

## Article

# Design and Implementation of an Automated Control System Based on a SCARA Robotic Arm Platform

Mengqi Liu <sup>1</sup>, Hanyu Xia <sup>1</sup>, Xinshuo Li <sup>1</sup>, Ying You <sup>1,2,\*</sup> and Leyi Zhou <sup>3</sup>

<sup>1</sup> Detroit Green Technology Institute, Hubei University of Technology, Wuhan 430068, China; 2411611131@hbut.edu.cn (M.L.); 2411611122@hbut.edu.cn (H.X.); 2511611119@hbut.edu.cn (X.L.)

<sup>2</sup> Hubei Provincial Key Laboratory of Modern Manufacturing Quality Engineering, Hubei University of Technology, Wuhan 430068, China

<sup>3</sup> Pittsburgh Joint Institute, Wuhan Polytechnic University, Wuhan 430048, China; zhouleyi2007@163.com

\* Correspondence: youying@hbut.edu.cn

## Abstract

At present, although there are many SCARA manipulator solutions with vertical lifting functionality, they generally suffer from high maintenance costs and complex structures. Moreover, systematic performance evaluations based on international standards are lacking, leading to unclear critical performance boundaries such as accuracy and payload in practical applications. To address these issues, this paper designs and manufactures a low-cost SCARA manipulator for educational and research demonstrations as well as light-duty electronic parts assembly scenarios. A “leadscrew + stepper motor” scheme is adopted for vertical lifting, and an Arduino Mega 2560 development board serves as the core controller, significantly reducing system cost. A three-dimensional model is established using SolidWorks 2022, and kinematic simulations are carried out with MATLAB 2024a to preliminarily verify the feasibility of the mechanism. Subsequently, a physical prototype is built and experimental tests are conducted in accordance with the ISO 9283 standard. The experimental results show that the repeatability of the manipulator is controlled within the range of 0.05–0.3 mm, the path deviation caused by vibration lies between −0.52 mm and 0.3 mm, and the maximum payload capacity is 3.91 N. These experimental data can serve as a benchmark for the design and performance comparison of similar low-cost manipulators.

**Keywords:** SCARA manipulator; Arduino Mega 2560 development board; kinematic analysis; simulation experiment; ISO 9283 standard



Academic Editor: Teen-Hang Meen

Received: 24 March 2026

Revised: 23 May 2026

Accepted: 4 June 2026

Published: 9 June 2026

**Copyright:** © 2026 by the authors.

Published by MDPI on behalf of the International Institute of Knowledge Innovation and Invention. Licensee MDPI, Basel, Switzerland. This article

is an open access article distributed under the terms and conditions of the

[Creative Commons Attribution](#)

[\(CC BY\)](#) license.

## 1. Introduction

SCARA robots have become indispensable equipment in production and daily life due to their compact structure, fast response, and high repeatability [1]. This type of manipulator is particularly suitable for high-speed planar assembly tasks [2]. Traditional SCARA manipulators are usually mounted on a fixed base, and their vertical motion is limited by the integrated Z-axis lead screw, making them ineffective in multi-layer operations requiring a large vertical stroke. Meanwhile, there is a lack of low-cost SCARA manipulators for education, exhibition, and similar fields. To address these limitations, research institutions and enterprises have explored novel structural designs and application scenarios for liftable SCARA robots, achieving substantial progress. Low-cost manipulators and liftable SCARA robotic systems now show broad prospects.

To address the high cost of robotic arms, Teixeira et al. developed SCARAMouche, a low-cost SCARA robot whose main components are 3D printed. Their work validated

the feasibility of 3D printing technology for rapid iteration of SCARA structures [3]. Alex Clark et al. designed EduSCARA [4], an open-source SCARA educational platform with an RRPR configuration, using 3D printing and widely available standard components, specifically targeting low-cost teaching scenarios. Hassan et al. [5] developed a low-cost four-degree-of-freedom SCARA robot for educational and research contexts, integrating SVM and random forest models to enhance motion control performance. Besides material cost control, algorithmic cost optimization is also a key approach to reducing costs and improving efficiency. Zihan Zhang et al. [6] built a low-cost digital twin simulation platform for SCARA robots, encapsulating control commands through modular communication, effectively reducing the cost of physical testing of control algorithms. Adar and Özden designed a vision-based low-cost SCARA system [7], integrating image processing and kinematic calculations to achieve autonomous object recognition and grasping. Using open-source hardware and software, and sharing open-source design files, can also significantly reduce R&D and manufacturing costs. Felix Herbst et al. [8] constructed the HELENE open-source 6-degree-of-freedom robotic arm platform using fully open-source low-cost hardware (stepper motors, 3D-printed structures) and the open-source ROS framework, thereby substantially lowering the entry barrier for education and light research scenarios.

Regarding the vertical lifting requirement of SCARA robots, recent research has explored three directions: mechanical structure innovation, control algorithm optimization, and adaptation to specific scenarios. Shi et al. developed a cable-driven SCARA structure that moves along a vertical gantry, concentrating actuators at the shoulder joint to reduce end-effector inertia. Their design achieved a horizontal reach of 0.5 m and vertical lifting under a 3 kg payload [9]. However, cable transmission introduces flexibility and hysteresis, causing end-effector vibrations during high-frequency start-stop cycles, thereby affecting positioning accuracy. Oriental Motor introduced the OVR series SCARA robot compatible with the EL series electric lifting column, extending the vertical stroke to 50–300 mm for compact material transfer between multi-layer conveyor lines [10,11]. Nevertheless, this modular design requires independent control of the lift joint and horizontal joints, resulting in trajectory deviation during coordinated motion, while the limited lift stroke cannot meet applications with large height variations. Shibaura Machine developed a ceiling-mounted SCARA configuration, releasing floor space through vertical installation, suitable for assembly and inspection in confined spaces [12,13]. Although this approach improves workspace utilization, the fixed mounting height cannot dynamically adapt to varying workpiece heights, and maintenance costs are high. In semiconductor manufacturing, Li et al. designed a liftable SCARA wafer-handling robot, applying structural modal analysis and optimization to improve motion accuracy and stability for reliable wafer transfer [14]. However, in practice, the mismatch between the lift range and horizontal reach limits compatibility with various wafer box specifications. The pharmaceutical and food packaging industries have adopted liftable SCARA systems for high-speed picking, conveyor tracking, and quality inspection, with their compact form and hygienic design meeting cleanroom requirements [15]. Nonetheless, existing models still require improvements in protection rating and cleanability to fully comply with stringent food-grade production standards.

Although the above solutions demonstrate potential in addressing major pain points in the current development of robotic arms, most suffer from two common issues. First, many rely on dedicated controllers or complex mechanical structures, leading to high system costs and difficulty in promotion in education, maker, or small-to-medium enterprise contexts. Second, there is a lack of systematic performance evaluation based on a unified international standard, making key indicators incomparable and non-reproducible, thus failing to serve as a benchmark reference for similar designs.

To address the above problems, This paper designs and manufactures a low-cost SCARA manipulator for educational and research demonstrations as well as light-duty electronic parts assembly scenarios. The paper adopts an Arduino Mega 2560 development board as the core controller, combines a “leadscrew + stepper motor” scheme for vertical lifting, uses predominantly 3D-printed materials, and develops a comprehensive control system for a SCARA manipulator. The main contributions of this paper are as follows:

1. **Established a low-cost robotics framework with reproducibility.**  
To address the problems of complex structures, closed-source code, and poor reproducibility in existing low-cost designs, this paper proposes an open-source SCARA robotic arm solution based on an Arduino Mega 2560 and PETG-based 3D printing. The framework is directly reproducible and provides a reusable low-cost robotics framework for educational demonstrations and light-duty industrial prototyping.
2. **Proposed a transferable methodology for general conversion.**  
To bridge the gap between kinematic models and low-cost embedded control in stepper-motor-driven robots, this paper establishes an analytical conversion from joint angles to micro-steps, along with the corresponding transmission calculations. This method is configuration-independent and transferable to other stepper-motor-based robotic arms, filling the engineering gap between theoretical kinematics and low-cost embedded implementation.
3. **Provided a standardized evaluation benchmark dataset based on ISO 9283 [16] with reproducibility.**  
To overcome the lack of unified performance evaluation standards and incomparable results in existing low-cost SCARA robots, this paper conducts systematic experimental evaluations strictly following the ISO 9283 standard. The resulting dataset is reproducible and comparable, offering a standardized evaluation benchmark for future design optimization and performance comparison of low-cost SCARA arms.
4. **Based on the above low-cost robotics framework, delivered experimentally measured performance boundaries for light-industry and educational scenarios.**  
To address the shortage of systematic experimental validation and reliable data for industrial selection, this paper builds a physical prototype and measures key performance boundaries—such as repeatability ( $<0.3$  mm) and motion stability (fluctuation  $<0.2$  mm)—in typical pick-and-place and educational demonstration tasks. These results provide direct design references and selection guidelines for subsequent research and industrial applications.

## 2. Materials and Methods

### 2.1. Design Objectives and Application Scenarios

This system takes light industrial small electronic component assembly as well as teaching and scientific research demonstration as its main application scenarios. The design goal is to minimize manufacturing and operating costs while improving system reproducibility, on the premise of satisfying the basic functions of the robotic arm.

### 2.2. Overall Design Scheme of the Robotic Arm

#### 2.2.1. Design and Modeling of Robotic Arms

In this paper, the SCARA robotic arm [17] frame is used as the basic platform, and the SCARA robotic arm contains three motion axes, which together create its unique motion characteristics. The structure provides high rigidity in the horizontal plane while maintaining compliance in the vertical direction, making it particularly suitable as a platform manipulator. The end of the SCARA robotic arm is a four-degree-of-freedom [18] manipulator, and its arm part is composed of two parts: a large arm and a small arm. The pneumatic

servo module is the main component of the end effector, consisting of three parts: a vacuum suction cup, a connecting air tube, and an air pump. The end effector of the system adopts a vacuum suction cup [19], which is connected to an air pump through a gas tube wrapped inside the robotic arm, and the air pump is connected to the Arduino Mega controller through a line to achieve integrated control. The system adopts three motors to drive the robotic arm, which can mainly realize the horizontal movement of the big arm and the small arm in the same plane and the lifting movement in the vertical plane. Stepper motor 1 drives the whole lifting platform to move vertically along the Z axis through the lead screw nut mechanism; stepper motor 2 drives the boom to rotate in the XOY plane through the synchronous belt transmission mechanism; and stepper motor 3 (located under the drive pulley shown in Figure 1) is also driven by a synchronous belt to drive the small arm to rotate in the XOY plane. The 3D modeling diagram of this system is shown in Figure 1.

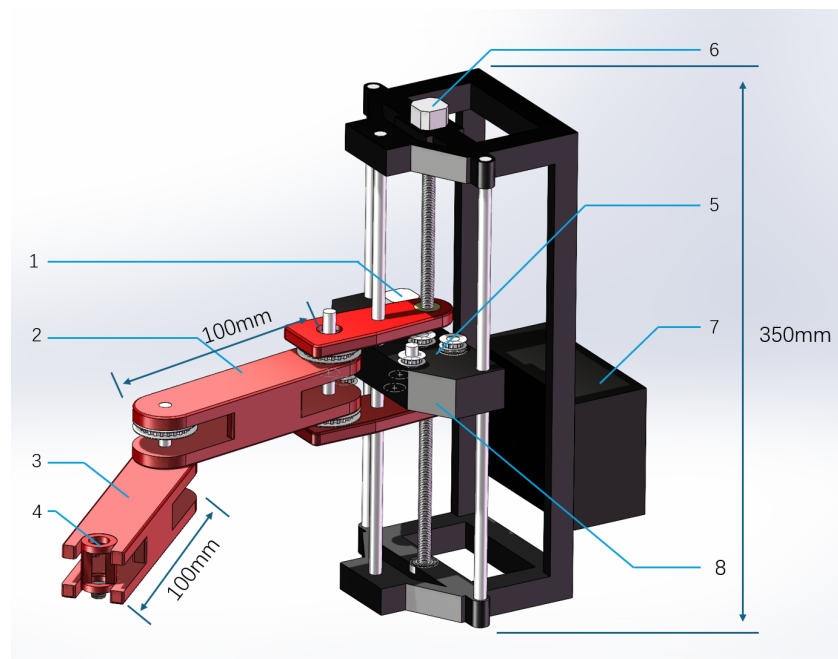


**Figure 1.** 3D modeling diagram of the SCARA robotic arm system. 1—Stepper Motor 1; 2—SCARA Upper Arm; 3—SCARA Forearm; 4—Suction Cup; 5—Lifting Platform; 6—Stepper Motor 2; 7—Development Board; 8—Stepper Motor 3.

