German University in Cairo (GUC)
Faculty of Engineering & Materials Science (EMS)
Mechatronics Department (MCTR)

Robotics (EDPT1009) Word Document Template

10/19/2021

Project Milestone: MS#5

Team Number: #09

Milestone #5
ROBOTICS (EDPT1009) W'21 COURSE PROJECT

Trajectory Planning and Kinematic Modeling of a Robotic Manipulator

Team Member 01 Seif Eldin Refaat, Team Member 02 Zeina Hesham, Team Member 03 Yahia Mohamed, Team Member 04 Hazem Sherif, Team Member 05 Sarah El Shemy, Team Member 06   Omar Samir.

Abstract- This project focuses on developing a fully functional handling robot that operates as part of a coupled industrial robotic system. The robot's primary task is to manage cut pieces from a production line by picking them up and transporting them. Using a suction-based end effector, the robot will handle both the finished parts and any scrap material, ensuring an efficient workflow. The simulation, carried out in CoppeliaSim using a URDF model, will serve as a digital twin of the hardware. This integration will allow for seamless transition from the virtual environment to the real-world system, ensuring accurate synchronization and control between simulation and hardware.

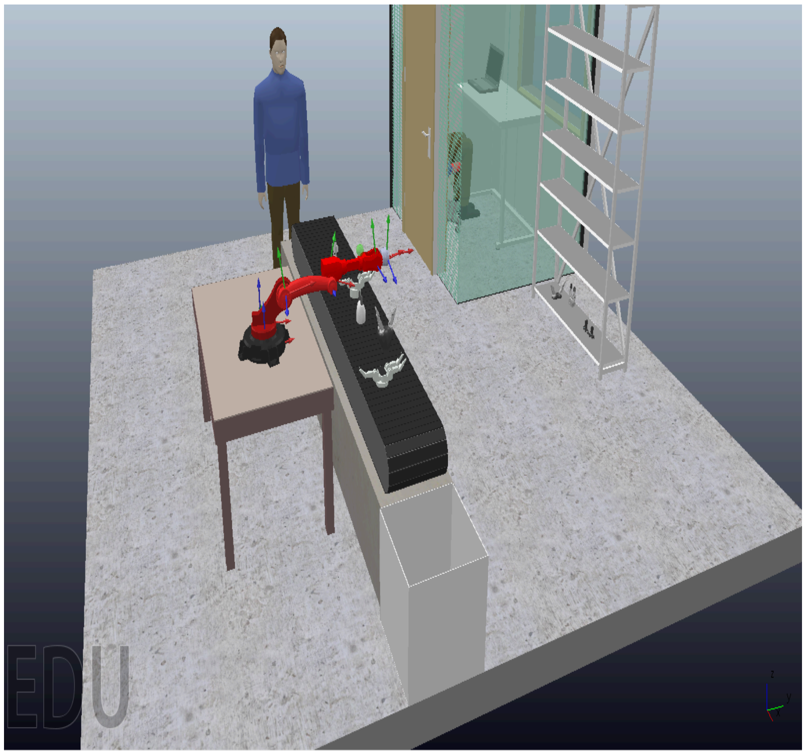


Figure 1 Robot in created environment

Keywords: DH Convention, Trajectory Planning, Robotic Manipulator, ForwardKinematics, Inverse Kinematics, Coppeliasim simulation.

1. INTRODUCTION

1.1 : Types of Robotic Arms and Their Applications

In the article *Robotic Arms: Different Types and When to Use Them* by Catherine Bernier, various types of robotic arms are discussed, including articulated, SCARA, Cartesian, and collaborative robots (cobots). Articulated arms, with their multiple joints, offer maximum flexibility and are used in industries like automotive, aerospace, and pharmaceuticals for tasks requiring high dexterity and reach. Six-axis robots, a type of articulated arm, are particularly versatile for complex operations. SCARA robots are suitable for high-speed, low-flexibility tasks such as pick-and-place operations.

Cobots, designed to work alongside humans, are becoming increasingly popular in industries that require close human-robot collaboration. These robots prioritize safety and flexibility, allowing for seamless integration into workspaces where human interaction is necessary. For our project, articulated robotic arms are the most applicable due to their flexibility in performing handling tasks.

