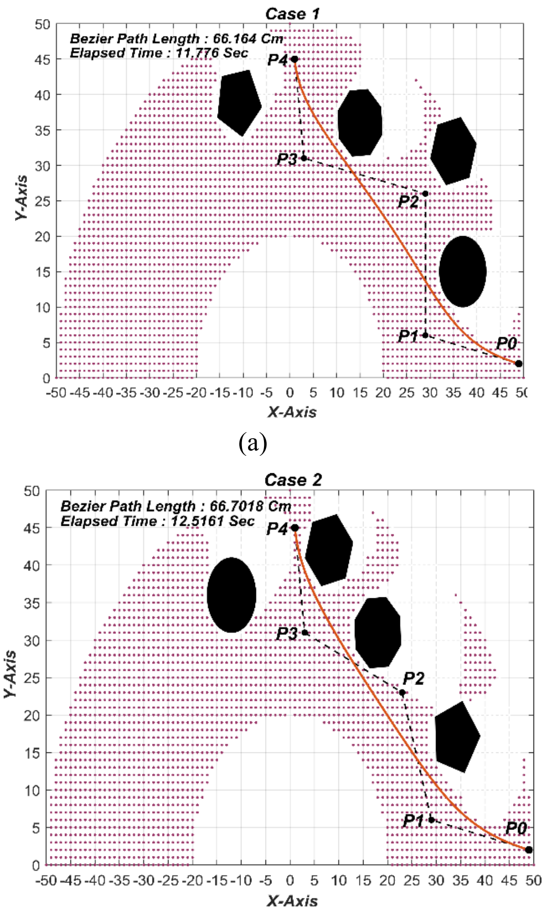


Figure 9. Bezier path planning with two obstacle configurations: (a) Case 1, (b) Case 2

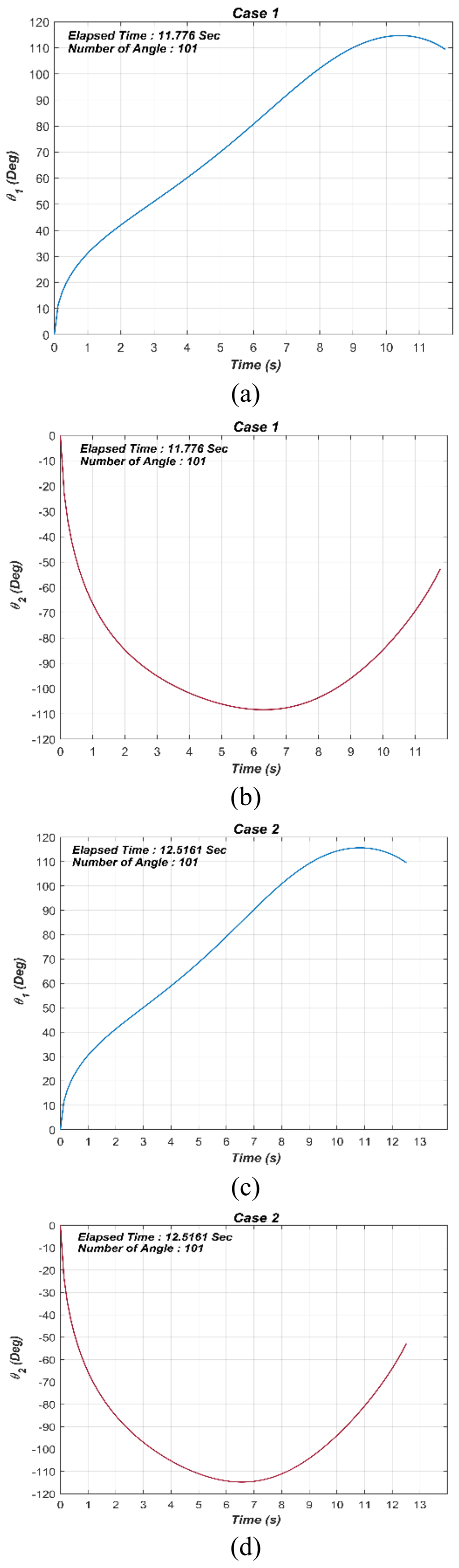
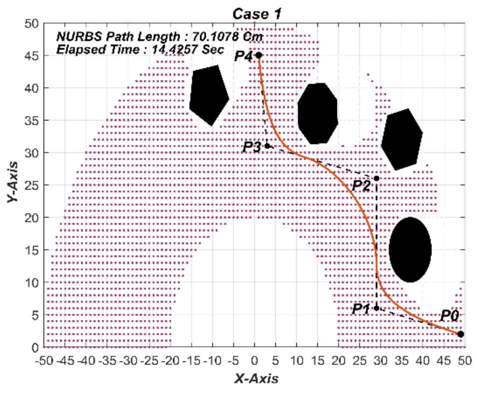
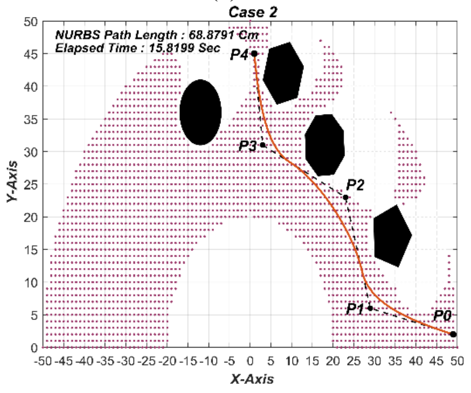