### 2.2.2. SCARA Robotic Arm Motion Control Circuit Design

The microcontroller platform used was an Arduino Mega 2560 (manufactured by Arduino LLC, now Arduino SA, Turin, Italy). The software environment consisted of the Arduino Integrated Development Environment (IDE) version 2.3.2, which includes the AVR-GCC compiler version 7.3.0-atmel3.6.2. All version numbers are reported to ensure reproducibility of the experiments. The motion control of the system designed in this article is implemented through the Arduino Mega 2560 development board [20].

1. **Core Control Unit:** The U1 component shown in Figure 2 is an ATmega2560 microcontroller, which is the core of the embedded system. This 8-bit AVR processor provides 256KB of Flash program memory and 8KB of SRAM, forming the computing and control center of the development board.
2. **Communication Module:** U5 and X1 form the communication module, responsible for establishing a communication connection with the host computer and transmitting signals. U5 is an ATmega16U2 chip, responsible for implementing USB communication. This is the core component of the communication module. It converts the USB signals from the computer into UART serial signals recognizable by the main

chip U1, allowing users to communicate with the microcontroller via a USB cable. The communication interface is X1, which is a USB-B type interface. It is used to connect the USB cable, providing a 5 V power supply and a communication data channel. In this context, this interface is primarily responsible for communication with the host computer.

3. **Power Module:** As shown in Figure 3, IC1 and IC2 form the circuit's voltage conversion module, responsible for converting external voltage into the operating voltage for the Arduino Mega 2560. The Arduino Mega 2560 connects to an external power supply via a power cable, with the power interface being component U2 in Figure 3. The power interface is connected to the transformer IC1 through diode D1. IC1 is an NCP1117ST50T3G linear regulator that converts the external 7–12 V DC input into a stable 5 V system voltage. IC3 is an LP2985-33DBVR regulator, which can further convert the 5 V voltage to 3.3 V, providing operating voltage for components such as communication modules and resonant modules.

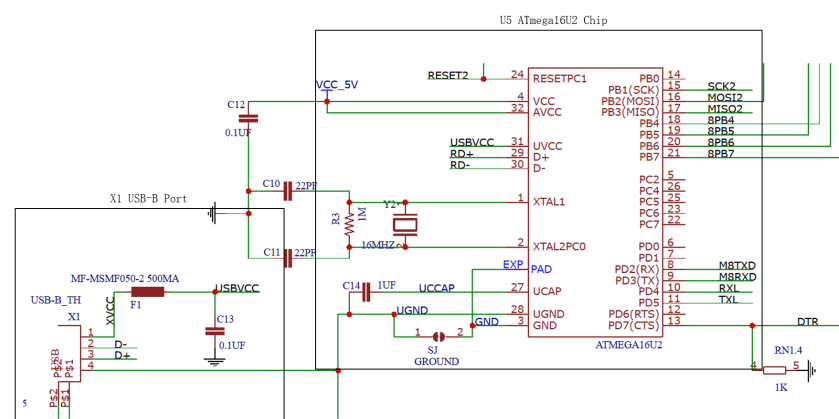


**Figure 2.** Communication module composed of U5 ATmega16U2 chip and X1 USB-B interface.

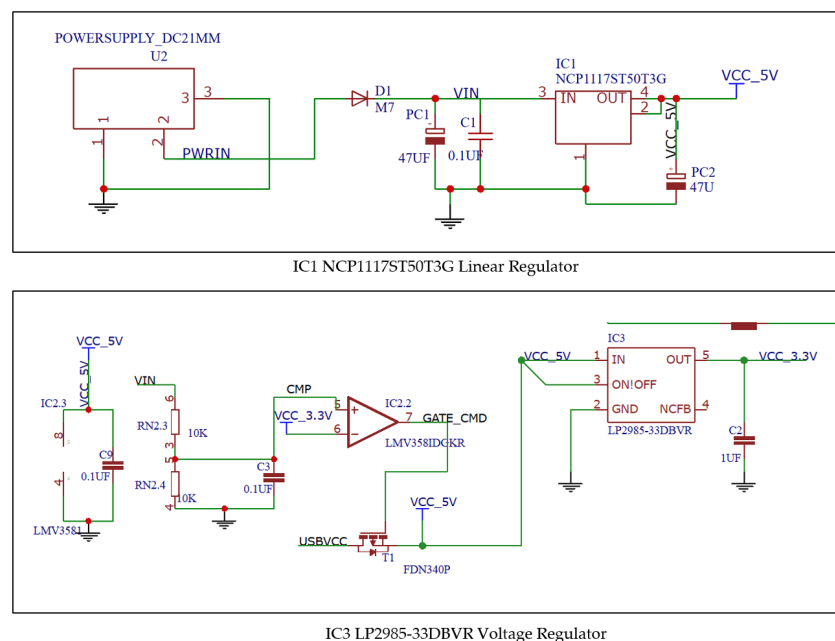


**Figure 3.** Transformer module composed of IC1 and IC3.

In addition to the above components, the development board includes several pull-up and pull-down resistors, damping resistors, inductors, capacitors, and diodes. The reset circuit ensures that the system can reliably initialize and start. The overall schematic of the Arduino Mega 2560 and the positions of the modules are shown in Figure 4.

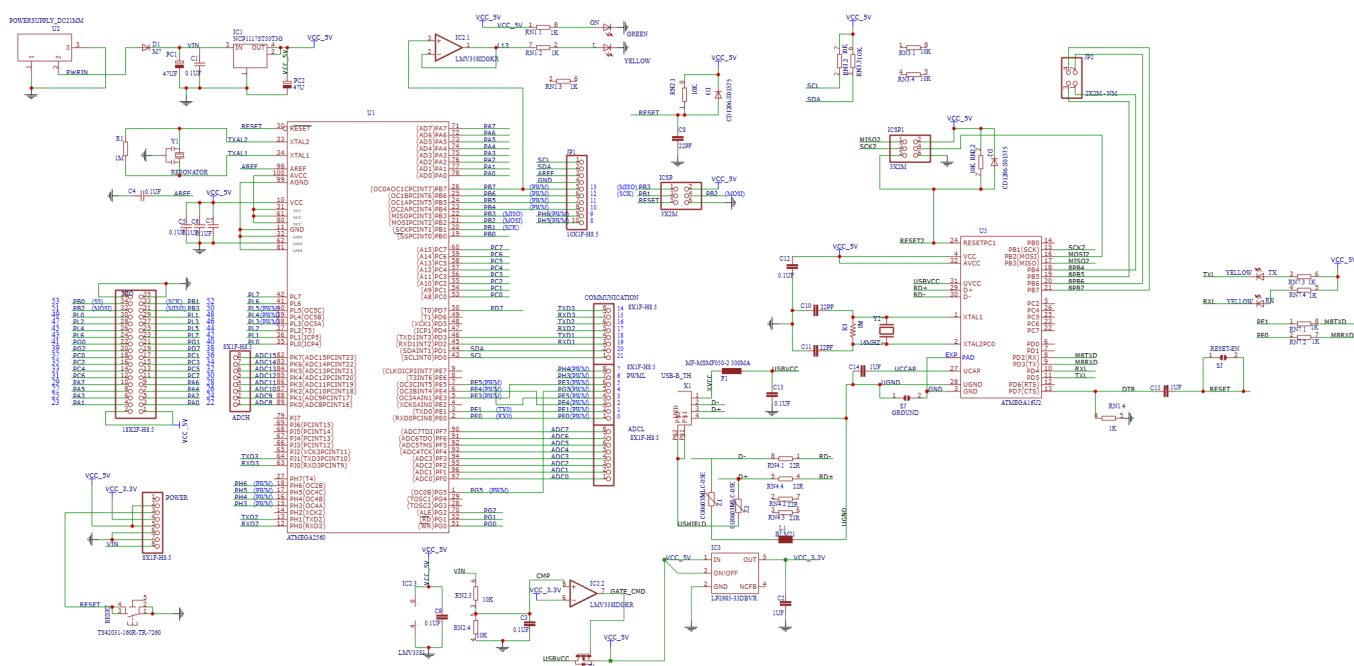


Figure 4. Arduino Mega 2560 Schematic.

This article designs two communication modes: one is Bluetooth communication mode, and the other is USB control communication. Bluetooth communication control and USB control provide a double guarantee for the system's signal transmission. The schematic of the HC05 is shown in Figure 5, where U1 is the main chip of the HC05, responsible for handling all Bluetooth communication protocols and data transmission and reception.

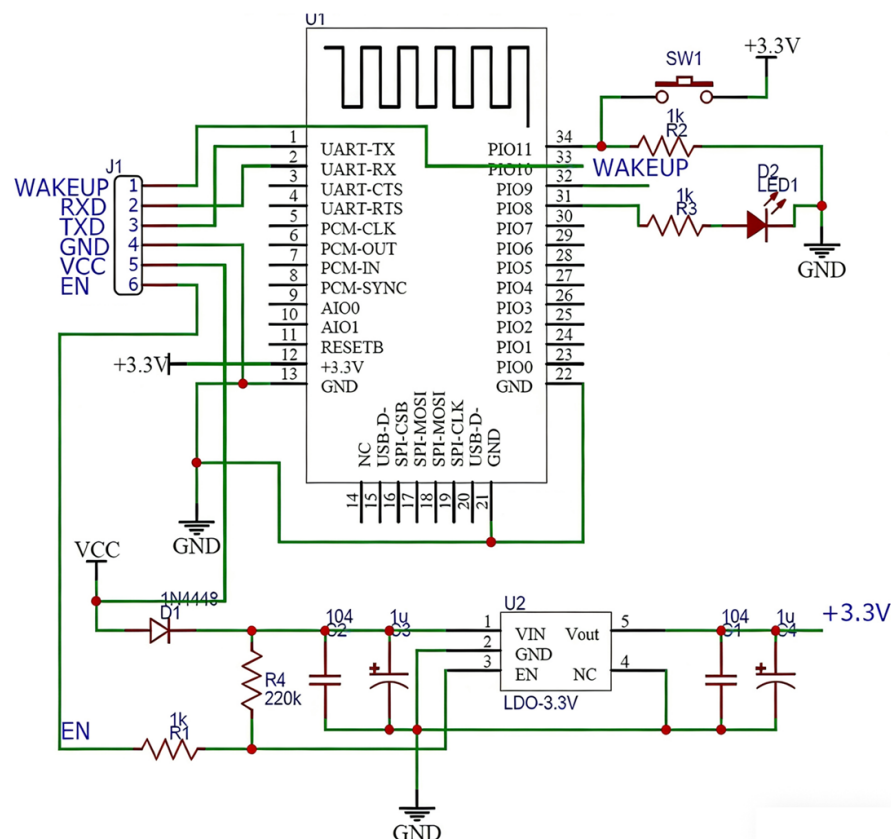


Figure 5. The complete schematic diagram of the HC05.

In addition, the HC05 also consists of diodes, resistors, and capacitors. Figure 5 shows the complete schematic diagram of the HC05.

### 2.3. Robotic Arm Kinematics Method

#### 2.3.1. Robotic Arm Space Motion Construction Method

In order to effectively control the system, the system building method is used to control the robotic arm. This article takes the end of the boom as the origin, establishes the spatial Cartesian coordinate system on this basis, and defines the point as the origin of the entire coordinate system, that is, the (0, 0, 0) point. The length of the boom and the small arm of the system is 100 mm, which corresponds to 100 unit lengths in the spatial coordinate system model, and the lifting height of the robotic arm is strictly limited to the range of 100 mm to 200 mm from the base of the system, corresponding to 100 length units in the spatial coordinate system model, that is, within the range of 100–200 on the Z axis.

#### 2.3.2. Robotic Arm Forward and Inverse Operation Algorithm

In order to effectively control the movement of the robotic arm, the mutual pushing formula of the spatial coordinates of the SCARA manipulator and the rotation angle of the manipulator is derived through theoretical derivation, that is, the inverse solution and the correct solution formula of the manipulator. In this design, the SCARA arm is defined as  $L_1$ , the forearm is defined as  $L_2$ , and the end-effector coordinates are defined as  $X_3$ ,  $Y_3$ , and  $Z_3$ . The height of the robotic arm is  $h$ , the angle between the boom and the X-axis is defined as  $\alpha$ , and the angle between the arm and the forearm is defined as  $\beta$ . By establishing a coordinate system as shown in Figure 6 and performing forward kinematic calculations [21], the robotic arm processes the coordinate instructions from the host computer and substitutes them into the solution equation to determine the rotation angle required for  $L_1$  and  $L_2$ .

