ME404
Robotics
Fall 2024-25

Under Supervision of

Dr. Omar Abdelaziz

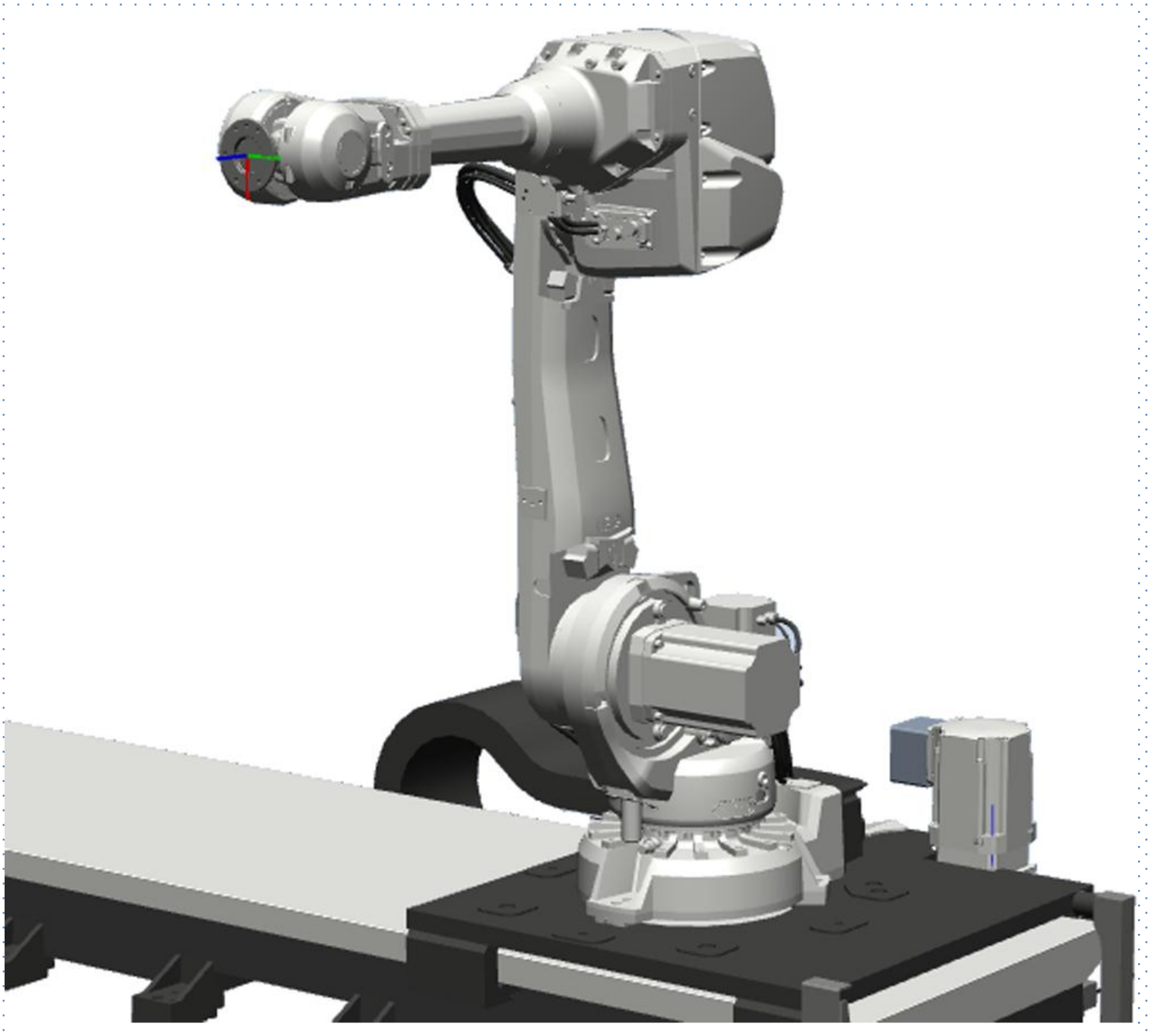
Submitted

by

Mohamed Abdelmoniem

201147

Report on ABB IRB 40-255
Robot Project using CoppeliaSim and python



Movement

Axis movements:	Working range:	Maximum speed:
Axis 1	+180° to -180°	175°/s
Axis 2	+150° to -90°	175°/s
Axis 3	+75° to -180°	175°/s
Axis 4	+400° to -400°	250° (20/2.50 has 360°)/s
Axis 5*	+120° to -125°	250° (20/2.50 has 360°)/s
Axis 6	+400° to -400°	360° (20/2.50 has 500°)/s

