الجمهورية الجزائرية الديمقراطية الشعبية

République Algérienne Démocratique et Populaire

وزارة التعليم العالي والبحث العلمي

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Département : Génie électrique Et Informatique Industrielle

Mémoire de projet de fin d'études

Pour l'obtention du diplôme du master en Automatique

Spécialité
Automatique Et Informatique Industrielle

Trajectory planning and simulation of the industrial robot ABB-IRB 6700 based on MATLAB and RobotStudio

Réalisé par

Aboubakeur BENBOUZID
Mohamed Amine RAHMANI

Présent(e) et soutenue publiquement le (06/07/2022)

Composition du Jury :

Président	M. Amar REZOUG	M.C.A	ENST
Promoteur	M .Samir KHELOUAT	M.C.B	ENST
Promoteur	M .Mohamed Amine MAMOU	M.C.B	USTHB
Examinateur	Mme. Zahira OUSAADI	M.A.A	ENST
Examinateur	M .Nait Slimani BOUKHALFA	M.A.A	ENST

Trajectory planning and simulation of the industrial robot ABB-IRB 6700 based on MATLAB and RobotStudio

RAHMANI Mohamed Amine
Génie Electrique et Informatique Industrielle
Ecole Nationale Supérieure de Technologie
Algiers, Alegria
m_rahmani@enst.dz

M .Samir KHELOUAT
Ecole Nationale Supérieure de Technologie
Algiers, Alegria

BENBOUZID Aboubakeur
Génie Electrique et Informatique Industrielle
Ecole Nationale Supérieure de Technologie
Algiers, Alegria
a_benbouzid@enst.dz

M .Mohamed Amine MAMOU
Ecole Nationale Supérieure de Technologie
Algiers, Alegria

Abstract- This study discusses the issue of finding forward kinematics and inverse kinematics for industrial robots and formulating them in MATLAB and using simulation to ensure the reliability of the model and its ability to planning and tracking trajectories. To prove this, we formulate forward kinematics and inverse kinematics model for the ABB-IRB 6700 robot. We use interpolation method for generating trajectory in joint space and in Cartesian space of the robot and evaluating its ability in tracking it, to ensure the correctness of the proposed models, we formulate a communication between Matlab and Robotstudio, and show that the end effector of the IRB 6700 robot, available on Robotstudio, follows the same trajectory as in Matlab.

Keywords*: industrial robot, MATLAB, RobotStudio, trajectory planning, simulation.*

I. PREFACE

In robotic, the correctness of the forward and inverse kinematic is of great utilities to perform control law. By definition, the forward kinematic allows to determine the Cartesian position of the end effector knowing the joint angles of the robot. The cartesian displacement, anticipated angular displacement, angular velocity, and angular acceleration of each joint in a robot is known as joint space trajectory planning.

Determine the inverse kinematic model using the final position result Robot's Cartesian space, demonstrates that the trajectories of the robot moves in a smooth and consistent manner. In order to double-check the information through the simulation, the practicality of the trajectory planning method of similar robot model is constructed in MATLAB environment with the help of RobotStudio.

When working with robots, modeling and simulation enable you to address algorithms quickly and test different cenarios by imitating the behavior of real-world systems. These functions provide kinematic models for manipulators robots to model their motion and that can be done with MATLAB.

The choice of a particular trajectory has direct and relevant implications on several aspects of the design and use of an automatic machine, like the dimensioning of the actuators and of the reduction gears, the vibrations and efforts generated on the machine and on the load, the tracking errors during the motion execution [1]. For these reasons, in order to understand and

appreciate the particularities of different techniques available for trajectory planning.

It is not enough to rely on MATLAB simulation to establish that the above-mentioned trajectory planning is in line with the actual work scenario. It's also important to analyze as closely as possible to the real robot trajectory [2].