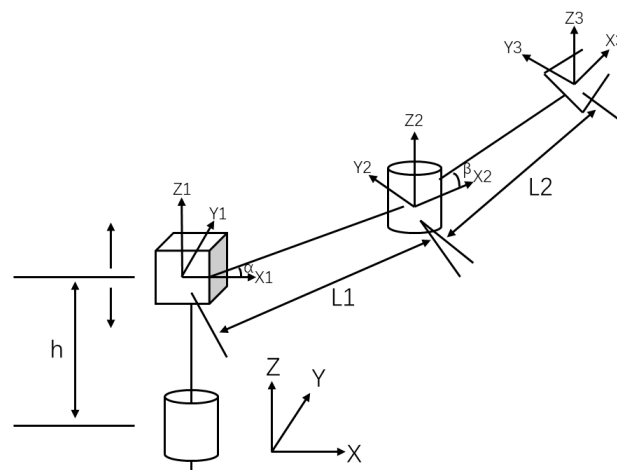


**Figure 6.** Coordinate system and kinematic parameters of the SCARA robotic arm.

When performing top-down kinematic calculations for the robotic arm, the position of the end-effector platform is defined as  $(X_3, Y_3, Z_3)$ , where

$$X_3 = L_1 \cos \alpha + L_2 \cos(\alpha + \beta) \quad (1)$$

$$Y_3 = L_1 \sin \alpha + L_2 \sin(\alpha + \beta) \quad (2)$$

$$Z_3 = h \quad (3)$$

The atan2(y, x) function can directly calculate and return the arctangent value in radians, with a range of  $[-\pi, \pi]$ . The advantage of this function is its ability to accurately

calculate the polar angle (the angle formed from the origin to the point) for points in all four quadrants, effectively avoiding calculation errors that may arise due to different quadrant positions.

Based on this, the inverse kinematic solution of the robotic arm [22] is derived as:

$$\beta = \arccos\left(\frac{x^2 + y^2 - L_1^2 - L_2^2}{2L_1L_2}\right) \quad (4)$$

$$\alpha = \text{atan2}(y, x) + \text{atan2}(L_2 \sin \beta, L_1 + L_2 \cos \beta) \quad (5)$$

$$Z_3 = h \quad (6)$$

### 2.3.3. Kinematic Theoretical Calculation Method

In order to achieve precise control of the robotic arm during the subsequent prototype construction, this article derives the relationship between the number of steps of the stepper motor and the rotation angles  $\alpha$  and  $\beta$  of the robotic arm (hereafter referred to as  $\theta$  in the following derivation) by substituting the parameters of the robotic arm into the relevant kinematic formulas.

$$R_{\text{gear}} = \frac{G_m}{G_e} \quad (7)$$

where  $R_{\text{gear}}$  is the gear ratio,  $G_m = 120$  is the number of teeth on the large gear, and  $G_e = 20$  is the number of teeth on the stepper motor gear.

The stepper motor used in this system requires 200 pulses to complete one revolution in full-step mode. To achieve more precise control, the full-step is divided into 16 equal parts, thus obtaining the micro-step number of the motor.

The number of micro-steps per motor revolution is:

$$S_{\text{motor-rev}} = S_{\text{full}} \times M \quad (8)$$

where  $S_{\text{motor-rev}}$  is the number of micro-steps per motor revolution,  $S_{\text{full}} = 200$  is the full-step number, and  $M = 16$  is the division factor.

Since the motor rotates  $R_{\text{gear}}$  times for one revolution of the output shaft, we have:

$$S_{\text{joint-rev}} = S_{\text{motor-rev}} \times R_{\text{gear}} \quad (9)$$

that is,

$$S_{\text{joint-rev}} = S_{\text{full}} \times M \times \frac{G_m}{G_e} \quad (10)$$

where  $S_{\text{joint-rev}}$  is the number of motor steps required for one revolution of the output shaft.

Then, the conversion from joint angle to motor steps is performed. The number of steps per radian for the joint is:

$$S_{\text{per-radian}} = \frac{S_{\text{joint-rev}}}{2\pi} \quad (11)$$

where  $S_{\text{per-radian}}$  is the number of steps per radian for the joint.

Therefore, for a given joint angle  $\theta$ , the target number of steps for the stepper motor is:

$$P = \frac{S_{\text{full}} \cdot M \cdot (G_m / G_e)}{2\pi} \cdot \theta \quad (12)$$

where  $P$  is the total number of micro-steps for the motor.

## 2.4. Overall Design of the Robotic Arm System

### 2.4.1. Hardware Control System Design

The core controller of this system is the Arduino Mega 2560. Stepper motor drivers serve as the driving units, while a 5 V air pump and three 42-step stepper motors act as the actuators. Three limit switches serve as the input modules. In terms of communication, the system uses two types of interfaces: one is a wireless interface through the HC-05 [23] master–slave Bluetooth module, and the other uses the built-in USB port of the Arduino Mega 2560 as a wired interface. This effectively solves the issue in automated manufacturing where lower-level devices cannot receive command signals or when command signals are delayed. A schematic diagram of the control logic is shown in Figure 7.

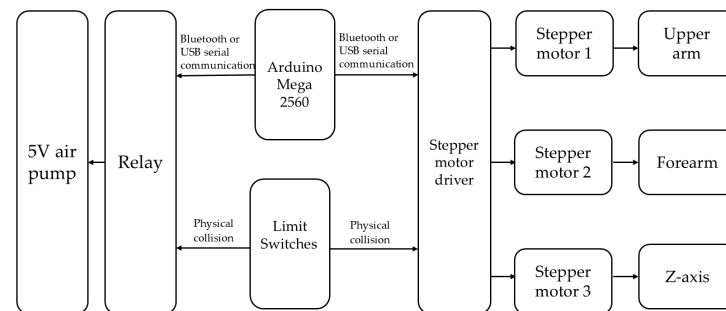


**Figure 7.** Schematic diagram of the control logic.

The system designed in this article communicates with the host computer through the Serial0 port of the Arduino Mega 2560. Its main function is to send command codes from the host computer to the Arduino Mega 2560, as shown in Table 1. The codes direct the robotic arm to move to specified spatial coordinate points, control the movement speed of the arm, and manage the operation of the air pump. Through the command codes, it is possible to set the robotic arm’s relative and absolute coordinates, query the real-time coordinates of the arm, and check the status of the limit switches.

**Table 1.** Core G/M code instruction set.

| Command    | Function                          | Use Case                     |
|------------|-----------------------------------|------------------------------|
| G0 X Y Z F | Point-to-Point Trajectory Control | Motion Command               |
| G90        | Absolute Positioning              | Coordinate Reference Setting |
| G91        | Relative Positioning              | Coordinate Reference Setting |
| M3         | Suction On                        | Grip                         |
| M5         | Suction Off                       | Release                      |
| M114       | Query Real-time Coordinates       | Monitor and Debug            |
| M119       | Query Limit Switch Status         | System Diagnosis             |

On the left side of the operation interface, there is a command sending dialog box where users input the robot arm’s control codes (G-code). By clicking the “Open Serial Port” button, the code instructions are sent to the robotic arm; simultaneously, the serial port log area on the right side will display the corresponding command sending log in real time. From this interface, more operations can be performed, such as putting the robotic arm into sleep mode, setting the sleep duration, moving the robotic arm step by step, and viewing the command log.

### 2.4.2. Software Data Processing

The control flow diagram of the system is shown in Figure 8.

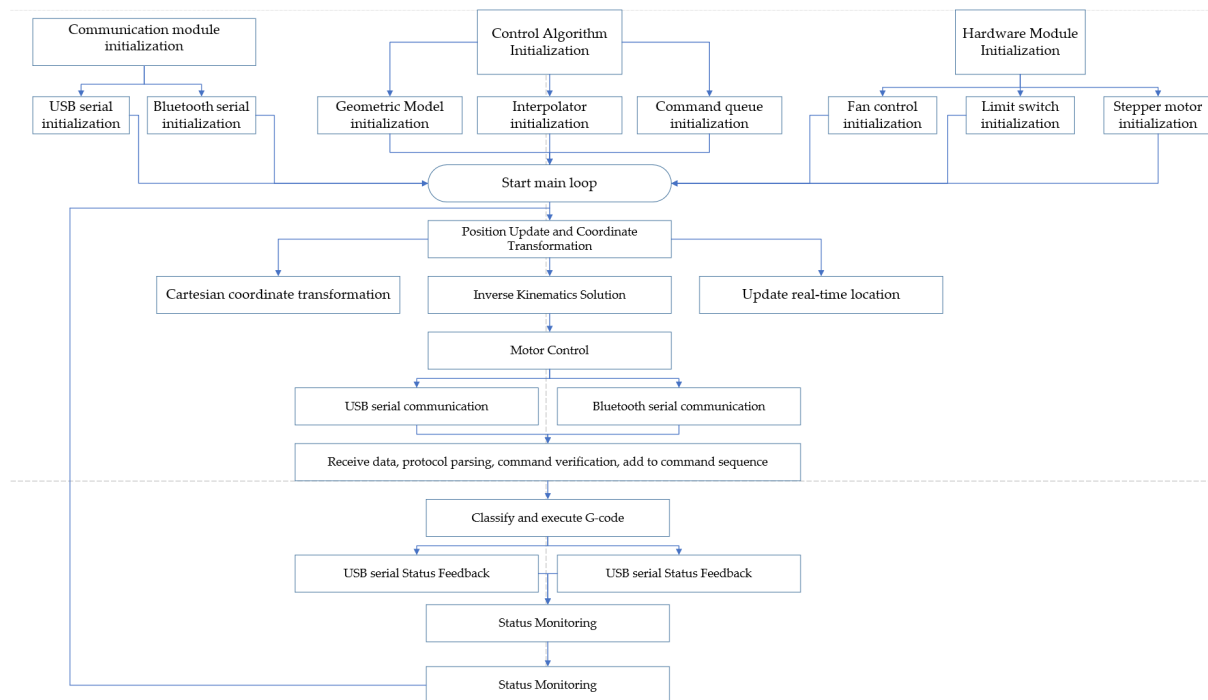


**Figure 8.** Control flow diagram of the system.

After the system is powered on, it first performs initialization, including initializing the USB communication port and Bluetooth communication port in the communication module; initializing the stepper motor, limit switches, and cooling fan in the hardware module; and initializing the geometric model, interpolator, and command queue in the control algorithm. After initialization is complete, the serial ports and hardware enter standby mode, waiting for commands from the host computer. When the host computer sends a command, the control algorithm enters the main loop, automatically controls the motor to update the current position, and automatically sends the preset initial position (200, 0, 200); the Arduino Mega 2560 will then perform inverse kinematics calculations on the initial coordinates sent by the host computer and drive the motors to reach the specified position. Then, the host computer control program can send various command codes via USB communication serial port or Bluetooth communication serial port. After receiving the commands, the Arduino Mega 2560 will first parse the protocol and verify the commands, and then place the commands at the end of the command queue. We ensured that the robotic arm can execute commands in sequence.

After a command sequence is added to the instruction, the Arduino Mega 2560 will classify the instruction codes and then execute them one by one. Once all commands have been executed, the system monitors two signal input serial ports. If no new commands are issued, the system will update the status, restart the main loop, and repeat the above steps.

This section will specifically introduce the detailed processing steps of the system for commands issued by the host computer. As shown in Figure 9, in the main loop, the program monitors commands from the USB interface and Bluetooth through the `command.handleGcode()` function. The `command.handleGcode()` function uses the `Serial.available()` function to check the data buffer, and once it detects a command ending with “\n”, it transfers the data to the next stage.

The parser is responsible for performing syntax analysis of the commands based on the instruction codes shown in Table 1 and extracting key information such as coordinates and speed. The parsed commands are encapsulated within the internal mechanism and added to the end of the command queue. This process decouples the randomness of

communication reception from the sequential execution of motion, preventing command loss and ensuring the stable execution of the command chain.