1.2: Robotic Arms in Industry 4.0

The article *The Transformative Power of Industrial Robot Arms in Industry 4.0* highlights how robotic arms contribute to the development of smart factories by integrating digital technologies with physical production. In Industry 4.0, robot arms equipped with sensors and software can communicate with other machines, gather real-time data, and optimize production processes autonomously. This integration helps reduce human error, improve productivity, and ensure consistent product quality.

Robotic arms are pivotal in predictive maintenance, enabling real-time monitoring and fault detection to minimize downtime. Additionally, emerging technologies such as augmented reality (AR) and virtual reality (VR) are enhancing the programming and simulation of robots, allowing operators to interact with machines in a more immersive way. This concept is directly relevant to the project's digital twin approach, where the simulated robot's performance will mirror the hardware in real-time, ensuring seamless synchronization.

1.3: Design and Control of pick and PlaceRobots

The article *Robot Arm for Pick and Place Process* presents the design and control of a 4 degree-of-freedom (DOF) robotic arm intended for autonomous pick-and-place operations. This arm, controlled via an Arduino microcontroller, uses a Proportional-Integral-Derivative (PID) controller to regulate the joints' movement, ensuring smooth and precise handling. The robot's kinematic model includes both forward and inverse kinematics, allowing for accurate end-effector positioning.

The application of such control systems aligns with our project, where precise movement and grip control are critical for handling cut materials. By simulating similar control systems in CoppeliaSim, we ensure the robot can perform complex handling tasks with minimal error, an essential aspect of optimizing real-world industrial processes.

1.4: Industrial Robotics and Smart Manufacturing

The *Pick and Place Robotic Arm: A Review Paper* further explores the impact of AI and machine learning on robotics, transforming them from static machines into adaptive systems. These technologies enable robots to continuously learn and improve workflows, leading to increased efficiency and reduced costs. Industrial robots, particularly those used for material handling, benefit from innovations such as AI-powered data analytics, which allow for real-time decision-making and process optimization.

This insight reinforces the importance of incorporating AI into robotics, potentially enabling future iterations of our project to improve through machine learning. As robotic arms evolve, their ability to dynamically adjust to new tasks and environments will further enhance automation in smart factories.

1.5: Selected Application: Handling Robot for Cut Materials

Other articles discuss the coordinated use of robotic arms for cutting and handling materials. In this selected application, one robot equipped with a cutting tool precisely cuts materials such as cardboard or wood. A second robot, equipped with a suction-based end effector, picks up the cut materials and transports them for further processing or packaging. By automating both the cutting and handling processes, the system enhances efficiency and accuracy.

For our project, we are tasked with simulating the second robot, which handles the cut pieces using a suction-based end effector. This robot must carefully pick up and transport the cut material without damaging it, ensuring that both the blank and scrap materials are handled appropriately. The use of articulated arms and the integration of a suction-based end effector are central to this task, as they provide the flexibility and precision needed to optimize the handling process.

2. Methodology

2.1: Draft Flow

1. **URDF Model Import and Configuration**

- The first step is to import the URDF model of the handling robot into CoppeliaSim.
- The robot's joints and degrees of freedom (DoF) will be verified to ensure accurate movement and flexibility.
- The suction-based end effector will be assigned to the robot for handling tasks.

2. **Environment Setup**

- A simulation environment resembling the production line will be created, including a conveyor belt for cut materials.
- Racks for scrap materials will also be included to ensure proper handling of both reusable and scrap pieces.
- Workspace boundaries will be defined to prevent collisions and ensure safe robot movement.

3. **End-Effector Configuration**

- The suction gripper will be calibrated to securely pick up cut materials.
- Vacuum sensors will be integrated to monitor grip strength, ensuring the materials are held securely without damage.