Figure 10. Bezier path planning joint variation θ_1 (a, c) and θ_2 (b, d) for both cases



(a)



(b)

Figure 11. NURBS path planning: (a) Case 1 and (b) Case 2

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 Figure 8.

The third part is the path planning process. As we mentioned in Sections 5, and 6, the path planning process for both Bezier and NURBS curves is explained in Pseudo Code in Figures 5, and 6 respectively. This process is restricted only to an acceptable area. In other words, the algorithms should find the shortest path from start to destination point within the acceptable area in shortest time as possible, otherwise the algorithm will shut down. Figure 9 shows two cases of a path planning algorithm using the Bezier curve.

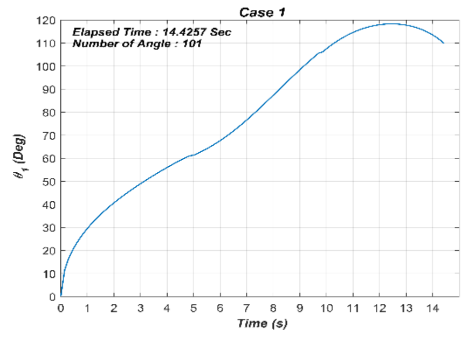
The fourth degree Bezier path planning as shown in Figure 9, is plotted due to the location of the control polygon and the value of the weight of every control point, where the control points coordinates and the weight for both cases are given in Table 2.

The generated paths in both cases are smooth, flexible, and continuous, where in case 1, the total length of the path from start to destination point is 66.164 cm, in total estimated time 11.776 sec, whereas in case 2, the path's entire length being 66.7018 cm, in total estimated time 12.5161 sec.

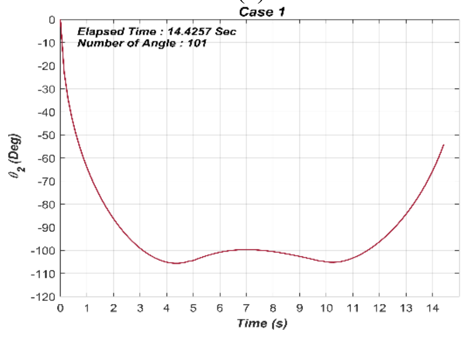
Also, Figure 10, shows the variation of the joint's angles with the estimated time for both cases.

Now the third order NURBS curve, as shown in Figure 11, is plotted based on the knot vector as explained in Section 5, where the sequence and type of knot vector are like in Figure 4(b), in addition to the control polygon location and the weight values, as manifested in Table 2.

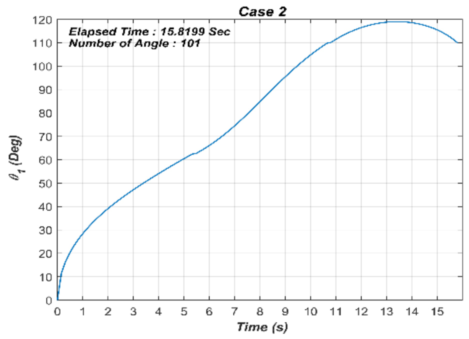
The generated paths in both cases consist of three segments, each segment controlled by three control points. In case 1, the path entire length being 70.1078 cm, in total estimated time of 14.4257 sec, whereas in case 2, the path entire length being 68.8791 cm, in total estimated time of 15.8199 sec. Figure 12, evinces the variation of the joint's angles with the estimated time for both cases.