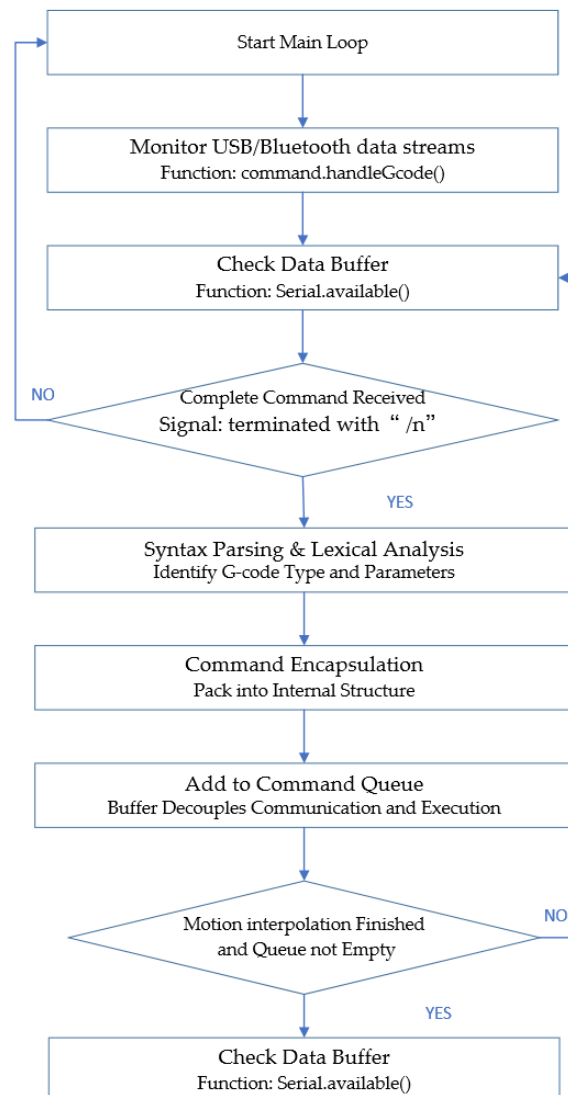


**Figure 9.** Command processing flowchart.

When the system detects that the current motion interpolation is complete (interpolator asF and the command queue is not empty, it will execute the commands sequentially starting from the first command. For query commands (such as M114, M119), the system will directly transmit them to the host computer via Bluetooth or USB. The host computer will then print the executed commands in the operation log. To address communication loss, a watchdog timer is implemented in the main loop. If no valid data packet is received within 500 ms, the system immediately stops all motion, enters a safe standby state, and sends the error code “COM\_LOST” to the host computer via the serial port or Bluetooth. For invalid commands, every received command undergoes CRC verification and validity checks, such as verifying the existence of an inverse kinematics solution and ensuring the target point lies within the safety boundaries. If a command is invalid, the system returns the error code “INVALID\_CMD” and ignores the instruction. When the system fails to reach the desired position, it reads the actual encoder or angle feedback after each interpolation cycle. If the deviation between the actual and target positions exceeds a preset threshold  $\pm 0.5$  mm, a “POS\_ERR” alarm is triggered, motion is stopped, and the host computer is notified. In the event of a collision, the system monitors joint current or torque. Once the torque

exceeds a safety threshold, a collision is detected, and the system immediately cuts off power to the actuators, returns the “COLLISION” status, and logs the relevant motion data.

## 2.5. Prototype Construction

### 2.5.1. Composition of the Prototype

The device shown in Figure 10 constitutes the experimental robotic platform prototype developed in this study. The lower-level controller uses an Arduino Mega as the mainboard, responsible for receiving commands from the host computer and outputting pulse signals to drive the stepper motors. The main mechanical structure consists of a SCARA robotic arm and a lifting platform, manufactured using PETG material through 3D printing technology. The end effector uses a silicone suction cup with a diameter of 20 mm, connected to a 5 V air pump via a silicone tube with an inner diameter of 3 mm. The entire system communicates with the host computer via a USB serial port.

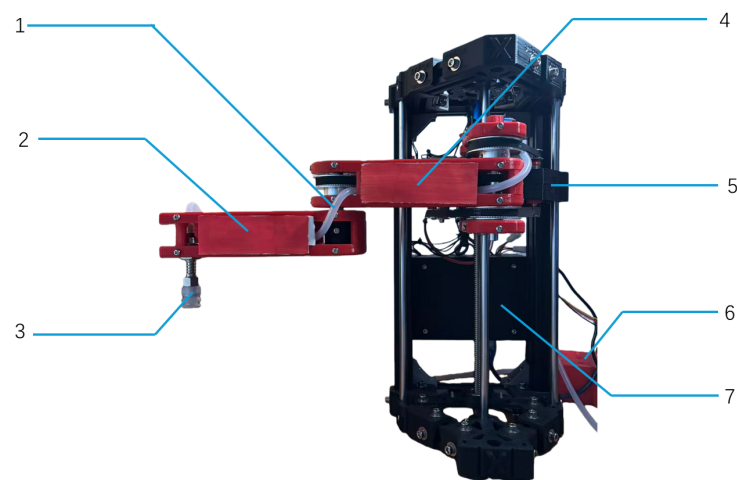


**Figure 10.** Prototype independently designed and built. 1. Air guide tube 2. Forearm 3. Suction cup 4. Upper arm 5. Lifting platform 6. Air pump 7. Arduino Mega 2560 protective case.

### 2.5.2. Circuit Wiring of the Prototype

The Arduino MEGA 2560 has 54 digital I/O pins, 15 of which support PWM output, and 16 analog input pins. These pins include: 4 hardware UART serial ports Serial0, Serial1, Serial2, and Serial3, which correspond to physical pins 0, 1, 14–19, labeled as TX and RX in Figure 11. It also has an SPI communication interface with MOSI, MISO, and SCK pins, and an I2C bus with SDA and SCL pins. Its peripherals and expansion interfaces include two ICSP programming interfaces, which support independent programming of the main controller and the USB interface chip.

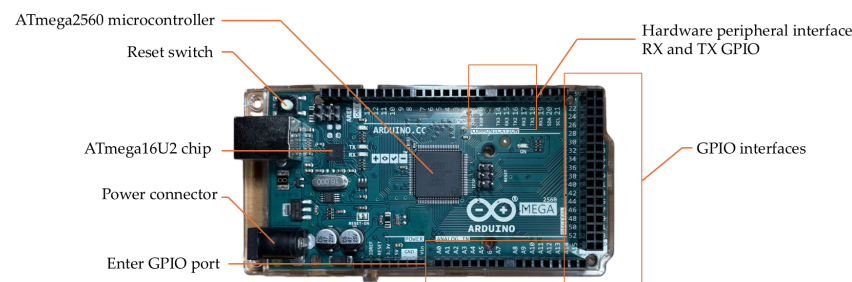


**Figure 11.** Arduino Mega 2560 physical picture.

The pin names of the HC05 Bluetooth communication module are labeled as shown in Figure 12.

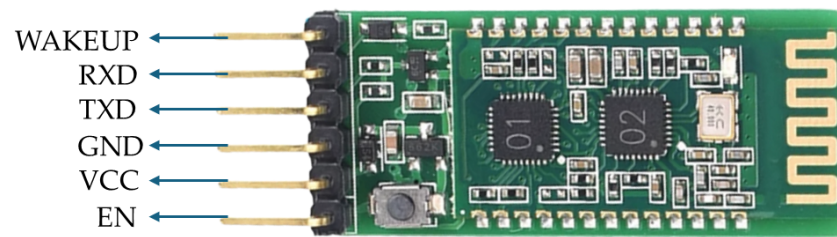


**Figure 12.** Pin configuration of the HC05 Bluetooth communication module.

In the system designed in this article, the wiring method is as follows: VCC is connected to the 5 V voltage interface of the development board, GND is connected to the GND interface of the development board, TXD is connected to the TX0 interface of the development board, and RXD is connected to the RX0 interface of the development board. The connection diagram is shown in Figure 13.

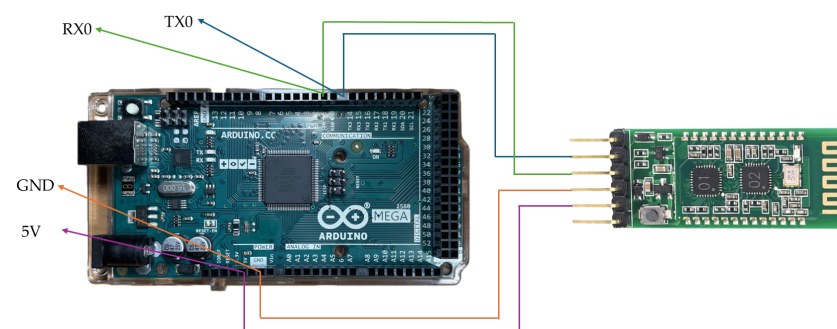


**Figure 13.** Connection diagram of HC05 Bluetooth module.

### 2.5.3. Drive System and End Effector of the Prototype

There are two types of joints in this system: rotary joints and linear joints. Specifically, the rotary joints of the large and small arms employ synchronous belt drives. The relationship between the joint output torque  $\tau_j$  and the motor torque  $\tau_m$  is as follows:

$$\tau_j = \tau_m \frac{N_{\text{main}}}{N_{\text{motor}}} \eta \quad (13)$$

where  $N_{\text{main}}$  is the number of teeth on the driving synchronous pulley,  $N_{\text{motor}}$  is the number of teeth on the driven synchronous pulley, and  $\eta = 80\%$  is the transmission efficiency.

The Z-axis lifting mechanism serves as a linear joint and employs a lead screw drive. The relationship between its end output force  $F_Z$  and the motor torque  $\tau_m$  is as follows:

$$F_Z = \frac{2\pi\eta\tau_m}{P_{\text{screw}}} \quad (14)$$

where  $P_{\text{screw}}$  is the lead of the screw, with a value of 4 mm.

Based on the above relationships, a 42 stepper motor with a holding torque of  $0.4 \text{ N} \cdot \text{m}$  was selected. According to calculations, the static maximum torque for joints 1 and 2 can reach  $1.2 \text{ N} \cdot \text{m}$ , while the theoretical output thrust of the Z-axis can reach 37.7 N.

The system uses a vacuum silicone suction cup as the end effector, with a suction cup diameter of 20 mm. Tests indicate that the 5 V air pump generates a vacuum pressure of approximately  $-50 \text{ kPa}$ . The formula for calculating the vacuum adsorption force is as follows:

$$F_{\text{Theoretical}} = P \times A \quad (15)$$

$$F_{\text{Actual}} = \lambda F_{\text{Theoretical}} \quad (16)$$

where  $F_{\text{Theoretical}}$  is the theoretical value,  $F_{\text{Actual}}$  is the actual value,  $\lambda$  is the safety factor,  $A$  is the effective adsorption area of the suction cup, and  $P$  is the vacuum pressure, i.e., the pressure difference relative to atmospheric pressure.

In practical operation, the suction cup cannot achieve 100% sealing, and its surface is not completely flat; therefore, a safety factor of 0.25 is introduced to make the experimental results more realistic. Theoretically, this system can provide approximately 3.9 N of reliable adsorption force. During motion, the dynamic torque is only 30% of the maximum torque.

### 3. Simulation and Experiment

#### 3.1. Robotic Arm Simulation Verification

To verify the performance of the system designed in this article as well as the parameters theoretically calculated above, a three-dimensional model of the system was built in SolidWorks. Subsequently, the system's dimensions, material properties, and other data were imported into MATLAB for a series of simulation experiments. After obtaining the corresponding simulation data, the results of the simulations were further validated by operating a prototype of the system.

##### 3.1.1. Workspace Simulation and Verification

The workspace is defined as the set of spatial positions that a specific reference point on the robot arm (usually the origin of the end-effector coordinate system) can reach under given constraints [24], representing the operational volume of the manipulator. In this study, the workspace is characterized through mathematical modeling in MATLAB, using the Monte Carlo method [25].

Set the lengths of both the upper arm and forearm of the robotic arm to 100 mm, with the upper arm able to rotate  $\pm 90^\circ$  and the forearm able to rotate  $\pm 150^\circ$ . Set the lifting platform to move along the z-axis within the range of 100–200 mm. The full Z-axis travel range is 0–200 mm. However, the physical structure of the lifting mechanism and the end effector occupy a portion of the lower range, and the extremely low region near 0 mm is rarely used in practical tasks. Therefore, the most commonly used and stable range of 100–200 mm was selected for testing in this paper. Select 5000 points as the sampling points for the Monte Carlo method, i.e.,  $N = 5000$ . Generate  $N$  independent joint configuration vectors  $(\alpha, \beta, z)$  through a uniform random distribution within feasible joint limits for joint space sampling. After converting the angle values to radians, use the forward kinematics equations of the SCARA robot to calculate the end-effector spatial coordinates  $(x, y, z)$  corresponding to all sampled points in parallel. Visualize the resulting end-effector positions as a 3D scatter plot, representing the geometric boundaries and spatial distribution characteristics of the workspace.

The reachable workspace model of the robotic arm can be developed through the Monte Carlo method [26], as shown in Figure 14.

In order to verify the operability of the robotic arm, this article analyzes and simulates the manipulability of each point within the reachable workspace mentioned above. Manipulability measures the ability of the robotic arm to move freely at a certain position. The greater the manipulability, the more flexible the robotic arm is at that position and the better its controllability. The formula for manipulability is as follows [27]:

$$W = \sqrt{\det(J \cdot J^T)} \quad (17)$$

where  $W$  is the manipulability measure,  $\det()$  is the determinant function,  $J$  is the Jacobian matrix, and  $J^T$  is the transpose of the Jacobian matrix.

