Development and Optimization of a Robotic Arm Prototype

1 Introduction

1.1 Overview of the Robot Manipulator

This documentation presents the design and simulation of a 6-DOF articulated robotic arm prototype intended for an industrial airplane manufacturing factory. The robot manipulator has been specifically designed to operate within a factory layout divided into multiple zones, each dedicated to different stages of the manufacturing process. At the end of each zone, semi-finished airplane components are produced, which are then transferred by the robotic arm to the assembly area for further processing.
The robotic arm serves as a key element of the factory's automation system, facilitating the smooth transition of components from one production stage to another. This system reduces human labor, optimizes workflow, and ensures precise handling of the semi-finished products. The manipulator's movement is crucial to maintaining production efficiency, as it must be able to navigate the factory's environment and place components accurately in the designated assembly area.

1.2 Motivation and Purpose

The motivation behind designing this robot manipulator is to create a prototype of the real robotic arm, enabling us to explore and test multiple design propositions, optimizations, and configurations without the risk of damaging the actual model. By developing a virtual or prototype version first, we can simulate various scenarios and evaluate the robot's performance in different conditions before moving forward with modifications to the real model.
This approach allows us to refine the robot's design iteratively, ensuring that any modifications or enhancements can be tested in a controlled environment. It also provides valuable insights into the potential areas for optimization, such as improving the range of motion, efficiency, and overall performance of the robot in the factory setting. Ultimately, this prototype will serve as a critical step in achieving a more robust and optimized robotic arm for the factory's automation system, ensuring its reliability and effectiveness in handling semi-finished components between production zones and the assembly area.

2 Workspace Analysis

2.1 Allocated Space

- **Base Dimensions:** 90mm x 90mm
- **Height:** 400mm

These two dimensions (**base** and **height**) define the limits within which the robot must fit.

2.2 Robot Dimensions

- **Base Shape:** Circular face with a diameter of 90mm.
- **Height:** 380mm

2.3 Analysis

Fit within Space: The robot has a circular base with a diameter of 90mm, which fits exactly within the allocated 90mm x 90mm base dimensions of the allocated space. This confirms that the robot fits well within the footprint of the available space in terms of base dimensions.

Height Considerations: The robot's height is 350mm, which is slightly less than the allocated space's height of 360mm. This ensures that the robot will not exceed the space's vertical limits and provides a small margin (10mm) for any adjustments or tolerances needed in the design.

The table below presents a comparative analysis of two potential robotic manipulator solutions: **Custom-Built Robot (From Scratch)** and **SO-ARM 100 (Existing Robot)**. Each proposal is evaluated against key criteria, including **range of motion, design flexibility, development effort, component selection, and risk of failure**. The goal is to determine which option best aligns with our project requirements while ensuring optimal performance, adaptability, and feasibility.

Criteria	**Custom-Built Robot (From Scratch)**	**SO-ARM 100 (Existing Robot)**	**Requirements**	**Approved Proposal**
Range of Motion (Z-axis)	Maximum Z = 350mm	Maximum Z = +500mm	Should not exceed 360mm	**Custom-Built Robot** (Meets the requirement)
Base Diameter	Diameter = 90mm	Diameter = 120mm	Sould not exceed 90mm	**Custom-Built Robot** (More adaptable)
Design Flexibility	**Fully customizable**, we can modify dimensions, materials, and structure to fit our needs.	**Limited customization**, we must work with the existing structure and constraints.	Ability to adapt the robot's design to our specific needs.	**Custom-Built Robot** (More adaptable)
Development & Study Effort	Requires a **full study from scratch** (design, modeling, simulation, testing, and validation).	**URDF files are available**, allowing for **quick simulation and testing**.	AMinimize development time and effort.	**SO-ARM 100** (Faster implementation)
Component Selection Freedom	**Full control over component selection** (motors, control board, etc.).	**Limited to existing components**, which may not be optimal for our application.	Ability to choose components best suited for our needs.	**Custom-Built Robot** (More flexibility)
Risk of Failure	Higher (new system development)	Lower (proven technology)	Highly efficient system	**SO-ARM 100** (More flexibility)

Table 1: `Comparison table`

2.4 Conclusion

Based on our requirements, the most **convenient proposal** will be selected by balancing **customization, development time, component selection, and system reliability**. While the **Custom-Built Robot** offers greater flexibility and adaptability, the **SO-ARM 100** ensures faster implementation with lower development risk. The final decision will depend on prioritizing customization versus ease of integration.