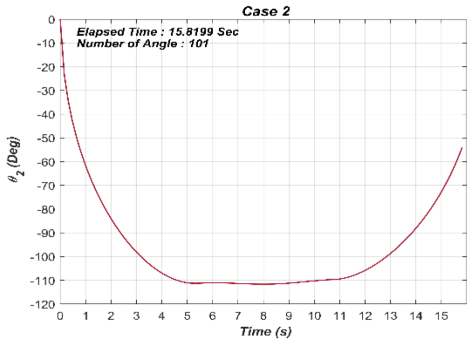
(a)



(b)



(c)

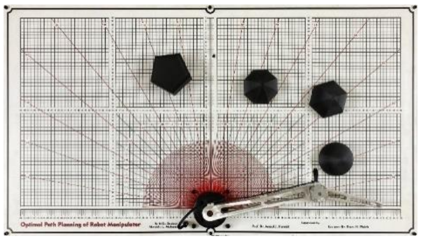


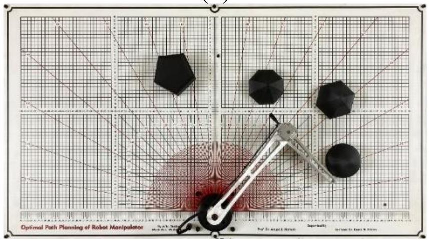
(d)

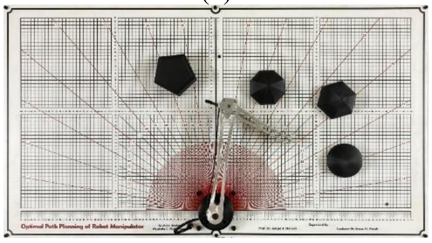
Figure 12. NURBS path planning joint variation θ_1 (a, c) and θ_2 (b, d) for both cases

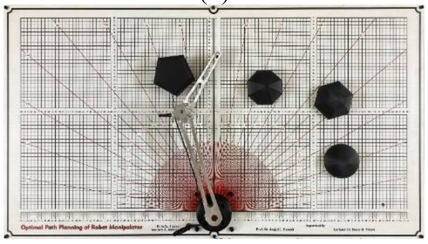
8. EXPERIMENTAL RESULTS

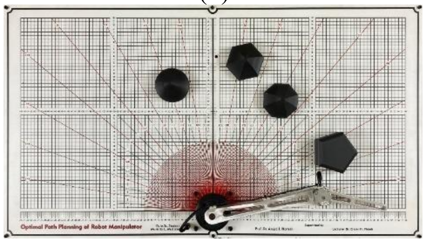
The obtained paths based on Bezier and NURBS path planning algorithms for both cases have been implemented in real time using MATLAB programming software, Arduino UNO, and two stepper motors. The first stepper motor - type Nema-17 is used to actuate the elbow joint, while the second stepper motor-type Nema-23 is utilized to drive the shoulder joint. Figure 13, shows some orientations and positions of manipulator's links during Bezier path planning. Figure 14, represents real-time implementation of NURBS path planning.

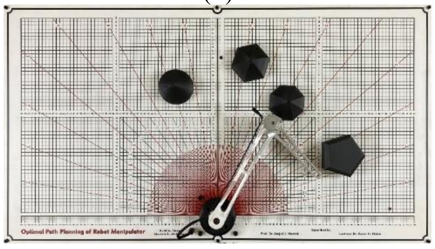

(1)

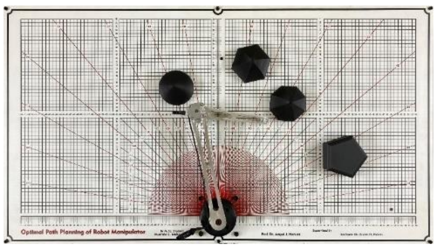

(2)


(3)


(4)


(5)


(6)