Substituting the variables of the SCARA robotic arm, the form of the Jacobian matrix is:

$$J = \begin{bmatrix} \frac{\partial x}{\partial \alpha} & \frac{\partial x}{\partial \beta} \\ \frac{\partial y}{\partial \alpha} & \frac{\partial y}{\partial \beta} \\ \frac{\partial z}{\partial \alpha} & \frac{\partial z}{\partial \beta} \end{bmatrix} \quad (18)$$

After simplification, the manipulability measure formula for the SCARA robotic arm can be expressed as:

$$W = L_1 \cdot L_2 \cdot |\sin \beta| \quad (19)$$

where  $L_1$  is the length of the upper arm,  $L_2$  is the length of the forearm,  $\alpha$  is the rotation angle of the upper arm, and  $\beta$  is the rotation angle of the forearm.

Reachable Workspace of SCARA Manipulator (Monte Carlo Method)

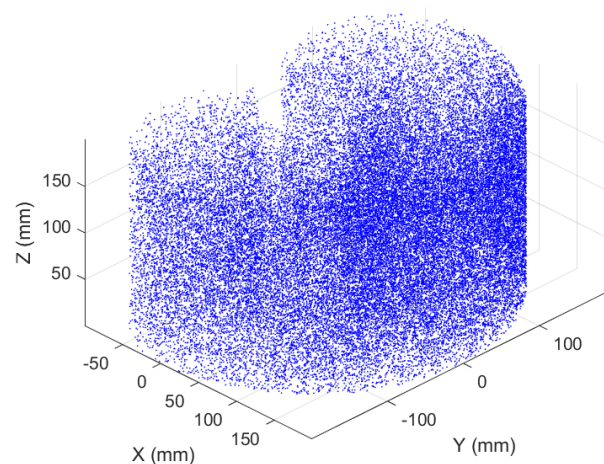


**Figure 14.** Reachable workspace model of the SCARA robotic arm.

After MATLAB simulation, the manipulability of the SCARA robotic arm is shown in Figure 15.

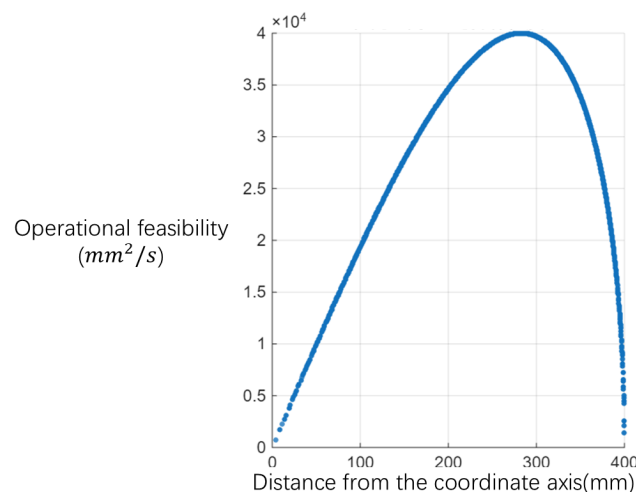


**Figure 15.** Manipulability of the SCARA robotic arm.

Therefore, it can be concluded that when the robotic arm's end effector is 282 mm from the x and y axes, the manipulability of the robotic arm is at its highest, making the arm most flexible at this position. Figure 16 shows the configurations of the robotic arm at the two points of highest manipulability, with a manipulability value of 40,000 in both cases.

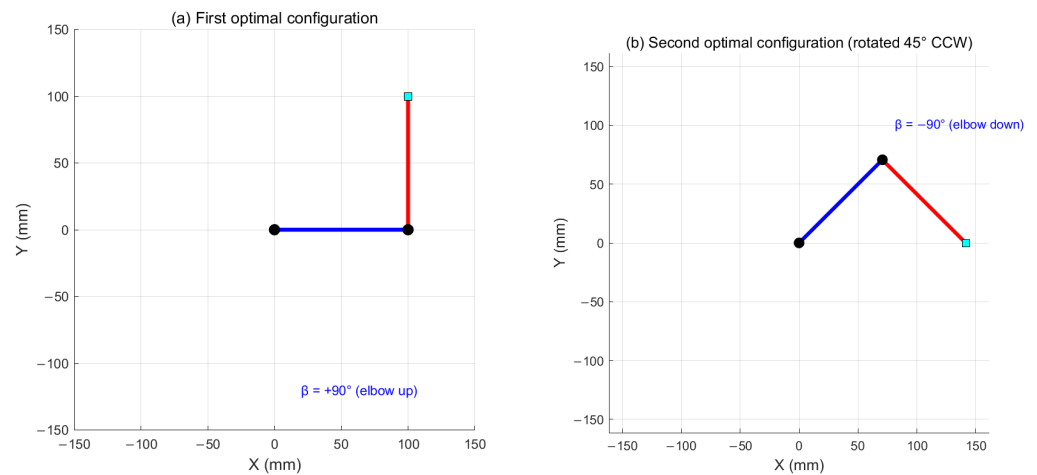


**Figure 16.** The two most operable configurations. (a) First optimal configuration. (b) Second optimal configuration.

### 3.1.2. Verification of Forward and Inverse Kinematics Formulas

To verify the accuracy and reliability of the robotic arm kinematics formulas, this article randomly selected 1000 points as sample points. First, inverse kinematics calculations were performed on these 1000 points, that is, the angles the robotic arm should rotate through were determined based on the coordinates. Then, forward kinematics calculations were conducted using these angles. The coordinates obtained from the second forward kinematics calculation were compared with the initial coordinates, and the error was calculated using the following formula. The frequency histogram is shown in Figure 17:

$$E = \sqrt{(x_i - x_0)^2 + (y_i - y_0)^2 + (z_i - z_0)^2} \quad (20)$$

where  $E$  is the Euclidean distance error,  $(x_i, y_i, z_i)$  are the initial coordinates, and  $(x_0, y_0, z_0)$  are the calculated coordinates.



**Figure 17.** Error frequency distribution histogram.

As can be seen from Figure 17, 94% of the points have errors within the range of  $0-6 \times 10^{-13}$  mm after the secondary calculation, fully demonstrating the reliability of the formulas in Section 2.2.

### 3.1.3. Maximum Load Verification Simulation

In order to verify whether the theoretical derivation results of the robotic arm's maximum load are correct, a static simulation experiment of the robotic arm was conducted in MATLAB. When simulating the dynamic maximum load capacity of the manipulator in MATLAB, the dynamic model of the manipulator is first established based on the Denavit–Hartenberg parameters, with inputs including the mass, center of mass, inertia tensor, and joint friction of each link. Then, the Monte Carlo method is used to randomly generate 2500 pose points within the reachable workspace. For each pose point, a high-speed, high-acceleration and deceleration trajectory containing maximum acceleration and maximum velocity is planned. The inverse dynamics function `inverseDynamics` is used to calculate the real-time driving torque of each joint at each time step. The load mass at the end effector is gradually increased, and iterations are performed until the peak torque of any joint exceeds the motor's allowable instantaneous peak torque, at which point the maximum dynamic load for that pose point is recorded. Finally, the `histogram` and `scatter/contourf` functions are used to plot the frequency distribution histogram and the heat distribution map. Statistical results from the 2500 points show that under motion conditions, the maximum load is 4 N, the minimum load is 3.23 N, and the average load is 3.91 N. This method allows rapid evaluation of the manipulator's dynamic load capacity without requiring a physical prototype. Within the reachable workspace of the robotic arm, the maximum force was analyzed at 2500 randomly sampled points, and the maximum load capacity at each point was recorded, resulting in the frequency distribution histogram shown in Figure 18. Furthermore, a maximum load heat distribution map of the robotic arm within its reachable workspace was created, as shown in Figure 19.

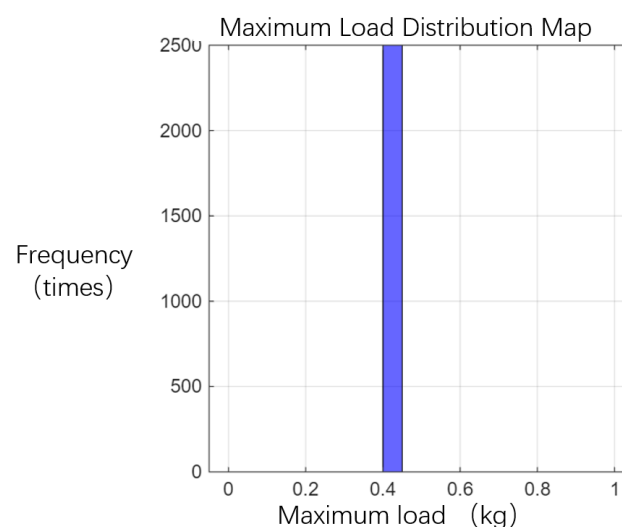


**Figure 18.** Frequency distribution histogram of maximum load capacity.

After simulation, it was found that the average maximum load of the robotic arm is 4 N, which basically matches the theoretical value of 3.9 N. During movement, based on the simulation, the motion torque is 30% of the maximum torque, that is  $0.36 \text{ N} \cdot \text{m}$ . By randomly sampling 2500 points, the frequency histogram of the robotic arm's load capacity during motion is obtained as shown in Figure 20, and the thermal distribution map is shown in Figure 21.

The above two images show the maximum load conditions of the robotic arm designed in this article under dynamic states. According to statistics, among the 2500 points selected during the movement of the robotic arm, the maximum load reaches 4 N, the minimum is 3.23 N, and the average load is 3.91 N.

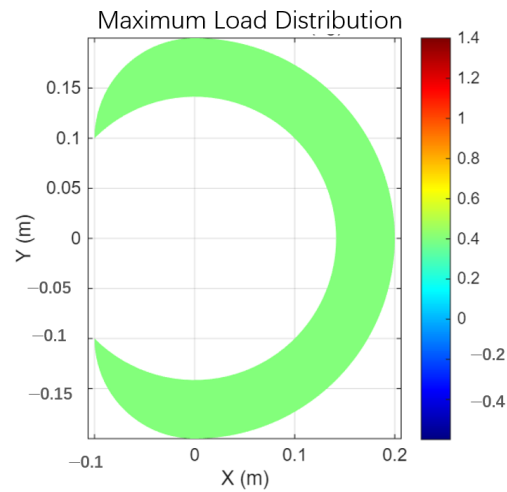


**Figure 19.** Heat distribution map of maximum load capacity within the reachable workspace.

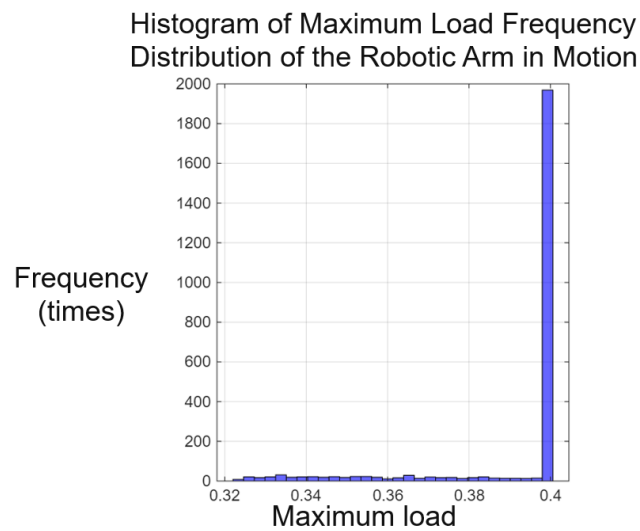


**Figure 20.** Frequency histogram of load capacity during motion.

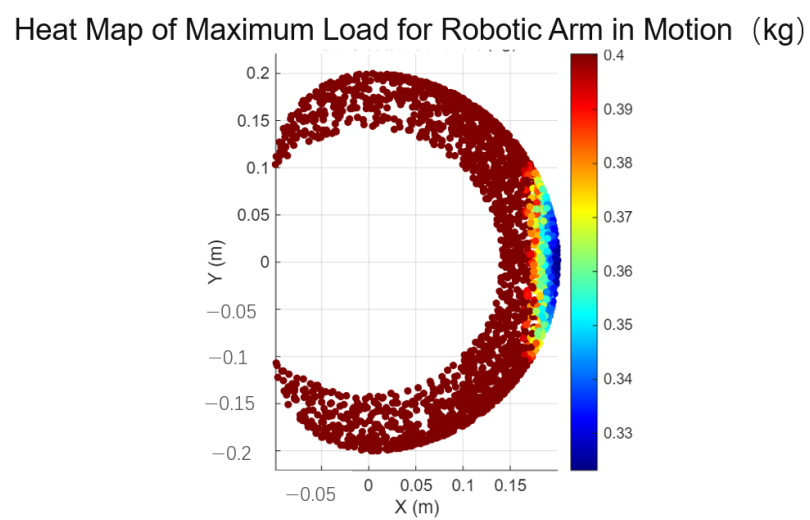


**Figure 21.** Heat distribution map of maximum load capacity during motion.

### 3.1.4. Path Error Analysis

In practical production applications, there may be precision requirements for the robotic arm's motion process, not just for reaching specified coordinate points. Therefore, this article analyzes the trajectory error during the robotic arm's movement, defining zero error as the straight line between two points. This article first conducts error analysis at speeds of 50 mm/s, 100 mm/s, 150 mm/s, and 200 mm/s, with simulation results shown in Figure 22.