4. **Task Programming (Pick and Place Operations)**
    - The robot will be programmed to perform pick-and-place operations, including picking up cut materials and placing them on the conveyor belt or designated area.
    - The robot will differentiate between scrap and usable materials, handling both appropriately.
    - A separate sequence will handle scrap collection, placing it into designated racks.
5. **Feedback Control System**
    - A PID control system will regulate the robot's movements, ensuring precision and correct grip pressure during handling.
    - Proximity sensors and force sensors will monitor movement and material handling to prevent damage

6. **Simulation Testing in CoppeliaSim**
    - Multiple simulation tests will be run to validate grip strength, movement accuracy, and placement efficiency.
    - Simulations will include testing with different material sizes and orientations to optimize the robot's handling capabilities.
7. **Performance Tuning**
    - The robot's parameters, such as grip pressure and movement speed, will be fine-tuned based on simulation results.
    - Issues like material slippage or delays in the pick-and-place cycle will be addressed through iterative tuning.
8. **Fault Handling and Error Management**
    - The robot will be programmed to handle faults, such as unsuccessful picking attempts, with retries or alerts for manual intervention.
    - An error-detection mechanism will pause operations if issues with grip or movement occur.
9. **Digital Twin Integration with Hardware**
    - The simulation in CoppeliaSim will be merged with the physical hardware to create a **digital twin**. This will ensure the simulation mirrors the hardware's real-time operations.
    - Data from the physical robot, such as joint positions, force feedback, and sensor readings, will be integrated into the simulation to maintain synchronization.
    - The digital twin will allow us to monitor the system in real-time, test scenarios virtually, and transfer improvements from the simulation directly to the hardware.
    - The hardware will be controlled by the same algorithms used in the simulation, ensuring that changes made in the simulation are immediately applicable to the physical system.
10. **Final Validation**
- After confirming the synchronization between the simulation and hardware, we will conduct final tests to ensure smooth operation of the digital twin.
- The handling robot's performance will be validated across various production scenarios, ensuring reliable operation within the coupled system.

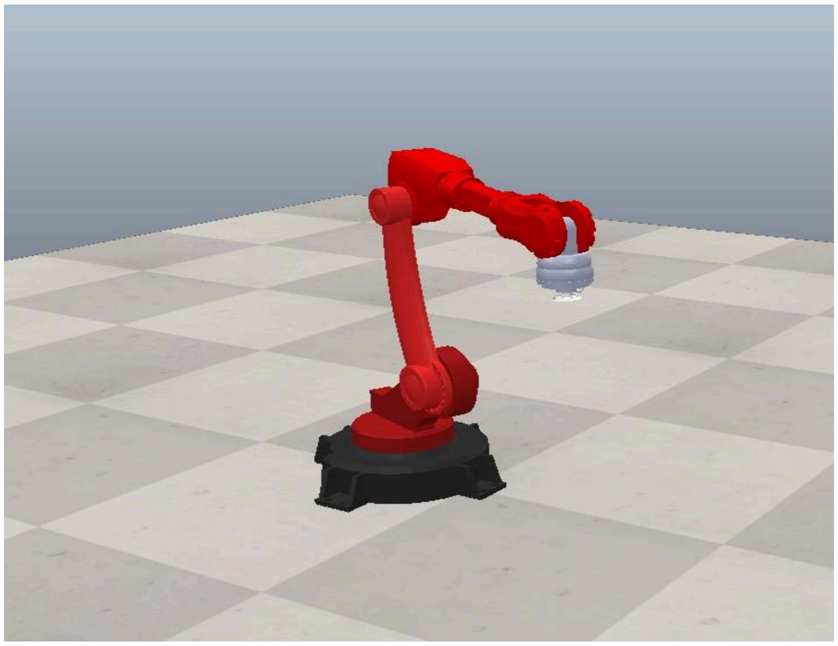


Fig1.1 Screenshot of the CAD Model

2.2 Robot Design

This part focuses on the design of a fully functional handling robot that operates within an industrial robotic system. Its primary task is to pick up and transport cut pieces from a production line. Using a suction-based end effector, the robot efficiently handles both finished products and scrap materials to optimize workflow. The robot's performance is simulated in CoppeliaSim, using a URDF model as a digital twin of the hardware. This digital twin allows for smooth integration and synchronization between the virtual simulation and the actual system, ensuring accurate control and coordination.