(7)

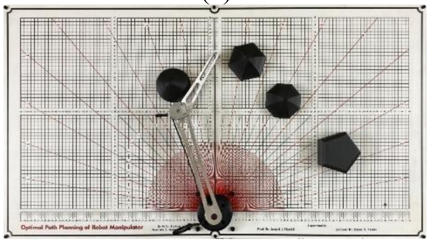
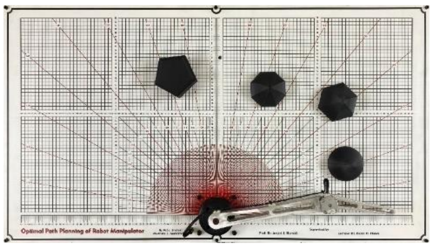
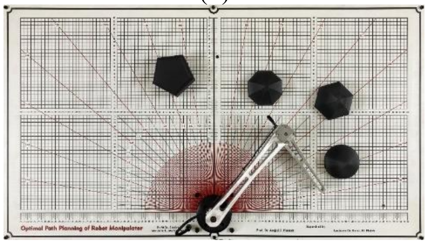
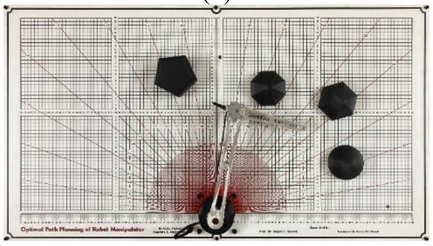
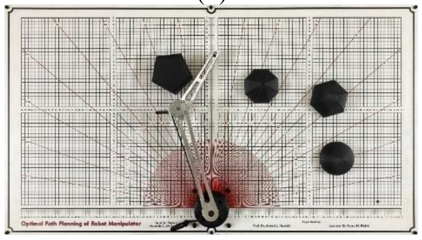

(8)

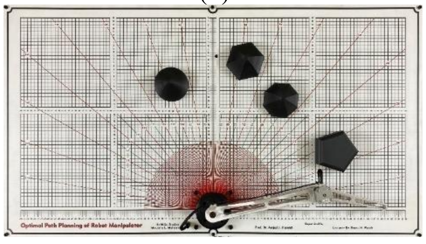
Figure 13. Real time Bezier path planning: (1, 2, 3, 4) Case 1 and (5, 6, 7, 8) Case 2

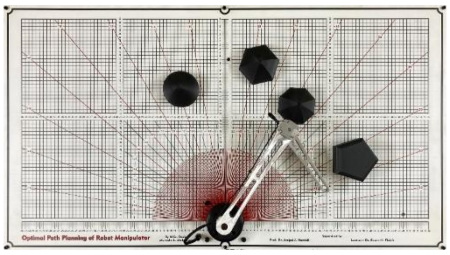

(1)


(2)

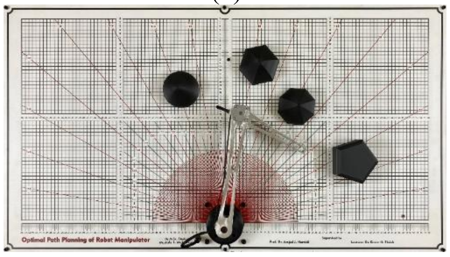

(3)


(4)

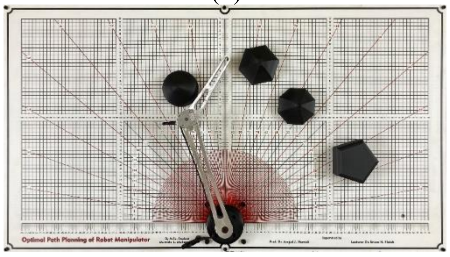

(5)



(6)



(7)



(8)

Figure 14. Real time NURBS path planning: (1, 2, 3, 4) Case 1, (5, 6, 7, 8) Case 2

Based on simulated and experimental results, the length and the time taken by proposed path planning algorithms are listed in Table 3. The table indicates that the length of path based on NURBS curve is longer than that based on Bezier curve. Also, the NURBS algorithm takes longer time than the Bezier algorithm. According to this, Bezier path planning is significantly faster and more efficient than NURBS path planning in generating the shortest path from start to destination point in the absence of static obstacles at various locations, as shown in Table 3.