This study takes the IRB-6700 robot as the research object, establishes the robot model on the MATLAB platform, performs the analysis of forward and inverse kinematics, and analyzes and simulates the joint space and Cartesian space trajectory planning of the robot. MATLAB calculates the positive kinematics solution through a certain algorithm to obtain the trajectory of the robot movement, which is used for research and design of industrial robots. RobotStudio simulates the actual robot motion trajectory, used to check whether the motion trajectory meets the process requirements. The purpose of this article is to use RobotStudio simulation software to establish a robot model consistent with the model built by MATLAB, simulate the actual robot motion trajectory with RobotStudio, and compare it with the motion trajectory calculated by MATLAB to prove the correctness of the motion trajectory calculated by MATLAB.

II. MODELING OF FORWARD KINEMATICS

Denavit-Hartenberg (DH) matrix is a matrix describing the translation and rotation between two dependent frames. It is a comprehensive approach to model the kinematics in the manipulators, where each axis movement affects the position of all descendant axes.

The DH matrix is composed by four matrices that describes the relationship between frame i and frame i -1. These matrices are, a forward rotation matrix Rx, with a rotation αi-1 around the x-axis of the previous frame i, a forward translation matrix D_x with the translation a_{i-1} along the previous frame i-1's x-axis, a joint rotation matrix Rz with a rotation θ_i around the frame i's z-axis and a joint translation matrix D_z. Each of these values are defined by the manipulator design, hence are static, except for the joint rotation angle θ_i which represent the joint variables. The DH coordinate system is depicted by figure 1.

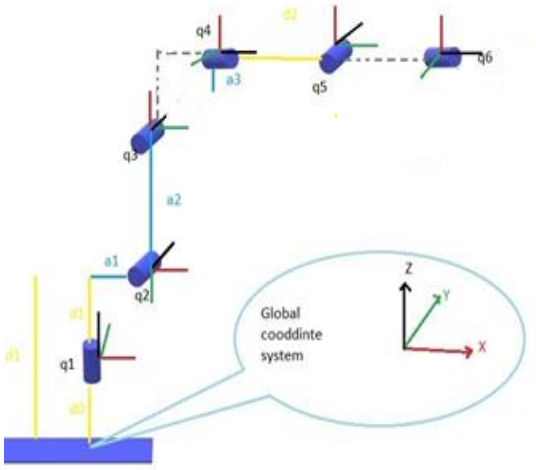


Figure 1: IRB 6700 D-H coordinates system

The formulation of the forward kinematic model is given by the following equation:

$$^{i-1}_{i}T =$$

$$Rot(x,\alpha_{i-1})Trans(x,a_{i-1})rot(z,\theta_i)Trans(z,d_i),$$

Where

$$Rot(x,\alpha_{i-1}) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & C\alpha_{i-1} & -S\alpha_{i-1} & 0 \\ 0 & S\alpha_{i-1} & C\,\alpha_{i-1} & d_i \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$Trans(x,a_{i-1}) = \begin{pmatrix} 1 & 0 & 0 & a_{i-1} \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$rot(z,\theta_i) = \begin{pmatrix} C\theta_{i-1} & -S\theta_{i-1} & 0 & 0 \\ S\theta_{i-1} & C\theta_{i-1} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$Trans(z,d_i) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Consequently

$$
{}^{i-1}_{i}T = \begin{pmatrix} C\theta_i & -S\theta_i & 0 & a_{i-1} \\ S\theta_i . C\alpha_{i-1} & C\theta_i . C\alpha_{i-1} & -S\alpha_{i-1} & -S\alpha_{i-1} . d_i \\ S\theta_i . S\alpha_{i-1} & C\theta_i . S\alpha_{i-1} & C\alpha_{i-1} & C\alpha_{i-1} . d_i \\ 0 & 0 & 0 & 1 \end{pmatrix}
$$

Note that C_i = cos (θ_i), S_i = sin (θ_i), $C_{i,j} = \cos(\theta_i + \theta_j)$ and $S_{i,j} = \sin(\theta_i + \theta_j)$

The D-H parameter table of the IRB-6700 robot was obtained in table 1.