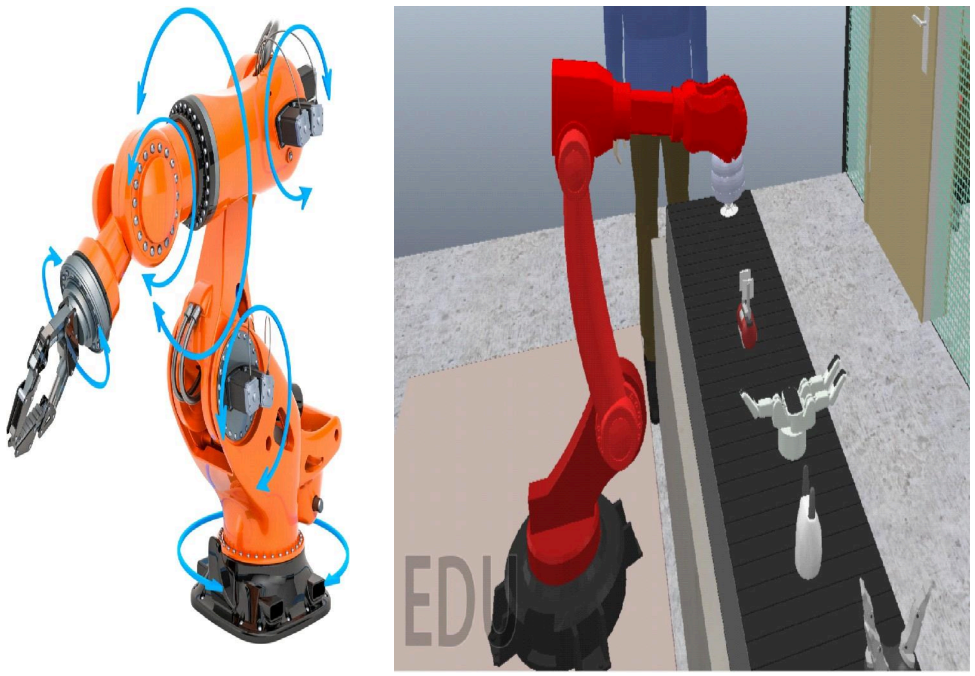


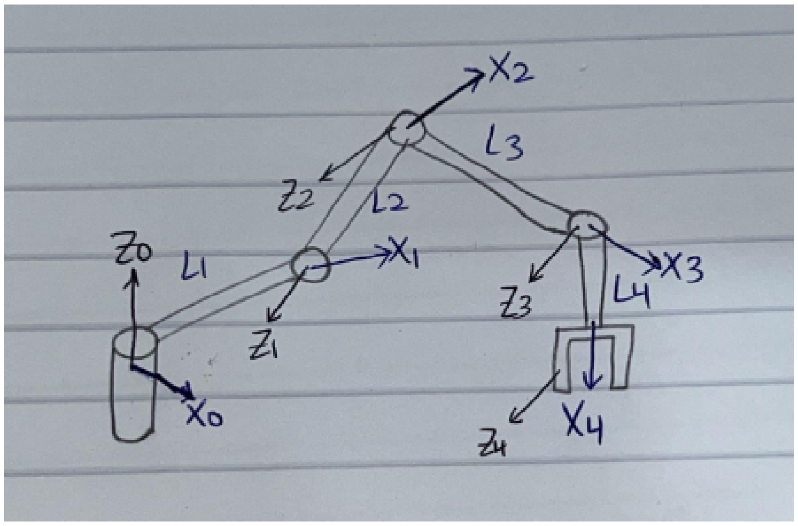
Fig. 1.2 Coupled Industrial Robotic Arm

In this paper we will cover many aspects of the robot built. In section I, we cover the hardware components used and the circuit diagram and design. Section II, the assignment of the robot's frame is going to be presented. Section III, is going to present the DH-convention analysis of the robot including the DH-Convention table and the DH-Convention Final Matrix. Then Finally we highlight and include the simulation results along with our conclusion and future recommendations.