Table 3. Comparison of the proposed path planning algorithms

Algorithm	**Case 1**		**Case 2**	
	Length (cm)	Time (sec)	Length (cm)	Time (sec)
Bezier Path Planning	66.164	11.776	66.7018	12.5161
NURBS Path Planning	70.1078	14.4257	68.8791	15.8199

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 [30-36]. 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) [37-43].

9. CONCLUSIONS

This study has addressed the analysis and implementation of path planning algorithm for planar and two-arm manipulator in the presence of obstacles. Two algorithms have been presented for executing path planning: Bezier and NURBS curves. Simulation based on MATLAB has been performed for both algorithms. Also, experimental and real-time scenarios have been conducted to show the validity of proposed algorithms. The results showed that the path obtained by Bezier curve is shorter than that based on NURBS curve. Moreover, the Bezier algorithm could take the robot gripper form start to end in less time than its counterpart. This work can be extended to consider other techniques of path planning such as A*, D*, and RRT.

REFERENCES

[1] Carbone, G., Gomez-Bravo, F. (2015). Motion and operation planning of robotic systems. Springer International Publishing. Switzerland. https://doi.org/10.1007/978-3-319-14705-5

[2] Choset, H., Lynch, K.M., Hutchinson, S., Kantor, G.A., Burgard, W. (2005). Principles of Robot Motion: Theory, Algorithms, and Implementations. MIT Press.

[3] Tzafestas, S. G. (2013). Introduction to mobile robot control. Elsevier. https://doi.org/10.1016/C2013-0-01365-5

[4] Elbanhawi, M., Simic, M. (2014). Sampling-based robot motion planning: A review. IEEE Access, 2: 56-77. https://doi.org/10.1109/ACCESS.2014.2302442

[5] Kunchev, V., Jain, L., Ivancevic, V., Finn, A. (2006). Path planning and obstacle avoidance for autonomous mobile robots: A review. In International Conference on Knowledge-Based and Intelligent Information and Engineering Systems, pp. 537-544. https://doi.org/10.1007/11893004_70

[6] Chen, L., Ma, Y., Zhang, Y., Liu, J. (2020). Obstacle avoidance and multitarget tracking of a super redundant modular manipulator based on Bezier curve and particle swarm optimization. Chinese Journal of Mechanical Engineering, 33(1): 1-19. https://doi.org/10.21203/rs.3.rs-19143/v2

[7] Tharwat, A., Elhoseny, M., Hassanien, A.E., Gabel, T., Kumar, A. (2019). Intelligent Bézier curve-based path planning model using Chaotic Particle Swarm Optimization algorithm. Cluster Computing, 22(2): 4745-4766. https://doi.org/10.1007/s10586-018-2360-3

[8] Wu, J., Snášel, V. (2014). A Bezier curve-based approach for path planning in robot soccer. In Innovations in Bio-inspired Computing and Applications, pp. 105-113. https://doi.org/10.1007/978-3-319-01781-5_10

[9] AL-Qassar, A.A., Abdulnabi, A.N. (2018). Optimal Path Planning Obstacle Avoidance of Robot Manipulator System using Bézier Curve. American Academic Scientific Research Journal for Engineering, Technology, and Sciences, 40(1): 6-17.

[10] Li, B., Liu, L., Zhang, Q., Lv, D., Zhang, Y., Zhang, J., Shi, X. (2014). Path planning based on firefly algorithm and Bezier curve. In 2014 IEEE International Conference on Information and Automation (ICIA), pp. 630-633. https://doi.org/10.1109/ICInfA.2014.6932730

[11] Jalel, S., Marthon, P., Hamouda, A. (2016). A new path generation algorithm based on accurate NURBS curves. International Journal of Advanced Robotic Systems, 13(2): 75. https://doi.org/10.5772/63072

[12] Marthon, P. (2013). Optimum path planning for mobile robots in static environments using graph modelling and NURBS curves. International Journal of Advanced Robotic Systems, 216-221.

[13] Lai, T.C., Xiao, S.R., Aoyama, H., Wong, C.C. (2017). Path planning and obstacle avoidance approaches for robot arm. In 2017 56th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE), pp. 334-337. https://doi.org/10.23919/SICE.2017.8105619

[14] Raheem, F.A., Abdulkareem, M.I. (2020). Development of A* algorithm for robot path planning based on modified probabilistic roadmap and artificial potential field. Journal of Engineering Science and Technology, 15(5): 3034-3054.