Joint	Joint parameters			
	a_i	α_i	d_i	θ_i
B	0	0	d_b	0
1	$a_1(320)$	0	$d_1(780)$	$\theta'_1 = \theta_1$
2	$a_2(1135)$	$-\pi/2$	0	$\theta'_2 - \frac{\pi}{2} = \theta_2$
3	$a_3(200)$	0	0	$\theta'_3 = \theta_3$
4	0	$-\pi/2$	$d_4(1182$	$\theta'_4 = \theta_4$
5	0	$\pi/2$	0	$\theta'_5 = \theta_5$
6	$a_6(0)$	$-\pi/2$	0	$\theta'_6 + \pi = \theta_6$

Table 1: D-H parameters of IRB 6700

Finding the matrix ${}^{i-1}_{i}T$ for each row of according to Table 1, the different transformation matrices can be found as below.

$$
{}^{0}_{1}T = \begin{pmatrix} cos\theta_1 & -sin\theta_1 & 0 & 0 \\ sin\theta_1 & cos\theta_1 & 0 & 0 \\ 0 & 0 & 1 & d_1 \\ 0 & 0 & 0 & 1 \end{pmatrix}
$$

$$
{}^{1}_{2}T = \begin{pmatrix} cos\theta_2 & -sin\theta_2 & 0 & a_1 \\ 0 & 0 & 1 & 0 \\ -sin\theta_2 & -cos\theta_2 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}
$$

$$
{}^{2}_{3}T = \begin{pmatrix} cos\theta_3 & -sin\theta_3 & 0 & a_2 \\ sin\theta_3 & cos\theta_3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}
$$

$$
{}^{3}_{4}T = \begin{pmatrix} cos\theta_4 & -sin\theta_4 & 0 & a_3 \\ 0 & 0 & 1 & d_4 \\ -sin\theta_4 & -cos\theta_4 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}
$$

$$
{}^{4}_{5}T = \begin{pmatrix} cos\theta_5 & -sin\theta_5 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ sin\theta_5 & cos\theta_5 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}
$$

$$
{}^{5}_{6}T = \begin{pmatrix} cos\theta_6 & -sin\theta_6 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -sin\theta_6 & -cos\theta_6 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}
$$

These individual matrices are combined to determine the overall transformation matrix which describes the forward kinematics. Hence, the end effector frame is described with respect to the base frame. By multiplying the matrices ${}^{0}_{1}T$ through ${}^{5}_{6}T$ yield ${}^{0}_{6}T$, where

$$
{}^{0}_{6}T = \begin{pmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{pmatrix}
$$

Containing the following equations which define the position of the end effector:

$$
\begin{aligned} p_x = a_1 cos(\theta_1) + a_3 cos(\theta_2 + \theta_3) cos(\theta_1) \\ - d_4 sin(\theta_2 + \theta_3) cos(\theta_1) \\ + a_2 cos(\theta_1) cos(\theta_2) \end{aligned}
$$

$$
\begin{aligned} p_y = a_1 sin(\theta_1) + a_3 cos(\theta_2 + \theta_3) sin(\theta_1) \\ - d_4 sin(\theta_2 + \theta_3) sin(\theta_1) \\ + a_2 cos(\theta_1) sin(\theta_1) \end{aligned}
$$

$$
\begin{aligned} p_z = d_1 - a_3 sin(\theta_2 + \theta_3) - d_4 cos(\theta_2 + \theta_3) \\ - a_2 sin(\theta_1) \end{aligned}
$$

The IRB-6700 robot model output by MATLAB with $\theta = [0, -\pi/2, 0, 0, 0, 0]$ is depicted by figure 1.