Position Kinematics

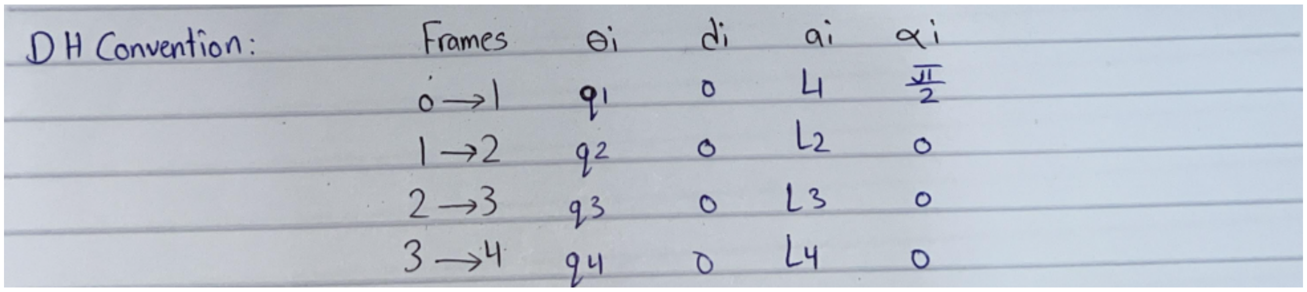
In our project, the hardware design of the handling robot incorporates key elements to enable precise and efficient material handling within an industrial setting. The robot features a 6-degree-of-freedom (DOF) arm, allowing for versatile movement and manipulation of cut pieces on the production line. Utilizing a suction-based end effector, the robot can securely grip and transport both finished products and scrap materials.

The robotic arm is powered by stepper motors, with each joint controlled through dedicated motor drivers to ensure precise articulation. To achieve smooth motion control, an Arduino microcontroller regulates the stepper motors, employing a Proportional-Integral-Derivative (PID) control strategy similar to that used in autonomous robotic arms. The PID controller enables accurate joint movement by minimizing error in the robot's positioning, providing the necessary precision for the handling tasks.



- **Topic 01: Forward Position Kinematics**

- **Topic 02: DH convention**



DH Convention:

Frames	θ_i	d_i	a_i	α_i
$0 \rightarrow 1$	q_1	0	L_1	$\frac{\pi}{2}$
$1 \rightarrow 2$	q_2	0	L_2	0
$2 \rightarrow 3$	q_3	0	L_3	0
$3 \rightarrow 4$	q_4	0	L_4	0

- **Topic 03: Final Matrix**

```
X = (L1 * sp.cos(q1) +
L2 * sp.cos(q1) * sp.cos(q2) -
L3 * sp.sin(q2) * sp.sin(q3) * sp.cos(q1) +
L3 * sp.cos(q1) * sp.cos(q2) * sp.cos(q3) +
L4 * (-sp.sin(q2) * sp.sin(q3) * sp.cos(q1) +
sp.cos(q1) * sp.cos(q2) * sp.cos(q3)) *
sp.cos(q4) +
L4 * (-sp.sin(q2) * sp.cos(q1) * sp.cos(q3) -
sp.sin(q3) * sp.cos(q1) * sp.cos(q2)) *
sp.sin(q4))

Y = (L1 * sp.sin(q1) +
L2 * sp.sin(q1) * sp.cos(q2) -
L3 * sp.sin(q1) * sp.sin(q2) * sp.sin(q3) +
L3 * sp.sin(q1) * sp.cos(q2) * sp.cos(q3) +
L4 * (-sp.sin(q1) * sp.sin(q2) * sp.sin(q3) +
sp.sin(q1) * sp.cos(q2) * sp.cos(q3)) *
sp.cos(q4) +
L4 * (-sp.sin(q1) * sp.sin(q2) * sp.cos(q3) -
sp.sin(q1) * sp.sin(q3) * sp.cos(q2)) *
sp.sin(q4))

Z = (L2 * sp.sin(q2) +
L3 * sp.sin(q2) * sp.cos(q3) +
L3 * sp.sin(q3) * sp.cos(q2) +
L4 * (-sp.sin(q2) * sp.sin(q3) + sp.cos(q2) *
sp.cos(q3)) * sp.sin(q4) +
L4 * (sp.sin(q2) * sp.cos(q3) + sp.sin(q3) *
sp.cos(q2)) * sp.cos(q4))
```

- **TOPIC 04 (Inverse Kinematics)**

Inverse kinematics (IK) is the process of determining the joint angles of a robotic arm that correspond to a desired end-effector position and orientation in Cartesian space. Unlike forward kinematics, which computes the position and orientation given the joint angles, IK is often more complex, requiring iterative numerical techniques for solutions, especially for multi-DOF robots.