3 Robot Modeling in CATIA

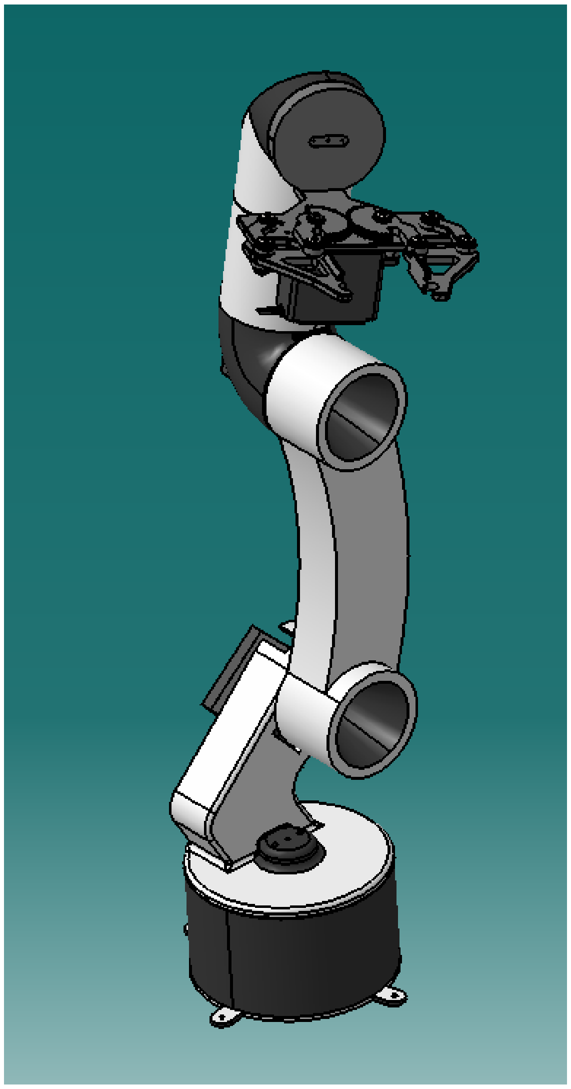


Figure 1: Robot Modeling in CATIA

3.1 CAD Model Description

In this step, each component of the robot manipulator was designed individually in CATIA, with the goal of 3D printing the parts for physical assembly. The process involved:

- **Component Modeling**: Each segment of the robotic arm, including the base, links, and end-effector, was created as a separate 3D model. The dimensions and geometries were carefully defined to ensure smooth assembly and proper range of motion.
- **Material Selection**: Since the parts will be 3D printed, materials like PLA or ABS were considered due to their lightweight properties and ease of printing. Strength and durability were also factored in, especially for load-bearing components.
- **Kinematic Considerations**: The CAD model was designed with revolute joints, ensuring that each joint had the necessary degrees of freedom. The placement of motors and actuators was also integrated into the design.
- **Assembly in CATIA**: After designing the individual components, they were assembled in CATIA's Assembly Design workbench. This allowed for precise positioning of each part, ensuring proper alignment and realistic motion

constraints. The assembly step was crucial for detecting interferences and making necessary adjustments before moving to physical manufacturing.

- **Export for Simulation**: Once finalized, the CAD files were exported in STEP format to be imported into Simscape Multibody, enabling realistic simulation and kinematic validation.

3.2 Individual Component Modeling

Each part of the robot was created separately in CATIA. This included the robot's base, arms, joints, and end-effector. Each component was modeled with the precise dimensions and features required for the robot's functionality and 3D printing.

3.3 Assembly Design

After creating all the individual parts, they were assembled together in CATIA's Assembly Design module. This stage focused on ensuring that all components fit together correctly, maintaining alignment and proper function.