[15] Shi, X., Fang, H., Guo, L. (2016). Multi-objective optimal trajectory planning of manipulators based on quintic NURBS. In 2016 IEEE International Conference on Mechatronics and Automation, pp. 759-765. https://doi.org/10.1109/ICMA.2016.7558658

[16] Chen, W., Sun, C., Liu, H., Liu, J., Tang, Y. (2017). Path planning scheme for spray painting robot with Bézier curves on complex curved surfaces. In 2017 32nd Youth Academic Annual Conference of Chinese Association of Automation (YAC), pp. 698-703. https://doi.org/10.1109/YAC.2017.7967499

[17] Siciliano, B., Sciavicco, L., Villani, L., Oriolo, G. (2009). Robotics. Advanced Textbooks in Control and Signal Processing. https://doi.org/10.1007/978-1-84628-642-1

[18] Lynch, K.M., Park, F.C. (2017). Modern Robotics Mechanics, Planning, and Control. Cambridge University Press.

[19] Rehiara, A.B. (2011). Kinematics of adept three robot arm. Robot Arms, 2: 21-38. https://doi.org/10.5772/17732

[20] Kawabata, K., Ma, L., Xue, J., Zhu, C., Zheng, N. (2015). A path generation for automated vehicle based on Bezier curve and via-points. Robotics and Autonomous Systems, 74: 243-252. https://doi.org/10.1016/j.robot.2015.08.001

[21] Choi, J.W., Curry, R.E., Elkaim, G.H. (2010). Continuous Curvature Path Generation Based on Bézier Curves for Autonomous Vehicles. IAENG International Journal of Applied Mathematics, 40(2).

[22] Hosaka, M. (2012). Modeling of Curves and Surfaces in CAD/CAM. Springer Science & Business Media. https://doi.org/10.1007/978-3-642-76598-8

[23] Salomon, D. (2007). Curves and surfaces for computer graphics. Springer Science & Business Media. https://doi.org/10.1007/0-387-28452-4

[24] Rogers, D.F. (2001). An introduction to NURBS: With historical perspective. Morgan Kaufmann. https://doi.org/10.1016/B978-1-55860-669-2.X5000-3

[25] Piegl, L.A., Tiller, W. (1998). Computing the derivative of NURBS with respect to a knot. Computer Aided Geometric Design, 15(9): 925-934. https://doi.org/10.1016/S0167-8396(98)00028-4

[26] Farin, G.E. (2018). NURBS from Projective Geometry to Practical Use. Second edition. CRC Press Taylor & Francis Group.

[27] Alwan, A.K., Hamdan, W. (2020). Geometric modeling of compound NURBS surfaces. Engineering and Technology Journal, 38(2): 277-287. https://doi.org/10.30684/etj.v38i2A.300

[28] Hlaváč, V. (2008). Manipulator trajectory planning. Czech Technical University in Prague, 39.

[29] Raheem, F.A., Sadiq, A.T., Abbas, N.A.F. (2019). Robot arm free Cartesian space analysis for heuristic path planning enhancement. International Journal of Mechanical Engineering, 19: 29-42.

[30] Humaidi, A.J., Badr, H.M., Ajil, A.R. (2018). Design of active disturbance rejection control for single-link flexible joint robot manipulator. In 2018 22nd International Conference on System Theory, Control and Computing (ICSTCC), pp. 452-457. https://doi.org/10.1109/ICSTCC.2018.8540652

[31] Humaidi, A.J., Hameed, M., Hameed, A.H. (2018). Design of block-backstepping controller to ball and arc system based on zero dynamic theory. Journal of Engineering Science and Technology, 13(7): 2084-2105.