The Newton-Raphson method approximates the solution by iteratively refining an initial guess q0. At each iteration, the method linearizes f(q) around the current estimate qk using a first-order Taylor expansion:

$$x_{\text{desired}} \approx f(q_k) + J(q_k)(q_{k+1} - q_k)$$

Here, J(qk) is the Jacobian matrix evaluated at qk representing the partial derivatives of f(q)f(q)f(q) with respect to the joint variables. Rearranging to solve for qk+1:

$$q_{k+1} = q_k + J(q_k)^{-1}(x_{\text{desired}} - f(q_k))$$

The Algorithm is as follows:

1. **Initialize**:
    - Start with an initial guess for the joint angles, q0.
    - Define a convergence tolerance (delta)

2. **Iterate**:

- Compute the current end-effector position $\mathbf{x}_k = f(\mathbf{q}_k)$.
- Calculate the error $\mathbf{e} = \mathbf{x}_{\text{desired}} - \mathbf{x}_k$.
- If e < delta stop; the solution has converged.
- Evaluate the Jacobian matrix, $J(\mathbf{q}_k)$.
- Update the joint angles using: $\mathbf{q}_{k+1} = \mathbf{q}_k + J(\mathbf{q}_k)^{-1}\mathbf{e}$
- Repeat until convergence.

3. **Check Solution**:

- Validate that the computed joint angles satisfy the desired end-effector pose within acceptable tolerances.

4. Considerations

- **Jacobian Inversion**: Direct inversion of J(qk) is computationally expensive and prone to numerical instability and inversion of a non-square Jacobian matrix is incorrect. Instead, methods like the **pseudo-inverse** are often used for stability: $J^\dagger = (J^T J)^{-1} J^T$
- **Singularities**: If the determinant of J(qk) approaches zero, the system is near a singular configuration. Specialized strategies (e.g., damped least squares) are used to handle such cases.
- **Initial Guess**: The success of the Newton-Raphson method depends heavily on a good initial guess, as poor initialization can lead to divergence.

Velocity Kinematics

Velocity kinematics is concerned with how the velocities of the joints of a robot correspond to the velocity of its end-effector. Forward velocity kinematics describes the relationship between joint velocities and the end-effector velocity, while inverse velocity kinematics works in the opposite direction, determining the required joint velocities for a desired end-effector velocity.

- **TOPIC 05 (Forward Velocity Kinematics)**

In forward velocity kinematics, the joint velocities are mapped to the end-effector velocity using the Jacobian matrix, which is a matrix of partial derivatives that relates changes in joint space (joint velocities) to changes in Cartesian space (end-effector velocity). The forward velocity kinematic equation can be written as: $\dot{x} = J(q) \cdot \dot{q}$

Where:

- $\dot{x}$ is the end-effector velocity (a vector in Cartesian space).
- $\dot{q}$ is the vector of joint velocities (in joint space).
- $J(q)$ is the Jacobian matrix, which depends on the current joint configuration q.

- **TOPIC 06 (Inverse Velocity Kinematics)**

Inverse velocity kinematics, on the other hand, seeks to determine the joint velocities required to achieve a desired end-effector velocity. The inverse velocity kinematic equation is: $\dot{q} = J(q)^{-1} \cdot \dot{x}$

Where:

- $\dot{x}$ is the end-effector velocity (a vector in Cartesian space).
- $\dot{q}$ is the vector of joint velocities (in joint space).
- $J(q)^{-1}$ is the inverse (or pseudo-inverse) of the Jacobian matrix.

In practice, the Jacobian matrix may not be square or invertible, particularly when the robot is in a singular configuration (where the determinant of the Jacobian approaches zero). In these cases, a pseudo-inverse is used: $J^\dagger = (J^T J)^{-1} J^T$

The forward and inverse velocity kinematics equations are crucial for controlling the motion of robotic arms, as they relate the velocities of the joints to the velocity of the end-effector in Cartesian space. These relationships are captured by the Jacobian matrix, which, depending on the robot's configuration, can be used to calculate joint velocities from desired end-effector velocities or vice versa. The Newton-Raphson method, pseudo-inverses, and other techniques are used to solve these equations iteratively, ensuring that the robot can achieve precise movements in its workspace.