3.4 Assembly Constraints

- **Coincidence Constraints:** Used to ensure that surfaces or points on different components align and remain in a fixed relationship.
- **Coaxiality Constraints:** Applied to ensure that cylindrical parts, such as joints or shafts, remain aligned along the same axis for correct movement and operation.

4 Simulation of the Articulated Arm Robot in MATLAB

To ensure the proper functioning of the articulated arm robot, a comprehensive simulation was conducted using **Simscape Multibody** in MATLAB. This approach allowed for an accurate representation of the robot's physical behavior, enabling precise analysis of its motion and operational feasibility. The simulation process was carried out step by step, starting with the **importation of CAD models** of the robot's components and culminating in the evaluation of the **end-effector's range of motion**.

4.1 Modeling the Robot in Simscape Multibody

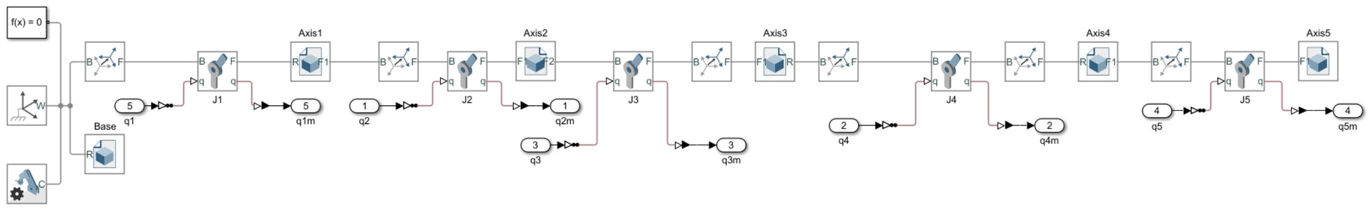


Figure 2: Robot Modeling ins Simscape Multibody

The first step in the simulation involved importing the CAD files of the robot's structural components, which were previously modeled in CATIA. Each component was defined as a separate body within the Simscape Multibody environment, preserving its physical characteristics such as mass and inertia. The components were then assembled using revolute joints, which provide the required degrees of freedom for rotation at each articulation. The correct definition of these joints was crucial to ensuring that the simulated motion accurately mirrored real-world behavior.
After constructing the mechanical model, the robot was transformed into a rigid body tree representation. This step allowed for the implementation of forward kinematics, enabling precise control of the joint positions and subsequent movement of the entire structure. By defining a rigid body tree, the robot could be manipulated using MATLAB functions designed for robotic systems, making it easier to analyze motion trajectories and optimize joint parameters.