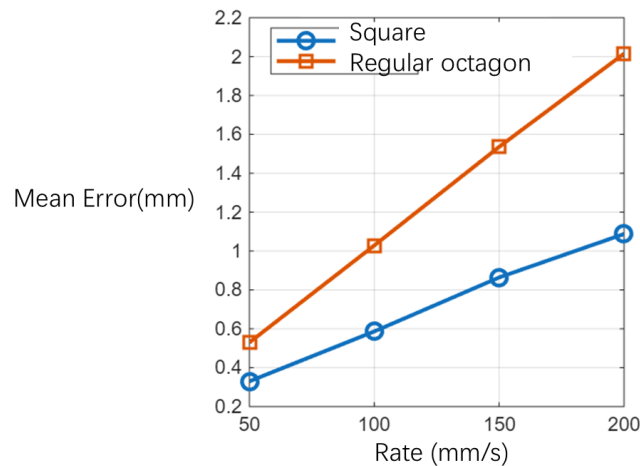


**Figure 22.** Average path error of the robotic arm at different speeds.

Then, in this article, the errors of the robotic arm relative to the standard path at each moment were recorded during square and regular octagon movements at a speed of 100 mm/s. The experimental simulation data are shown in Figure 23.

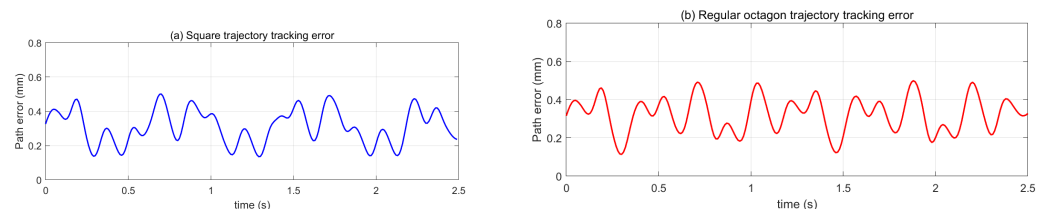


**Figure 23.** Trajectory tracking error statistical line chart: (a) Square trajectory. (b) Regular octagon trajectory.

### 3.1.5. Fatigue Analysis

The joint shaft of the SCARA robotic arm is one of the most complexly stressed and functionally critical structures in the entire system, and its fatigue performance directly affects the working life of the system [28]. In the system designed in this article, the joint shaft is made of 42CrMo high-performance alloy steel with a diameter of 12 mm. The maximum joint torque is 1.2 N · m, and during the movement of the robotic arm, the dynamic torque is 30% of the maximum torque, which is 0.36 N · m. A fatigue analysis simulation experiment of this critical joint shaft was conducted in MATLAB [29]. First, a kinematic and dynamic model of the manipulator is established based on the Denavit–Hartenberg parameters. The inverse dynamics method is used to calculate the driving torque of each joint under characteristic conditions such as maximum load. Then, according to the mechanics of materials formulas, the joint torque is converted into bending stress and torsional stress at the critical points of load-bearing components, thereby obtaining the stress time history  $\sigma(t)$ . Next, the rainflow cycle counting method is applied in MATLAB to statistically process the stress time history, extracting all stress amplitudes  $\sigma_a$  and mean stresses  $\sigma_m$ . The Goodman correction formula is then used to convert asymmetric stress cycles into

equivalent symmetric ones. Third, based on fatigue test data of the material used for the manipulator, a mathematical model of the S-N curve is constructed in MATLAB using curve fitting tools, employing the Basquin Equation  $\sigma_a = \sigma'_f(2N)^b$ . The S-N curve is plotted on a log-log coordinate system with the number of cycles  $N$  on the abscissa and the stress amplitude  $\sigma_a$  on the ordinate. Finally, the fatigue life  $N_i$  corresponding to each stress amplitude is calculated from the fitted S-N curve, and the cumulative damage is computed using Miner's linear cumulative damage rule  $D = \sum n_i / N_i$ . When  $D \geq 1$ , fatigue failure is determined, and the corresponding total number of cycles represents the end-of-life of the manipulator. The S-N curve of the SCARA robotic arm was obtained as shown in Figure 24 [30].

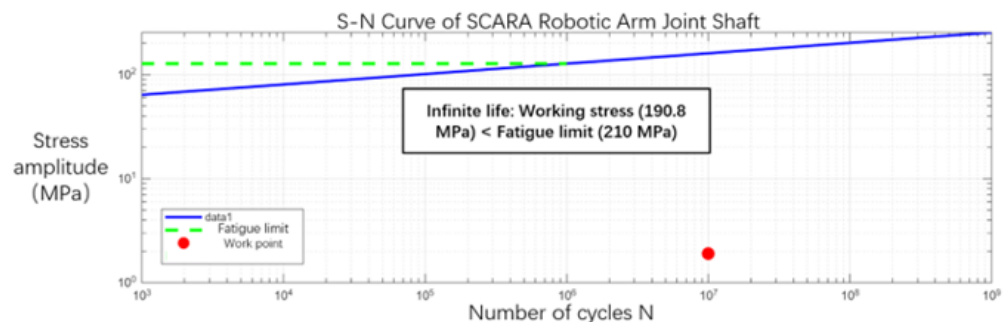


**Figure 24.** S-N curve of the SCARA robotic arm joint shaft.

### 3.2. Physical Verification of the Robotic Arm

The experiment was conducted in the laboratory. The manipulator was placed on a magnetic experimental platform, and the dial gauge was fixed to the platform via a magnetic base, lying in the same plane as the manipulator. All tests were conducted at an ambient temperature of 25 °C and a robotic arm operating speed of 20 mm/s.

#### 3.2.1. Verification of Kinematic Equations

To further verify the reliability and accuracy of the robotic arm's kinematic equations, a grasping experiment was conducted as shown in Figure 25. The coordinate commands were input into the host computer to guide the robotic arm to move to the specified position. Once in position, the air pump was activated, and the suction cup lifted the object. Subsequently, the following coordinate commands were input to transport the object to the target location. The errors between the actual placement coordinates and the commanded coordinates were recorded for each trial.

Grasping experiments were conducted according to the experimental method, with a total of 20 simulation experiments. The experimental results are summarized in Table 2.

According to the data in Table 2, out of 20 experiments, 19 had placement deviations of less than 0.2 mm, accounting for 95% of the total. In 5 experiments (25%), the deviation ranged from 0.15 mm to 0.2 mm, while in 14 experiments (70%), the deviation was between 0.1 mm and 0.15 mm. No deviations below 0.1 mm occurred.

To verify the positioning stability of the robotic manipulator designed in this paper, repeatability experiments were conducted in accordance with the ISO 9283 standard. Five test points were selected: four vertices of a cube and its center point. Their coordinates are as follows:

$$\begin{aligned}
 P_1 &: (50, 30, 100) \text{ mm}, \\
 P_2 &: (110, -30, 100) \text{ mm}, \\
 P_3 &: (80, 0, 150) \text{ mm}, \\
 P_4 &: (110, 30, 200) \text{ mm}, \\
 P_5 &: (50, -30, 200) \text{ mm}.
 \end{aligned}$$

Motion commands were sent from the host computer to the manipulator to move it to each specified target point. The position reached by the manipulator's end-effector at the first arrival was set as the reference origin. Dial gauges were installed in the X, Y, and Z directions, with their probes contacting the end-effector. All dial gauges were then zeroed. Subsequently, the manipulator was returned to its home position and commanded to move again to the same target point. The readings of each dial gauge were recorded, which represent the deviation of the second arrival position relative to the first one (the reference origin). This procedure was repeated 30 times for each target point.

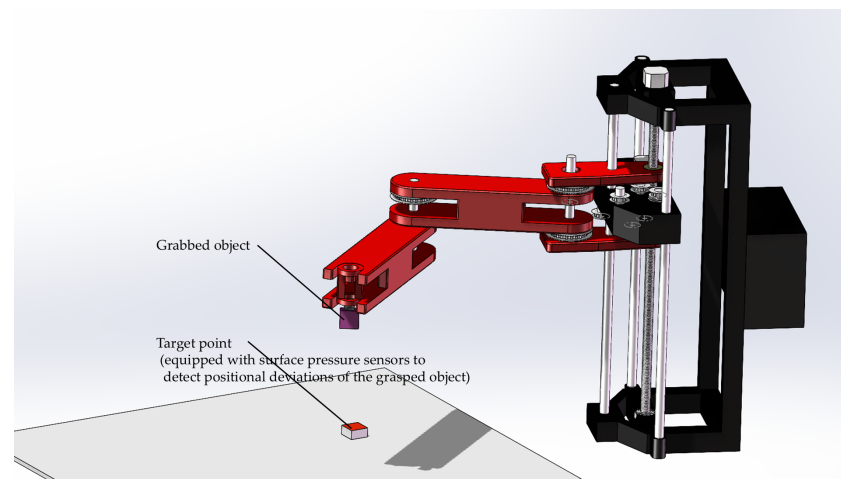


**Figure 25.** Grasping experiment setup for kinematic verification.

**Table 2.** Summary of test data.

| Error Range (mm) | Frequency | Ratio (%) |
|------------------|-----------|-----------|
| 0–0.10           | 0         | 0         |
| 0.1–0.15         | 14        | 70        |
| 0.15–0.2         | 5         | 25        |
| >0.2             | 1         | 5         |

Based on the statistical method prescribed in the ISO 9283 standard, the center of the error distribution (i.e., the mean error vector) for each coordinate axis was calculated. Furthermore, the magnitude of the three-dimensional mean deviation vector (composed of the mean deviations along each axis), as well as the magnitude of three times the standard deviation component, were computed. From these, the radius of the enveloping sphere,  $R_P$ , was obtained to evaluate the repeatability of the manipulator's positioning. According to the statistical method specified in the ISO 9283 standard, the mean error center for each dimension among the error points is calculated as:

$$l_j = \frac{1}{n} \sum_{i=1}^n j_i$$

Three times the standard deviation for each dimension is computed as:

$$3S_j = 3\sqrt{\frac{1}{n-1} \sum_{i=1}^n (j_i - l_j)^2}$$

The magnitude of the three-dimensional mean deviation vector is obtained by:

$$\bar{l} = \sqrt{l_x^2 + l_y^2 + l_z^2}$$

The magnitude of the three-times-standard-deviation component is calculated as:

$$S_3 = \sqrt{(3S_x)^2 + (3S_y)^2 + (3S_z)^2}$$

These are combined into a scalar quantity in three-dimensional space:

$$RP = \bar{l} + S_3$$

All experimental end points of the manipulator are enclosed within a sphere centered at  $\bar{l}$  with radius  $RP$ . Theoretically, 99.73% of the end points should lie inside this sphere during operation.

The position accuracy of the manipulator is computed as:

$$AP = \sqrt{(l_x - l_{x,\max})^2 + (l_y - l_{y,\max})^2 + (l_z - l_{z,\max})^2}$$

The enveloping spheres of the five coordinate points are shown in Figure 26.