* Axis 5 for IRB 4600-20/2.50 +120°-120°

Electrical connections

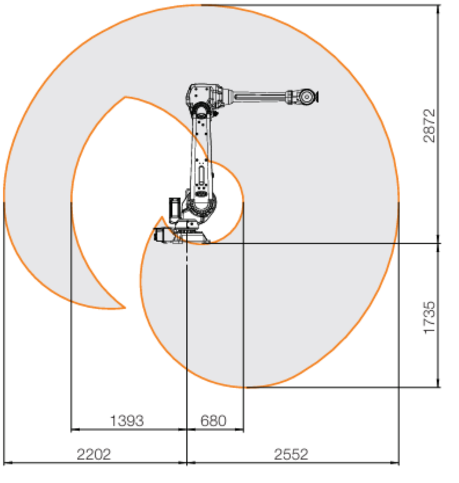
Supply voltage:	200-600 V, 50-60 Hz

Environment

Ambient temperature for mechanical unit:

During operation:	+5° C (41° F) to + 45°C (113°F)
During transportation and storage:	-25° C (-13° F) to +55° C (131° F)
For short periods (max 24 h):	up to +70° C (158° F)
Relative humidity:	Max 95%
Safety:	Double circuits with supervisions,

IRB 4600-40/2.55



•1-Robot Model and DH Table:

Initially, I referred to the Denavit-Hartenberg (DH) parameters provided in the robot's documentation. However, upon comparing these parameters with the robot's behavior in CoppeliaSim, I noticed discrepancies

- .I recalculated the **DH table** based on the robot's actual dimensions and joint parameters observed in CoppeliaSim.
- The revised DH table is provided below:

Link	$\boldsymbol{\theta_i}$	$\boldsymbol{d_i}$	$\boldsymbol{a_i}$	$\boldsymbol{\alpha_i}$
1	θ_1	d_1	a_1	$-\frac{\pi}{2}$
2	$\theta_2 - \frac{\pi}{2}$	0	a_2	0
3	θ_3	0	a_3	$-\frac{\pi}{2}$
4	θ_4	d_4	0	$\frac{\pi}{2}$
5	$\theta_5 + \pi$	0	0	$\frac{\pi}{2}$
6	θ_6	d_6	0	0

D1	0.690
D4	0.960
D6	0.135
A1	0.510
A2	0.900
a3	0.175

2. Forward and Inverse Kinematics

•Forward Kinematics (FK):
Using the revised DH table, I implemented a custom function to compute the forward kinematics of the robot. This function was verified and tested with known configurations.

•Inverse Kinematics (IK):
An inverse kinematics function was developed to compute joint angles for given end-effector positions in the task space.

- **Validation:** The results from the FK and IK functions were cross-validated to ensure accuracy. Both functions provided matching results, confirming the correctness of the calculations.

3-Task Implementation

•Defining Points in Task Space:
Several points were defined in the task space for the robot to follow. These points represent a specific trajectory required for the task.

•Linear Interpolation:
To create a smooth motion, linear interpolation was applied between the defined points. This interpolation divided the trajectory into smaller segments, ensuring precise movement.

- For each interpolated point, the IK function was used to compute the corresponding joint angles.

4-Drawing Implementation

•Adding a Drawing Plane:

A drawing plane was added to the simulation environment using **Add > Primitive Shape > Plane**. This plane was positioned within the robot’s workspace.

•Adding a Felt Pen:

A felt pen component was attached to the robot’s end effector. This was achieved via **Components > Modifiers > Felt Pen**. The pen was used to draw lines between the trajectory points.

5-Results

•The robot was successfully programmed to trace a trajectory in the shape of **"201147"**, representing my unique identifier.

•The final drawing was achieved with high accuracy, demonstrating the effectiveness of the FK and IK algorithms and the interpolation method.