4.2 Actuating the Joints and Observing Motion

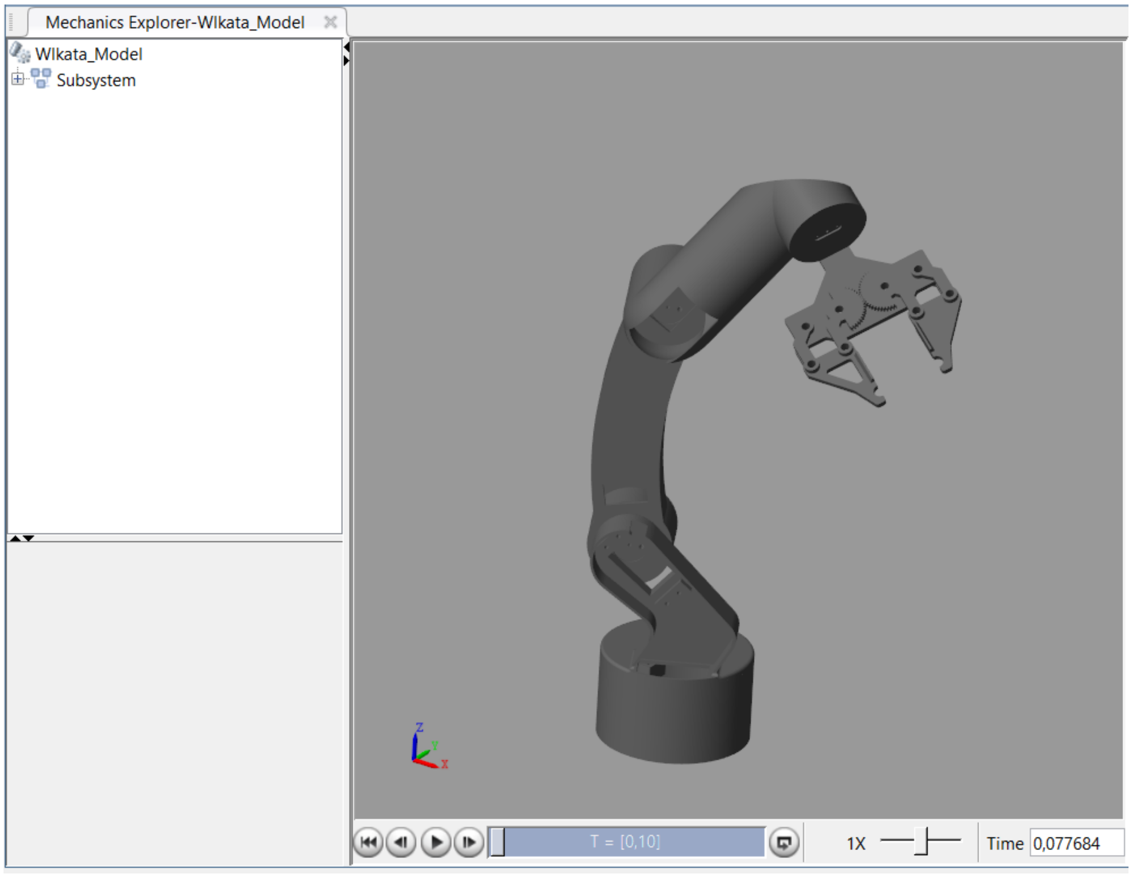


Figure 3: Simulation of the robot

Once the structure was defined, the revolute joints were actuated to simulate movement. This was achieved by applying input torque or angular velocity to each joint, allowing the robot to perform predefined motions. The primary goal of this step was to ensure that the kinematics of the robot were functioning correctly, with smooth and continuous articulation between the joints. Additionally, various joint configurations were tested to evaluate the overall flexibility and reach of the robotic arm.