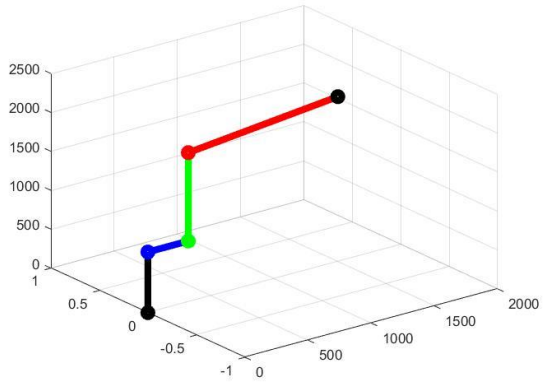


Figure 2: IRB -6700 robot model output by Matlab

III. MODELING THE INVERSE KINEMATICS OF THE ROBOT.

Unlike linear equations, there are no general algorithms that may be employed to solve a set of nonlinear equations. In considering methods of solution, it will be wise to define what constitutes the "solution" of a given manipulator. A manipulator will be considered solvable if the joint variables can be determined by an algorithm that allows one to determine all the sets of joint variables associated with a given position and orientation [3].

In this section the manipulator solution is done purely algebraically by using Paul method:

$$\theta_1 = atan2(y, x)$$

$$\theta_2 = atan2(-ce_1 + ae_2\ , de_1 - be_2)$$

Where

$$a = -a_1 + ysin(\theta_1) + xcos(\theta_1)$$

$$b = d_1 - z$$

$$d = a_1 - ysin(\theta_1) - xcos(\theta_1)$$

$$c = d_1 - z$$

$$e_1 = a_2 + a_3cos(\theta_3) - d_4\ sin(\theta_3)$$

$$e_2\ = d_4cos(\theta_3) + a_3sin(\theta_3)$$

And

$$p_x = xp_y = yp_z = z$$

$$\theta_3 = \text{atan2}(\mu_1, \mu_2)$$

With

$$\mu_1 = \frac{(a_3\cos(\theta_3) - K)}{d_4}$$

$$\mu_2 = \frac{a_3K \pm \sqrt{(a_3K)^2 - (a_3^2 + d_4^2)(K^2 - d_4^2)}}{{a_3}^2 + {d_4}^2}$$

$$K = \frac{1}{2a_2}\big((-a_1 + xcos(\theta_1) + ysin(\theta_1))^2 - {a_2}^2 - {a_3}^2 - {d_4}^2 + (z - d_1)^2\big)$$

And

$$\theta_4 = atan2(q1, q)$$

$$\theta_5 = atan2(\pm\sqrt{q1^2 + q^2}, C)$$

$$\theta_6 = atan2(a, b)$$

Where:

$$q = r_{23}cos(\theta_1) - r_{13}sin(\theta_1)$$

$$q1 = r_{33}sin(\theta_2 + \theta_3) - r_{13}cos(\theta_2 + \theta_3)cos(\theta_1) - r_{23}cos(\theta_2 + \theta_3)sin(\theta_1)$$

$$C = r_{33}cos(\theta_2 + \theta_3) - r_{13}sin(\theta_2 + \theta_3)cos(\theta_1) - r_{23}sin(\theta_2 + \theta_3)sin(\theta_1)$$

$$a = -r_{12}\big(c_5s_1s_4 - s_{2,3}c_1s_5 + c_{2,3}c_5c_1c_4\big) + r_{22}\big(s_{2,3}s_1s_5 + c_5c_1c_4 - c_{2,3}c_5s_1c_4\big) + r_{32}(c_{2,3}s_5 + s_{2,3}c_4c_5)$$

$$b = r_{12}\big(c_4s_1 - c_{2,3}c_1s_4\big) - r_{22}\big(c_1c_4 - c_{2,3}s_1s_4\big) + r_{32}s_{2,3}s_4$$

To perform the validation of forward and inverse kinematics models computed previously, we introduce a given joints $\theta = [0, -\pi/2, 0, 0, 0, 0]$ and compute the end effector position using the forward kinematics. The result is represented by the following matrix:

```
1.0e+03 *

    0.0000         0    0.0010    1.5025
         0   -0.0010         0         0
    0.0010         0   -0.0000    2.1150
         0         0         0    0.0010
```

And we get with the inverse kinematic model using the previous obtained matrix; we get the same joint variables.

$$\theta = [0, -\pi/2, 0, \pi/2, 0, \pi/2]$$

This proves the correctness of the obtained forward and inverse kinematics models.