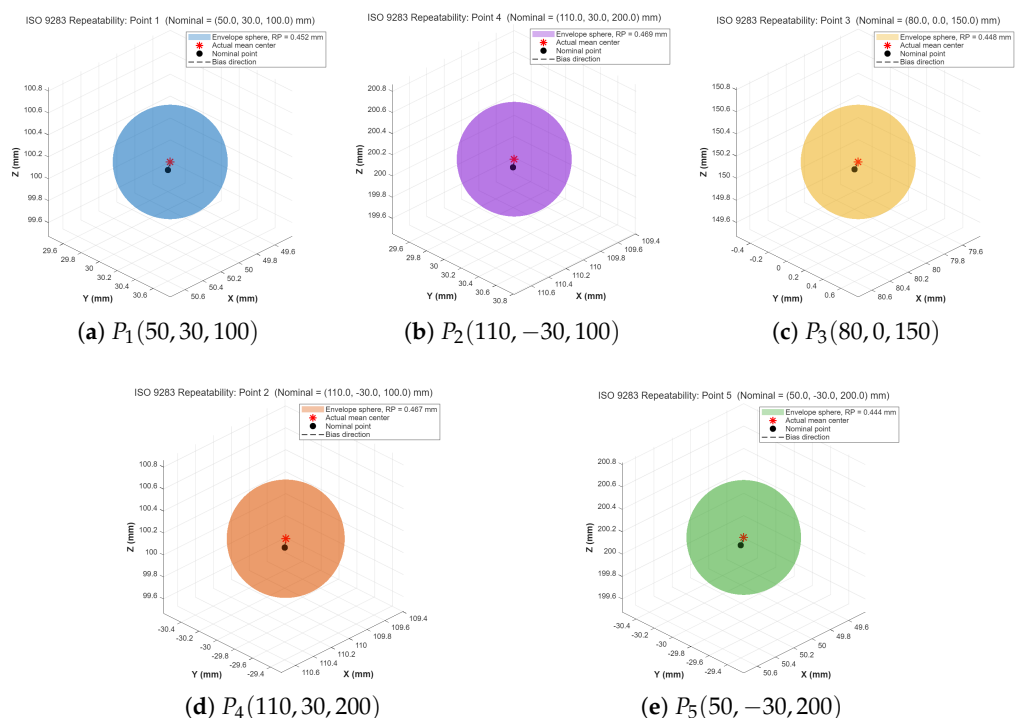


**Figure 26.** Enveloping spheres for the five test points [16].

The detailed repeatability data, including the nominal coordinates, mean deviation centers, actual sphere centers, and the RP values for each test point, are presented in Table 3.

Given the high requirements for trajectory stability during robotic arm motion in precision assembly applications, this paper further conducts a straightness experiment on the linear motion trajectory of the manipulator. In the experiment, the robotic arm moves along the straight line  $x = 100$  mm in the positive  $y$ -axis direction at a speed of 20 mm/s, with a total travel distance of 60 mm. On this straight line, measurement sections are set

every 10 mm. At each section, a dial indicator is mounted in the  $x$ -direction (perpendicular to the motion direction) to detect the deviation of the robotic arm end-effector in the  $x$ -direction when passing through that section. The readings of the dial indicators are recorded during both the forward motion (from start to end) and the return motion (from end to start). One set of data is obtained for the forward path and one set for the return path. The trajectory vibration line graphs for the forward and return paths are shown in Figure 27 and Figure 28, respectively. The experimental data statistics table is shown in Table 3.

**Table 3.** Repeatability test results for five test points.

| Point              | Mean Deviation Center | Actual Sphere Center        | RP (mm) |
|--------------------|-----------------------|-----------------------------|---------|
| P1 (50, 30, 100)   | (0.085, 0.112, 0.153) | (50.085, 30.112, 100.153)   | 0.452   |
| P2 (110, −30, 100) | (0.092, 0.104, 0.161) | (110.092, −29.896, 100.161) | 0.467   |
| P3 (80, 0, 150)    | (0.078, 0.119, 0.147) | (80.078, 0.119, 150.147)    | 0.448   |
| P4 (110, 30, 200)  | (0.095, 0.108, 0.159) | (110.095, 30.108, 200.159)  | 0.469   |
| P5 (50, −30, 200)  | (0.082, 0.114, 0.149) | (50.082, −29.886, 200.149)  | 0.444   |

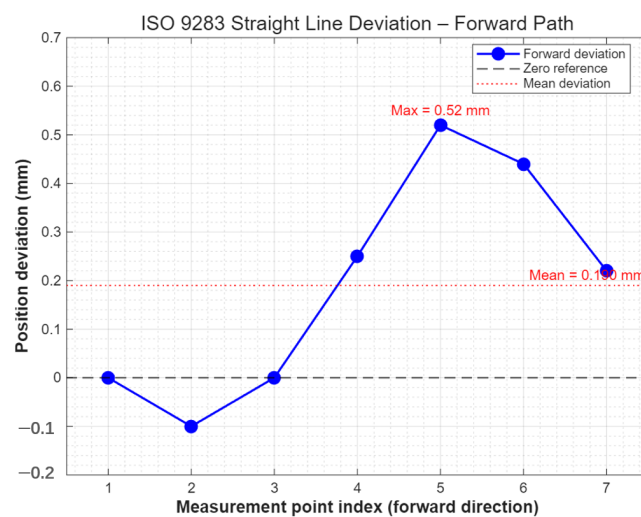


**Figure 27.** Forward path trajectory vibration [16].

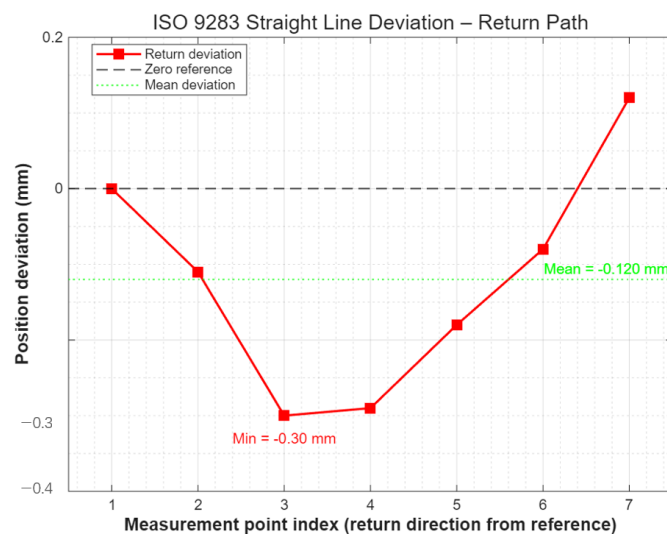


**Figure 28.** Return path trajectory vibration [16].

The forward and return path data are presented in Table 4 and Table 5, respectively.

**Table 4.** Forward path: position deviation and adjacent-point difference.

| Point No. | Position Deviation (mm) | Adjacent Difference (mm) |
|-----------|-------------------------|--------------------------|
| 1         | 0                       | –                        |
| 2         | −0.10                   | −0.10                    |
| 3         | 0                       | +0.10                    |
| 4         | 0.25                    | +0.25                    |
| 5         | 0.52                    | +0.27                    |
| 6         | 0.44                    | −0.08                    |
| 7         | 0.22                    | −0.22                    |

**Table 5.** Return path: position deviation and adjacent-point difference.

| Point No. | Position Deviation (mm) | Adjacent Difference (mm) |
|-----------|-------------------------|--------------------------|
| 1         | 0                       | –                        |
| 2         | −0.11                   | −0.11                    |
| 3         | −0.30                   | −0.19                    |
| 4         | −0.29                   | +0.01                    |
| 5         | −0.18                   | +0.11                    |
| 6         | −0.08                   | +0.10                    |
| 7         | 0.12                    | +0.20                    |

### 3.2.2. Reachable Space Verification

In order to verify whether the reachable space of the robotic arm simulated in Section 3.1.1 is accurate, a physical prototype verification was conducted in this article. Several points located within the reachable workspace calculated in Section 3.1.1 were selected, and the coordinates of these points were sent to the microcontroller via the upper computer control program. The experiments showed that the prototype could move to each of the selected coordinate positions.

To make the experimental results more convincing, the coordinate points chosen in this article were all located at the edges of the reachable workspace. The coordinate positions are shown in Figure 29:

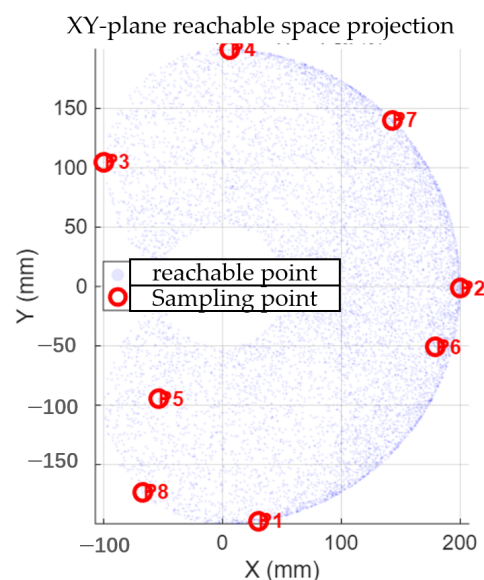


**Figure 29.** Selected coordinate points at the edges of the reachable workspace.

The Cartesian coordinates and corresponding joint angles of the selected points are shown in Table 6.

**Table 6.** Cartesian coordinates and corresponding joint angles of sampling points.

| Point | Cartesian Coordinate System (mm)    | Joint Angle (deg)                            |
|-------|-------------------------------------|----------------------------------------------|
| 1     | X = 30.05, Y = −197.73, Z = 155.70  | $\alpha = -81.4^\circ, \beta = -0.0^\circ$   |
| 2     | X = 199.97, Y = −1.55, Z = 161.75   | $\alpha = 0.4^\circ, \beta = -1.7^\circ$     |
| 3     | X = −99.82, Y = 104.74, Z = 130.44  | $\alpha = 90.0^\circ, \beta = 87.3^\circ$    |
| 4     | X = 5.99, Y = 199.91, Z = 181.82    | $\alpha = 88.3^\circ, \beta = 0.0^\circ$     |
| 5     | X = −53.91, Y = −94.66, Z = 200.00  | $\alpha = -62.7^\circ, \beta = -114.0^\circ$ |
| 6     | X = 179.18, Y = −51.21, Z = 100.01  | $\alpha = 5.3^\circ, \beta = -42.6^\circ$    |
| 7     | X = 142.60, Y = 140.23, Z = 153.22  | $\alpha = 44.4^\circ, \beta = 0.2^\circ$     |
| 8     | X = −66.83, Y = −173.41, Z = 197.54 | $\alpha = -89.4^\circ, \beta = -43.4^\circ$  |

To verify the manipulability of the robotic arm, the following trajectory tracking experiment was conducted. Different sequences of coordinates were input into the upper computer control program to drive the robotic arm along the predefined trajectory. The key coordinate points passed by the end effector of the robotic arm were recorded.

1. **Square Experiment.** By sequentially inputting the coordinates in Table 7, the motion trajectory of the end effector was recorded as shown in Figure 30.
2. **Regular Octagon Experiment.** By sequentially inputting the coordinates in Table 8, the motion trajectory of the end effector was recorded as shown in Figure 31.

In practice, due to the limitations of the robotic arm lift platform support structure, point P5 (30, 0, 150) is unreachable, but it is still retained in the diagram to illustrate the characteristics of the regular octagon trajectory experiment.