4.3 Evaluating the End-Effector's Motion

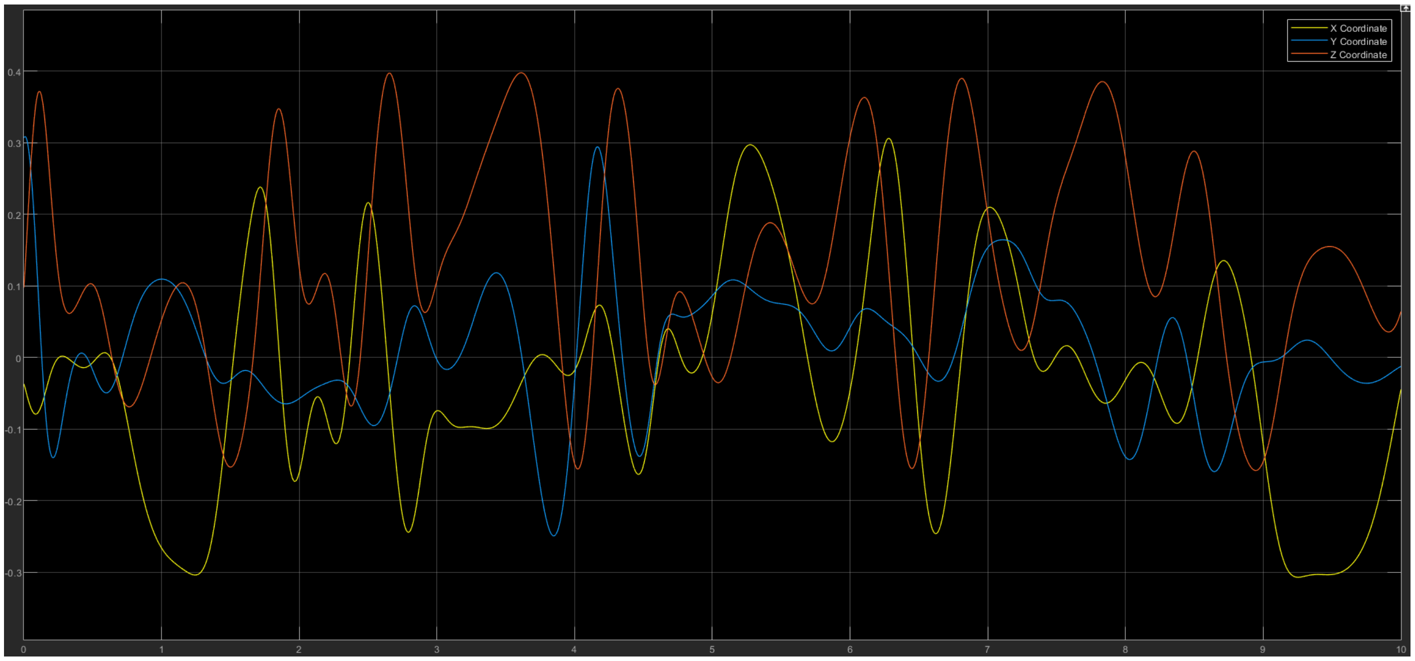


Figure 4: Range of motion of the End-Effector in Cartesian Coordinates

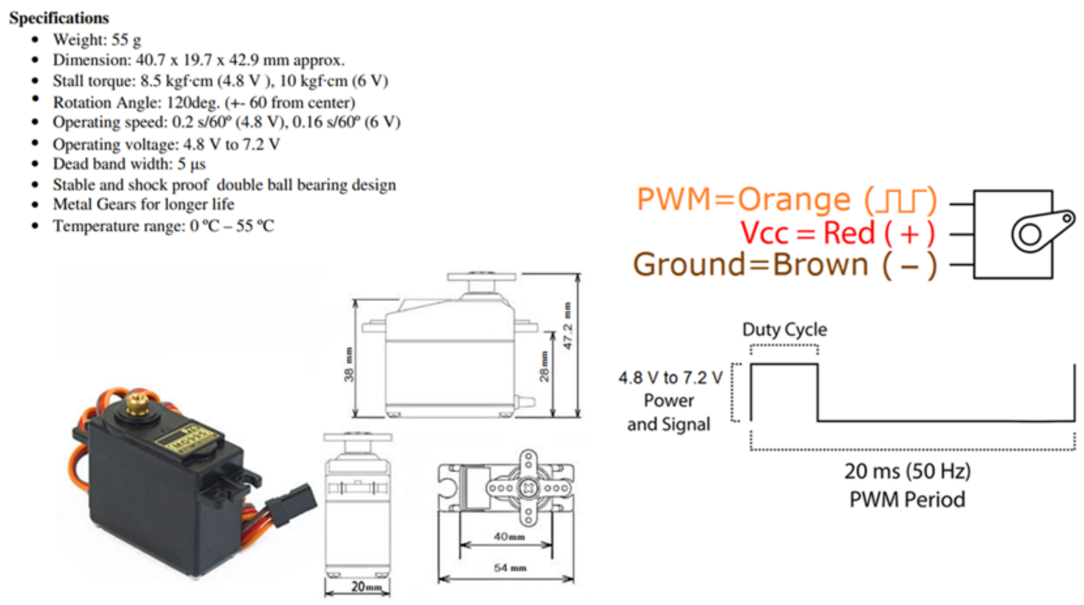
A crucial part of the simulation was analyzing the trajectory of the end-effector. To achieve this, homogeneous transformations were applied to determine the Cartesian coordinates of the end-effector at different time steps. By computing these transformations, it was possible to visualize and measure the workspace of the robotic arm, ensuring that it meets the design specifications required for its task.
This evaluation was particularly important in verifying that the robot could precisely transport objects from one conveyor to another, as required in the AmudCube system. Any limitations in reach or unexpected deviations in motion could be identified and corrected before proceeding to physical implementation.