VI. TRAJECTORY PLANNING ANALYSIS AND SIMULATION

A. Trajectory Planning of Joint Space

Usually, it is desirable for the motion of the manipulator to be smooth. For our purposes, we will define a smooth function as a function that is continuous and has a continuous first derivative. Sometimes a continuous second derivative is also desirable. Rough, jerky motions tend to cause increased wear on the mechanism and cause vibrations by exciting resonances in the manipulator. In order to guarantee smooth paths, we must put some sort of constraints on the spatial and temporal qualities of the path between and his points. At this point, there are many choices that may be made and, consequently, a great variety in the ways that paths might be specified and planned. Any smooth functions of time that pass through the points could be used to specify the exact path shape.

Therefore, the third-order or fifth-order polynomial interpolation method is used more frequently [4]. For that, the quintic polynomial interpolation method is used to generate a trajectory:

$$q(t) = \mathrm{a0} + \mathrm{a1t} + \mathrm{a2}t^2 + \mathrm{a3}t^3 + \mathrm{a4}t^4 + \mathrm{a5}t^5$$

There are a total of 6 unknown coefficients a0~a5 in the quintic polynomial function. By deriving its first and second order, the angular velocity and angular acceleration can be obtained as a function of time t. At $t_0 = 0$, the starting joint angle, angular velocity, and angular acceleration are respectively:

$$q(0) = qi\,, \qquad q'(0) = 0, \qquad q''(0) = 0$$

And at the end time t_f, the ending joint angle, angular velocity, and angular acceleration arerespectively:

$$q(tf) = qf, \qquad q'(tf) = 0\,, \qquad q''(tf) = 0$$

Hence, there are 6 equations, and 6 unknown coefficients in the quintic polynomial function can be calculated and found as follow:

$a0 = qi,\ a1 = 0\ \ a20$, a3=10D/tf ^{3}, a_4=15D/ tf^4, a_5=6D/ tf^5

where

$$D = qf - qi$$

For simulation, we take $qi = [0, \frac{-\pi}{2}, 0,0,0,0]$ as the starting joint angles, and the ending is $qf = [\frac{\pi}{2}, \frac{\pi}{2}, \frac{\pi}{4}, \frac{\pi}{6}, \frac{\pi}{3}, \frac{-\pi}{6}]$. The cycle time is TF=10s. The results are depicted by Figures 2 to 7.

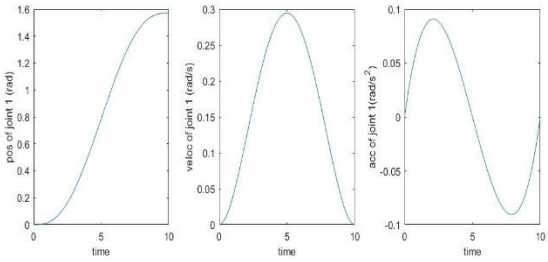


Figure 3: Angular position, velocity, and acceleration of joint 1.