2.3 Trajectory Planning

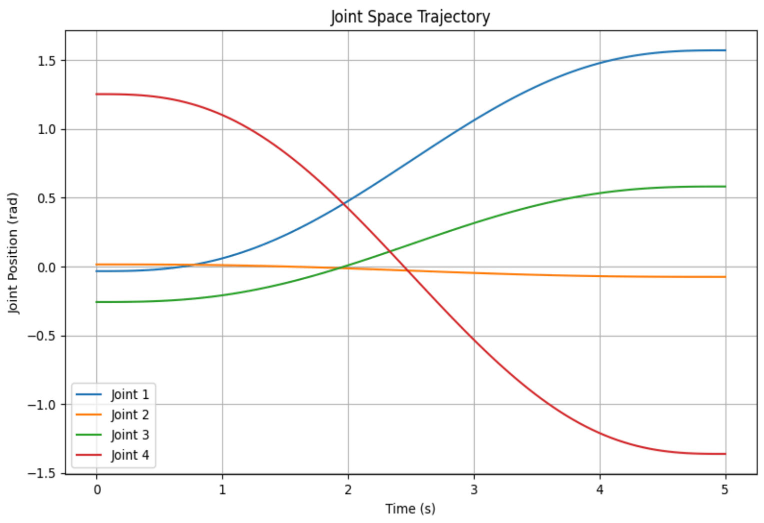
Joint Space Trajectory

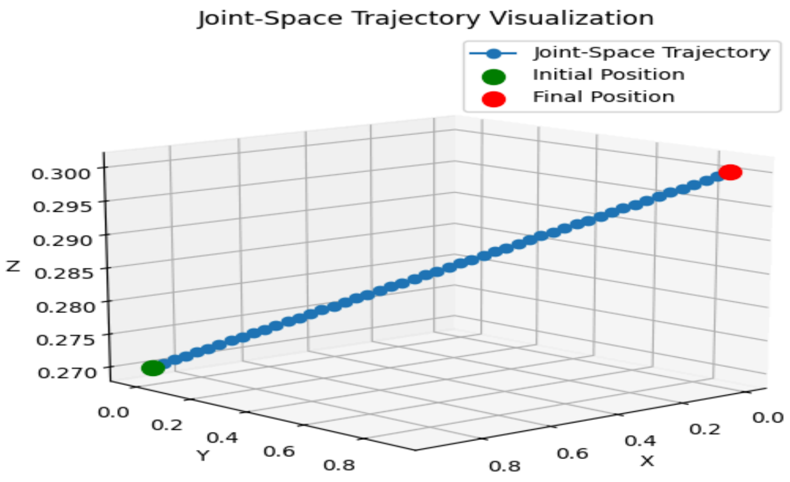
For the joint space trajectory, we used 5th order polynomial equations to make the motion of the robot smoother than when using 3rd order polynomial equations. We assumed that the robot starts from rest and ends at rest. we also assumed that the initial and final acceleration are zeros.

The equations of the 4 joints of the robot are as follows:

```
q1 = 0.00308013698536483*t**5 -
0.0385017123170604*t**4 +
0.128339041056868*t**3 -
0.0334416864159532

q2 = -0.000173412364935744*t**5 +
0.00216765456169681*t**4 -
0.00722551520565602*t**3 +
0.0152213360253963

q3 = 0.0016113865448219*t**5 -
0.0201423318102737*t**4 +
0.0671411060342458*t**3 -
0.257649789992278

q4 = -0.00501830654071967*t**5 +
0.0627288317589958*t**4 -
0.20909610586332*t**3 +
1.25254767574043
```

As a visualization for the joint space trajectory, I provided two different graphs the first one shows the change of the joints position relative to the time and the second one shows a 3d view for the trajectory motion done by the robot, given that the time required for the trajectory is 5 seconds with a time sample of 0.1 seconds and initial position = [0.91971765 , -0.03076838 , 0.27], final position = [0, 0.9, 0.3]:



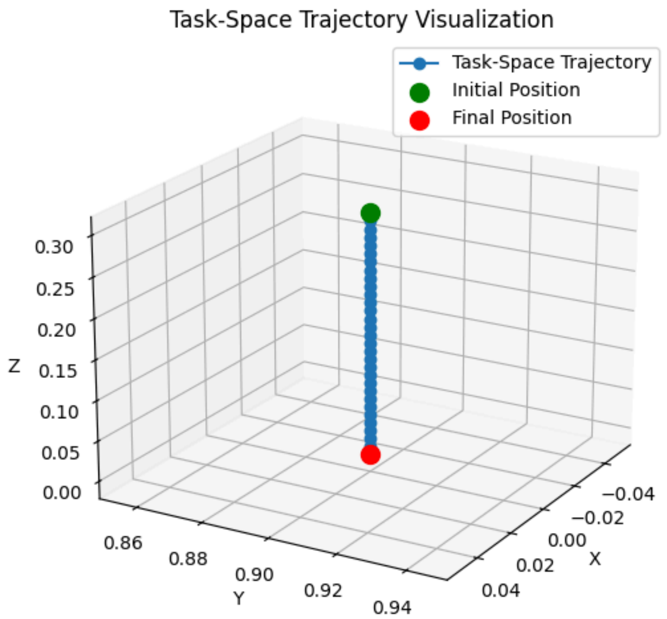


Task Space Trajectory

For the task space trajectory, we chose to perform it as a straight-line trajectory. And since the trajectory's initial position = [0, 0.9, 0.3] and final position = [0, 0.9, 0] the resulting equations for this line are:

$$X = 0$$
$$Y = 0.9$$
$$Z = 0.3 - 0.1t$$

As a visualization for the task space trajectory, I provided a 3d view for the trajectory motion done by the robot, given that the time required for the trajectory is 3 seconds with a time sample of 0.1 seconds:



For the validation of CoppeliaSim: we performed the joint space and task space trajectories together with the inputs discussed above and the resulting motion of the robot and final position correctly matched the calculations performed.

Conclusion

The reviewed literature provides a comprehensive overview of the current state of industrial robotics, particularly focusing on robotic arms and their applications in smart manufacturing environments. The selected application of handling cut pieces using a robotic arm with a suction end effector corresponds directly to the insights gained from these articles. By integrating articulated robotic arms with advanced control systems, sensors, and predictive maintenance capabilities, we aim to simulate a highly efficient and flexible robot in CoppeliaSim that can later be synchronized with its real-world counterpart.

The milestone successfully demonstrated the feasibility of the handling robot within the coupled industrial robotic system, achieving efficient material transportation and workflow optimization through the integration of the digital twin. The simulation results confirmed the system's ability to maintain accurate control and coordination, validating the design choices for hardware components and the end effector. Future recommendations include refining the control algorithms to further enhance synchronization, expanding the robot's functionality to accommodate more complex tasks, and integrating additional sensors for improved feedback and precision. Further milestones will focus on these enhancements and the implementation of a fully functional robotic system in a real-world setting.

References

- **Bernier, C. (2024, September 4)**. Robotic Arms: Different Types and When to Use Them. Content for Cobot.
- **RoboDK. (2023, August 1).** The Transformative Power of Industrial Robot Arms in Industry 4.0. RoboDK Blog
- https://ijasre.net/index.php/ijasre/article/view/1149/1697
- https://www.irjet.net/archives/V8/i2/IRJET-V8I2311.pdf
- **Dzedzickis, Andrius & Subaciute-Zemaitiene, Jurga & Šutinys, Ernestas & Prentice**, Urte & Bučinskas, Vytautas. (2021). Advanced Applications of Industrial Robotics: New Trends and Possibilities. Applied Sciences. 12. 135. 10.3390/app12010135.
- **IFR Presents World Robotics Report 2020**—International Federation of Robotics. Available online: https://ifr.org/ifr-pressreleases/news/record-2.7-million-robots-work-in-factories-around-t h e-globe
- **Agrawal, A., Gans, J.S. and Goldfarb, A., 2019**. Artificial intelligence: the ambiguous labor market impact of automating prediction. Journal of Economic Perspectives, 33(2), pp.31-50.
- **Wang, Weizheng. (2020).** Applied Research of Industrial Robots in Automotive Intelligent Manufacturing Production Line. Journal of Physics: Conference Series. 1550. 042061. 10.1088/1742-6596/1550/4/042061.