4.4 Conclusion

Through this simulation, we have successfully **modeled the articulated arm robot** in the **Simscape Multibody environment**, ensuring that all components and joints function correctly. Additionally, by implementing **forward kinematics**, we have accurately **determined the range of motion of the end-effector**, allowing us to assess its reach and verify that it meets the required specifications for the AmudCube system.

5 Control and Actuation

5.1 Servo motor used to control the robot



Specifications
- Weight: 55 g
- Dimension: 40.7 x 19.7 x 42.9 mm approx.
- Stall torque: 8.5 kgf·cm (4.8 V), 10 kgf·cm (6 V)
- Rotation Angle: 120deg. (+- 60 from center)
- Operating speed: 0.2 s/60º (4.8 V), 0.16 s/60º (6 V)
- Operating voltage: 4.8 V to 7.2 V
- Dead band width: 5 µs
- Stable and shock proof double ball bearing design
- Metal Gears for longer life
- Temperature range: 0 ºC – 55 ºC

Figure 5: MG995 Servo Datasheet

To ensure precise and efficient movement of the robotic arm, we have chosen **MG995 High-Speed Servo Actuators** for actuation. These servo motors will be used in the joints of the robot to provide controlled rotation and positioning. The **MG995 servo** is a widely used motor in robotics applications due to its affordability, reliability, and moderate torque output. According to the documentation of the motor mentioned in Figure 5, it operates at **4.8V to 7.2V**, delivers a **stall torque of up to 10 kg.cm at 6V**, and offers a maximum rotation angle of **180°**. This makes it suitable for the articulated arm, where precise angular control is required. Additionally, the servo features metal gears, enhancing durability and longevity compared to plastic-gear alternatives.

5.2 Choice of the Programming Card

To control the servos and implement motion planning for the robot, we need a **microcontroller or microcomputer** that can generate PWM signals for precise servo actuation. Several options are considered for this task, each with its own advantages and drawbacks.

5.2.1 Arduino (e.g., Arduino Uno, Arduino Mega)

- **Advantages**
  - Easy to program and widely used in robotics projects.
  - Large community support and plenty of tutorials.
  - Low cost ($10 - $40, depending on the model).
  - Provides sufficient PWM outputs for controlling multiple servos.
- **Disadvantages**
  - Limited processing power and memory, which may be a constraint if advanced control algorithms (such as inverse kinematics) are required.
  - No built-in wireless connectivity (unless using additional modules).

5.2.2 ESP32

- **Advantages**
  - Higher processing power than Arduino.
  - Built-in Wi-Fi and Bluetooth, allowing wireless communication.
  - More PWM channels than Arduino, enabling control of multiple servos.
  - Affordable price ($5 - $15).
- **Disadvantages**
  - More complex to program than Arduino.
  - Some PWM channels might be affected by Wi-Fi operation.

5.2.3 Raspberry Pi (e.g., Raspberry Pi 4)

- **Advantages**
  - High processing power, capable of running advanced algorithms (e.g., AI-based motion planning).
  - Can handle real-time image processing and sensor integration
  - Supports Python and ROS (Robot Operating System).
- **Disadvantages**
  - Higher price ($50 - $100, depending on the model and accessories).
  - Requires an external servo driver, as Raspberry Pi does not generate PWM signals natively for servos.
  - More power consumption compared to microcontrollers.

5.3 Final Decision

Since we are still in the developing phase of our project, there are several unknowns concerning the complete set of system requirements. At this point, it is difficult to predict the exact needs in terms of processing power, communication capabilities, and control complexity. As the project progresses, we will gather more information about these requirements and make an informed decision on the most suitable control card. For now, we are keeping all options open, including Arduino, ESP32, and Raspberry Pi, and will evaluate the best choice based on the project's evolving needs.