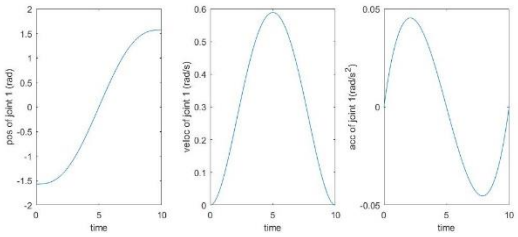


Figure 4: Angular position, velocity, and acceleration of joint 2

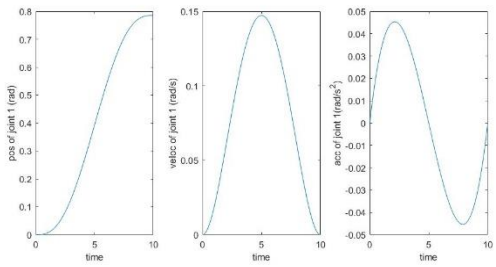


Figure 5: Angular position, velocity, and acceleration of joint 3

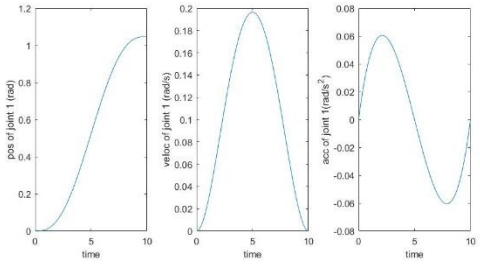


Figure 6: Angular position, velocity, and acceleration of joint 4

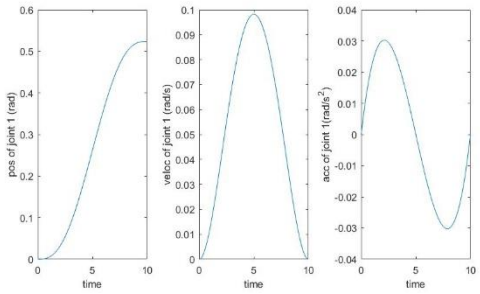


Figure 7: Angular position, velocity, and acceleration of joint 5

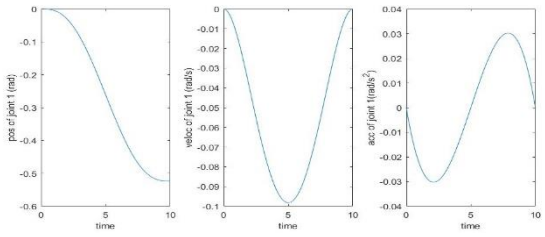


Figure 8: Angular position, velocity, and acceleration of joint 6

Use MATLAB's simulation model to do the trajectory planning of joint space, as shown in Figure 2 to Figure 7, It can be seen that the curve is smooth and continuous, and the robot can run smoothly.

B. Trajectory Planning in Cartesian Space

We use the quintic polynomial interpolation method with the starting point and end point of the straight line to obtain the straight line trajectory planning of the robot in Cartesian space.

$$p_i = \begin{pmatrix} 166 \\ 1200 \\ 575 \end{pmatrix} \text{ And } \quad p_f = \begin{pmatrix} 560 \\ 1200 \\ 960 \end{pmatrix}$$

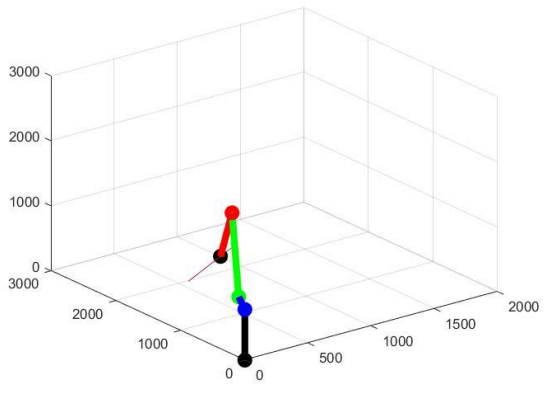


Figure 9: Robot moving in linear motion

We can see that the robot moves in smooth move, and to make sure that we will see how the joints angles change, with the inverse kinematics and Euler forward method we get angular position, velocity, and acceleration of joints as depicted by figures 10 to 14.