[32] Humaidi, A.J., Hameed, A.H. (2019). Design and comparative study of advanced adaptive control schemes for position control of electronic throttle valve. Information, 10(2): 65. https://doi.org/10.3390/info10020065

[33] Humaidi, A.J., Hameed, A.H., Hameed, M.R. (2017). Robust adaptive speed control for DC motor using novel weighted E-modified MRAC. In 2017 IEEE International Conference on Power, Control, Signals and Instrumentation Engineering (ICPCSI), pp. 313-319. https://doi.org/10.1109/ICPCSI.2017.8392302

[34] Humaidi, A.J., Badr, H.M. (2018). Linear and nonlinear active disturbance rejection controllers for single-link flexible joint robot manipulator based on PSO tuner. Journal of Engineering Science & Technology Review, 11(3). https://doi.org/10.25103/jestr.113.18

[35] Humaidi, A.J., Tala'at, E.N., Hameed, M.R., Hameed, A.H. (2019). Design of adaptive observer-based backstepping control of cart-pole pendulum system. In 2019 IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT), pp. 1-5. https://doi.org/10.1109/ICECCT.2019.8869179

[36] Hameed, A.H., Al-Dujaili, A.Q., Humaidi, A.J., Hussein, H.A. (2019). Design of terminal sliding position control for electronic throttle valve system: A performance comparative study. International Review of Automatic Control, 12(5): 251-260. https://doi.org/10.15866/ireaco.v12i5.16556

[37] Humaidi, A.J., Abdulkareem, A.I. (2019). Design of augmented nonlinear PD controller of Delta/Par4-like robot. Journal of Control Science and Engineering, 2019: 7689673. https://doi.org/10.1155/2019/7689673

[38] Humaidi, A.J., Kadhim, S.K., Gataa, A.S. (2022). Optimal adaptive magnetic suspension control of rotary impeller for artificial heart pump. Cybernetics and Systems, 53(1): 141-167. https://doi.org/10.1080/01969722.2021.2008686

[39] Humaidi, A.J., Kadhim, S.K., Gataa, A.S. (2020). Development of a novel optimal backstepping control algorithm of magnetic impeller-bearing system for artificial heart ventricle pump. Cybernetics and Systems, 51(4): 521-541. https://doi.org/10.1080/01969722.2020.1758467

[40] Ghanim, T., Ajel, A.R. (2020). Optimal fuzzy logic control for temperature control based on social spider

optimization. In IOP Conference Series: Materials Science and Engineering, 745(1): 012099. https://doi.org/10.1088/1757-899X/745/1/012099

[41] Al-Qassar, A.A., Al-Obaidi, A.S., Hasan, A.F., Humaidi, A.J., Nasser, A.R., Alkhayyat, A., Ibraheem K.I. (2021). Finite-time control of wing-rock motion for delta wing aircraft based on whale-optimization algorithm. Indonesian Journal of Science & Technology, 6(3): 441-456. https://doi.org/10.17509/ijost.v6i3.37922

[42] Humaidi, A.J., Oglah, A.A., Abbas, S.J., Ibraheem, I.K. (2019). Optimal Augmented linear and nonlinear PD control design for parallel robot based on PSO Tuner. International Review on Modelling and Simulations, 12(5). https://doi.org/10.15866/iremos.v12i5.16298

[43] Al-Qassar, A.A., Abdulkareem, A.I., Hasan, A.F., Humaidi, A.J., Kasim, I., Ibraheem, A.T.A., Hameed, A.H. (2021). Grey-wolf optimization better enhances the dynamic performance of roll motion for tail-sitter VTOL aircraft guided and controlled by STSMC. Journal of Engineering Science and Technology, 16(3): 1932-1950.

NOMENCLATURE

k	Order of polynomial
l_1	First link length, *cm*
l_2	Second link length, *cm*
n	Degree of polynomial
$P(t)$	Parametric curve
P_i	2D-Control point Coordinate, *cm*
P_n	Control Point, *cm*
$P_{nr}(t)$	Non-rational B-spline curve
$P_r(t)$	Rational B-spline curve
P_x	*End effector position in x-axis, cm*
P_y	*End effector position in y-axis,* cm
Q_i	3D-Control point Coordinate, *cm*
R	Distance from end effector to base of arm, *cm*
T	Knot vector
w_i	The weight

Greek symbols

θ_i	Joint angle, *rad*
α	Angle between the two links, *rad*
Ψ	Angle between first link and R, *rad*

Subscripts

$B_n^i(t)$	The Bernstein polynomial
$N_{ik}(t)$	Rational blending functions
$R_{ik}(t)$	B-spline basis functions
$R_n^i(t)$	The weight functions