6 Second proposal: Buy an existing robot online (SO-ARM 100)

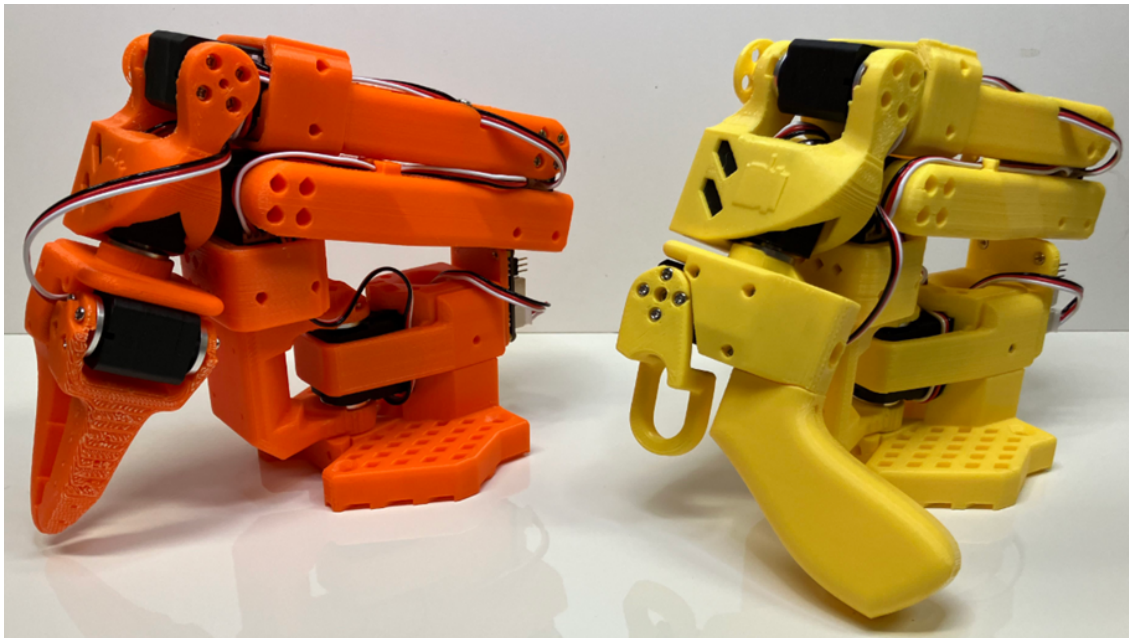


Figure 6: SO-ARM 100

6.1 Feature

- **Open-source and Affordable**: This is an open-source, cost-effective robotic arm solution developed by TheRobotStudio.
- **LeRobot Integration**: Specifically designed to integrate with LeRobot, which offers PyTorch models, datasets, and tools for reinforcement learning and imitation learning, enabling real-world robotics applications such as data collection, simulation, training, and deployment.
- **Complete Hardware Package**: The product package comes with all the essential hardware components, including motors, driver boards, power adapters, and the reComputer Nvidia Jetson edge computing unit.
- **Extensive Learning Resources**: Offers a wide range of comprehensive open-source resources, including assembly and calibration guides, along with tutorials for testing, data collection, training, and deployment, ensuring users can quickly get started and effectively develop robotic applications.

6.2 Specifications

SO-ARM100 Low-Cost AI Arm Kit	
Type	Arm Kit
Degree of freedom	6
Max Torque	19.5kg.cm@7.4V
Servo	STS3215 Bus Servo
Power Supply	$5.5mm * 2.1mm$ DC 5V4A
Angle sensor	12-bit magnetic encoder
Recommended Operating Temprature Range	0℃ to 40℃
Communication Method	UART

Table 2: Specifications of SO-ARM 100

6.3 Lists of parts needed

Part	Amount	Unit Cost
STS3215	6	\$14
Motor Control Board	1	\$11
USB-C Cable 2 pcs	1	\$7
Power Supply	1	\$10
Screwdriver Set	1	\$6
Total	—	\$118

Table 3: List of Parts

6.4 Conclusion

The second option, purchasing a robot online, is another excellent choice for our project, particularly due to its affordability. This approach allows us to bypass the lengthy process of 3D modeling and simulation, enabling us to move directly to the testing phase. Given the low cost and the availability of a ready-to-use robotic arm, this option presents a practical and efficient solution for quickly advancing our project.