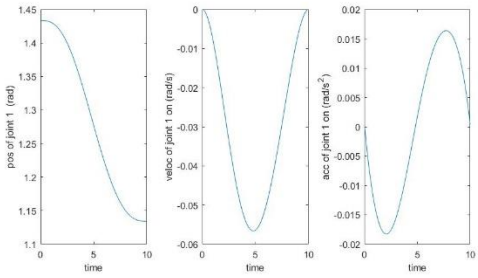


Figure 10: Angular position, velocity, and acceleration of joint 1 in linear path

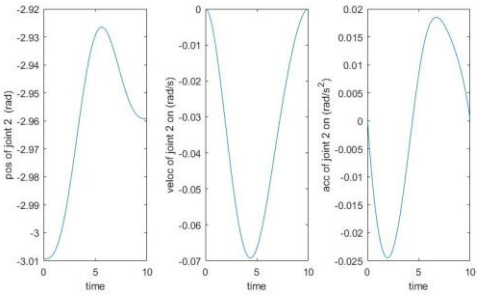


Figure 11: Angular position, velocity, and acceleration of joint 2 in linear path

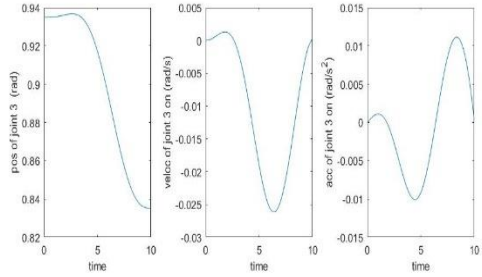


Figure 12: Angular position, velocity, and acceleration of joint 3 in linear path

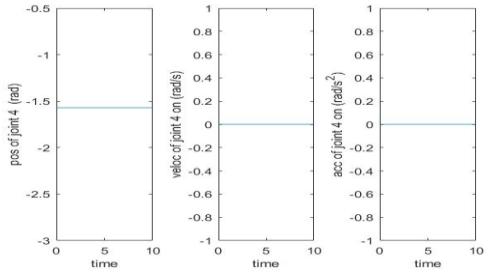


Figure 13: Angular position, velocity, and acceleration of joint 4 in linear path

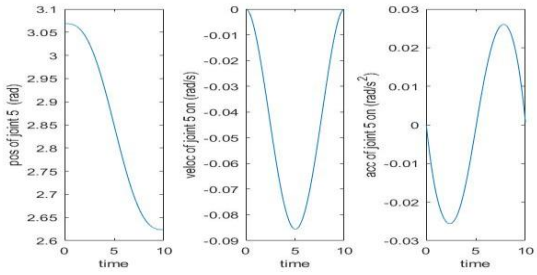


Figure 14: Angular position, velocity, and acceleration of joint 5 in linear path

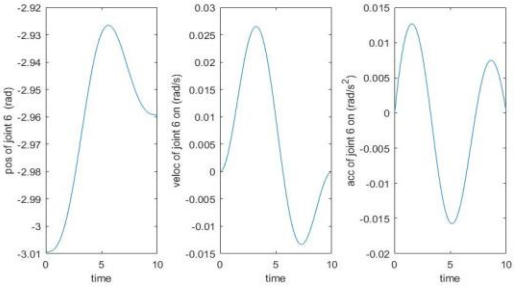


Figure 15: Angular position, velocity, and acceleration of joint 6 in linear path

Like we see, all the joints have a smooth path and that ensure again the smoothness of trajectory. With the algebraic form of the circle and the MATLAB code we will generate points that make up the path and every two points we use the quintic polynomial interpolation. The simulation is depicted by figure 16, where the end effector of the robot follows the circle.

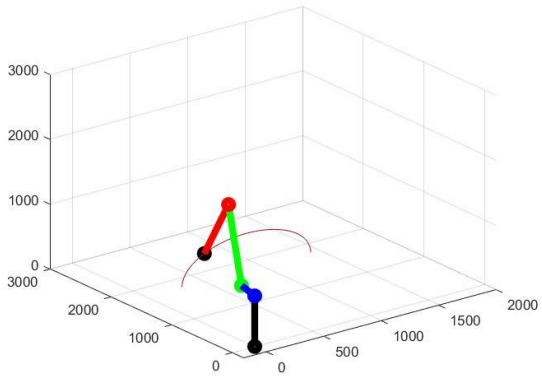


Figure 16: Robot Moves in a circular motion

V. VALIDATION WITH ROBOTSTUDIO

RobotStudio simulation software builds a robot model that is consistent with the model developed by MATLAB in order to validate the accuracy of the trajectory planning created by MATLAB, The result shown in Figure 7 and Figure 18.

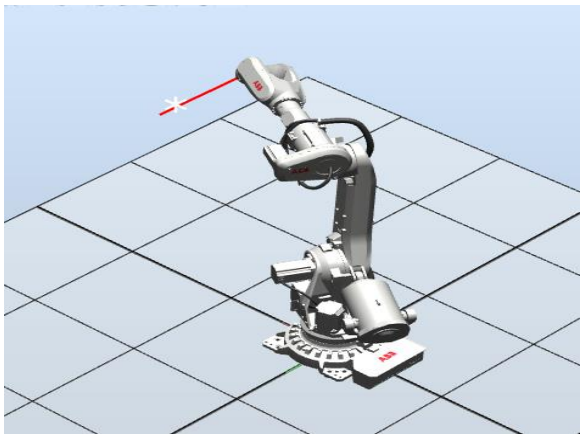


Figure 17: Robot moving in linear motion in RobotStudio

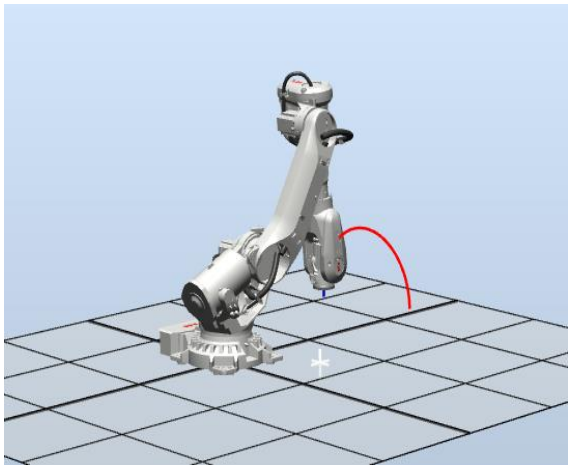


Figure 18: Robot Moves in a circular motion RobotStudio

It is clear that the two models are consistent, and this indicates the validity of the previously imposed model in tracking trajectory with MATLAB

IV. CONCLUSION

The forward and inverse kinematic modeling for ABB IRB 6700 Robot manipulator was performed analytically and validated using Matlab. Then, using a quintic polynomial method a trajectories both in joint and cartesian space are generated. The obtained simulation results show the smoothness of the generated trajectories, also the end effector of the robot follows the different generated trajectories in Cartesian space. Some validation results are given using RobotStudio to follow straight and circular path in the Cartesian space, which proves again the correctness of the forward and inverse kinematics. And since RobotStudio is built on the ABB virtual controller, which used to control ABB robots in real application, then the performed simulation on RobotStudio can be considered as realistic simulation.

REFERENCES

[1] Luigi biagiotti, calaudio melchiorri time optimal trajectory planning for redundant robot

[2] HAO Jianbao SONG Chun-Hua. Kinematics Analysis and Simulation of a 6-DOF Robot Based on MATLAB and RobotStudio 2018

[3]John J.Craig - Introduction to Robotics Mechanics and Control 3rd edition-Pearson Education, Inc. (2005)

[4]John J.Craig - Introduction to Robotics Mechanics and Control 3rd edition-Pearson Education, Inc. (2005), page 202-203