**Table 7.** Square trajectory coordinates.

| X Coordinate (mm) | Y Coordinate (mm) | Z Coordinate (mm) |
|-------------------|-------------------|-------------------|
| 110               | 30                | 150               |
| 50                | 30                | 150               |
| 50                | −30               | 150               |
| 110               | −30               | 150               |
| 110               | 30                | 150               |

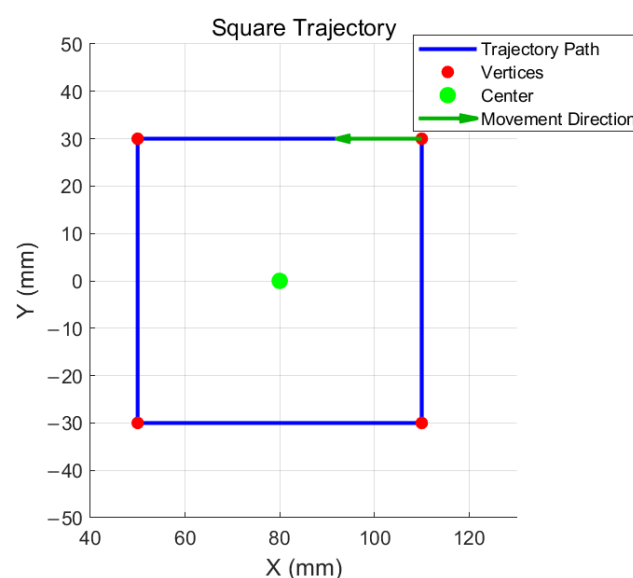
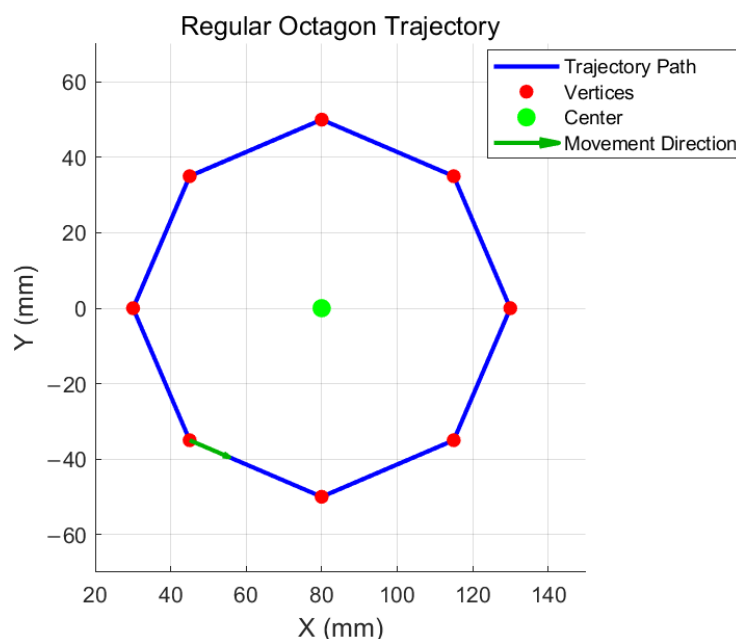


**Figure 30.** Motion trajectory of the end effector in the square experiment.

**Table 8.** Regular octagon trajectory coordinates.

| X Coordinate (mm) | Y Coordinate (mm) | Z Coordinate (mm) |
|-------------------|-------------------|-------------------|
| 45                | −35               | 150               |
| 80                | −50               | 150               |
| 115               | −35               | 150               |
| 130               | 0                 | 150               |
| 115               | 35                | 150               |
| 80                | 50                | 150               |
| 45                | 35                | 150               |
| 45                | −35               | 150               |

**Figure 31.** SCARA Robot Octagon Trajectory (XY Plane).

### 3.3. Maximum Load Verification

In order to verify the calculations and simulation results of the maximum load in Section 3.1.3, maximum load tests were conducted on the prototype using 3D-printed small blocks of different weights: 1 N, 2 N, 3 N, 4 N, and 5 N. The experimental results indicate that the prototype could reliably grasp blocks weighing 1 N, 2 N, and 3 N. Grasping a block weighing 4 N was more challenging, with some positions failing to achieve a successful grasp. Furthermore, the robotic arm was unable to grasp a block weighing 5 N under any tested condition.

## 4. Conclusions

The system is a scientific research achievement designed for light industrial scenarios (e.g., handling of small electronic components) as well as educational demonstration and teaching. The current system is not suitable for heavy-load or ultra-high-precision industrial tasks.

This study successfully designed and implemented a localized embedded control system based on Arduino Mega 2560, intended for controlling a SCARA robotic arm and pneumatic servo modules. The system supports both USB and Bluetooth communication, reducing transmission delay and overcoming the drawbacks of excessive reliance on a central server. Through a self-developed program, the host computer provides an intuitive interface for inputting commands and monitoring system status, facilitating operation and debugging.

This paper completed the design of the robotic arm structure, three-dimensional modeling, and kinematic analysis. Through analytical derivation, the maximum load capacity was calculated and subsequently validated through both simulation and experimental testing. The results indicate: the dynamic maximum load ranges from 3.23 N to 3.9 N, and the prototype can grasp objects weighing less than 0.4 kg; the positional error between the actual trajectory and the designated coordinates is on the order of 0.1–0.2 mm. The robotic arm exhibits vibration amplitudes below 2 mm at a speed of 100 mm/s, ensuring stable operation; fatigue analysis confirms that the stress on the rotating shaft remains below the material's fatigue limit.

Since our system is a self-developed low-cost SCARA prototype, there is currently no publicly available dataset of directly comparable systems. Therefore, this paper does not include quantitative comparisons. Currently, the system relies on predefined motion commands and struggles to adapt to complex and dynamic environments. Future work will integrate vision modules and artificial intelligence algorithms to enhance the system's autonomous decision-making and real-time adjustment capabilities, thereby improving its practicality and reliability in unstructured industrial settings.

**Author Contributions:** Conceptualization, M.L. and Y.Y.; Methodology, M.L. and H.X.; Software, M.L.; Investigation, M.L. and H.X.; Formal analysis, M.L. and H.X.; Validation, M.L. and H.X.; Resources, M.L. and H.X.; Project administration, M.L.; Funding acquisition, Y.Y.; Hardware design and prototyping, H.X.; Supervision, Y.Y.; Data curation, X.L.; Visualization, X.L.; Writing—original draft preparation, M.L.; Writing—review and editing, L.Z. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work was supported by the Key Project of the Scientific Research Program of the Hubei Provincial Department of Education under Grant No. D20221404 (Project title: “Research on the Mechanism of Variable-Diameter Winding Motion of a Snake-Like Robot Along Transmission Lines”).

**Data Availability Statement:** No new data was created in this study. All parameters and simulation data supporting the findings are included within the article.

**Conflicts of Interest:** The author declares no conflicts of interest.

## Abbreviations

The following abbreviations are used in this manuscript:

|       |                                                                                       |
|-------|---------------------------------------------------------------------------------------|
| SCARA | Selective Compliance Assembly Robot Arm or Selective Compliance Articulated Robot Arm |
| USB   | Universal Serial Bus                                                                  |
| UART  | Universal Asynchronous Receiver–Transmitter                                           |
| PWM   | Pulse Width Modulation                                                                |
| SPI   | Serial Peripheral Interface                                                           |
| I2C   | Inter-Integrated Circuit                                                              |
| ICSP  | In-Circuit Serial Programming                                                         |
| PETG  | Polyethylene Terephthalate Glycol                                                     |

## References

1. Arawade, S.S. State of Art Review on SCARA Robotic Arm. *Int. J. Adv. Res. Sci. Commun. Technol.* **2021**, *3*, 160–166.
2. Anonymous. *A Comprehensive Overview of Industrial Robot Configurations: Articulated, SCARA, Cartesian, Cylindrical, Spherical, and Parallel Kinematic Systems*; Technical Report; Amazon KDP: Seattle, WA, USA, 2021. Available online: <https://blackwells.co.uk/bookshop/product/Industrial-Robots-by-Cortez-Albornoz/9798748283977> (accessed on 5 May 2026).

3. Figueiredo, F.A.V.; Vasques, C.M.A. SCARAomuche: A 3D-Printed Low-Cost SCARA Robot. In Proceedings of the 2nd International Electronic Conference on Machines and Applications, Online, 18–20 June 2024.
4. Clark, A.; Martinez-Hernandez, U.; Assaf, T. EduSCARA: An open source RPRR educational SCARA platform. *HardwareX* **2026**, *26*, e00761. [CrossRef]
5. Aziz, A.G.M.A.; Al Dawsari, S.; Rafaat, A.E.; El-Magd, A.G.A.; Diab, A.A.Z. A Smart Four-DOF SCARA Robot: Design, Modeling, and Experimental Evaluation of a Low-Cost 4-DOF SCARA Robot with Machine Learning-Enhanced Pick-and-Place and Laser Engraving. *Automation* **2026**, *7*, 11. [CrossRef]
6. Zhang, Z.; Guo, Q.; Grigorev, M.A.; Kholodilin, I. Construction Method of a Digital-Twin Simulation System for SCARA Robots Based on Modular Communication. *Sensors* **2024**, *24*, 7183. [CrossRef]
7. Adar, N.G.; Özden, M. Design and Vision-Based Control of a Low-Cost SCARA Robot. *Hittite J. Sci. Eng.* **2025**, *12*, 111–120. [CrossRef]
8. Herbst, F.; Suppelt, S.; Schafer, N.; Chadda, R.; Kupnik, M. HELENE: Six-Axis Accessible Open-Source 3D-Printed Robotic Arm for Research and Education. *Hardware* **2025**, *3*, 7. [CrossRef]
9. Shi, Y.Q.; Chow, W.T.; Kwok, T.M.; Wang, Y.L. Lightweight and Low-Cost Cable-Driven SCARA Robotic Arm with 9 DOF. *Robotics* **2025**, *14*, 161. [CrossRef]
10. Oriental Motor. SCARA Picking Robot with Lifting Function Improves Productivity and Space Saving. Available online: <https://www.orientalmotor.com.my/my/solutions/material-handling/05> (accessed on 5 May 2026).
11. SEISANZAI Japan. Oriental Motor expands motion range of compact OVR robot series. *SEISANZAI Japan*, 20 June 2025.
12. Shibaura Machine. *Ceiling-Mounted SCARA Robots Launched by Shibaura Machine*; Connectivity 4IR: Kent, UK, 2025.
13. Emerald Publishing. Ceiling mounted SCARA robot set to save space and improve access to working areas. *Ind. Robot* **2007**, *34*, 5.
14. Li, J.J.; Zhou, H.B.; Xu, Y.L. Structural Design and Vibration Reduction Optimization of SCARA Wafer Transfer Manipulator. *J. Dyn. Control* **2024**, *22*, 12.
15. ENCY CAD/CAM. What Is a SCARA Robot? Structural Features, Operational Capabilities, and Industrial Relevance. Available online: <https://encycam.com/articles/what-is-a-scara-robot-structural-features-operational-capabilities-and-industrial-relevance/> (accessed on 5 May 2026).
16. ISO 9283:1998; *Manipulating Industrial Robots — Performance Criteria and Related Test Methods*. International Organization for Standardization: Geneva, Switzerland, 1998. Available online: <https://www.iso.org/standard/22244.html> (accessed on 5 May 2026).
17. Wang, Y.; Chen, Y. Kinematic Analysis of SCARA Robot Based on White Spirit Box Packaging Production Line. *Mech. Eng. Autom.* **2022**, *2*, 30–32.
18. Zhou, F.; Chen, F.L.; Shen, J.L.; Yang, X. Study of Kinematics Simulation of 4-DOF Manipulator Based on MATLAB. *Mach. Build. Autom.* **2016**, *45*, 115–119.
19. Liu, Z.Q.; Zhou, L. Structural Optimization of Quick-Change End-Effector for Industrial Robots in Multi-station Flexible Production. *Mod. Manuf. Technol. Equip.* **2025**, *9*, 94–96.
20. Qiang, X.M.; Shan, Y.; Hao, H. Smart Home System Based on Arduino Mega 2560. In Proceedings of the Electrical Engineering Conference, Krabi, Thailand, 7–9 March 2018; pp. 114–115. Available online: <https://www.cnki.com.cn/Article/CJFDTotat-DGJS201820028.htm> (accessed on 5 May 2026).
21. Xiong, D.; Tu, Y.; Qiu, W.Y. Design of an Automatic Loading and Unloading System for CNC Turning Based on AGV Chassis and SCARA Robotic Arm. *J. Hunan Univ. Arts Sci. (Nat. Sci. Ed.)* **2025**, *37*, 34–39.
22. Yang, Y.L.; Li, C.Y. Design and Experiment of SCARA-based Flexible Grape Picking Robot Arm. *J. Agric. Mech. Res.* **2025**, *47*, 72–78.
23. Guan, Z.Y. The Design and Implementation of a Small Car Following Driving System Based on HC-05. *Shanxi Electron. Technol.* **2025**, *1*, 7–10, 44.
24. Li, H.Y.; Dong, S.L.; Ye, D.P. Kinematics Analysis of Five-DOF Manipulator Based on Matlab. *J. Ningde Norm. Univ. (Nat. Sci. Ed.)* **2025**, *37*, 264–271.
25. Gao, B.; Zhang, J.; Deng, X.; Wu, L.; Wang, X.; Zheng, Z. Research on PMSG Low Voltage Ride Through Performance Optimization Based on Combined Weighting-Monte Carlo Method. *Acta Energaie Solaris Sin.* **2025**, *46*, 439–449.
26. Li, B.L.; Chen, Z.F. Kinematic and Workspace Analysis of a Five-Degree-of-Freedom Parallel Packaging Robot. *Mod. Mach.* **2024**, *6*, 22–26.
27. Li, Z.K.; Fu, M.L.; Jiang, G.D. Improved Dynamic Movement Primitives Method Based on Redundant Manipulator Manipulability. *Comput. Integr. Manuf. Syst.* **2025**, *31*, 36–48.
28. Qin, G.; Chen, D.; Wu, J.; Li, Y.; Liu, H. Study on Conversion Factor of Bending Fatigue Limit for Gears from 50% to 99% Reliability. *Trans. Mater. Heat Treat.* **2025**, *46*, 210–222.

29. Xia, G.; Zhou, B. Research on Fatigue Life Evaluation Method of Escalator Truss Based on S-N Curve. *Build. Sci. Technol.* **2024**, *6*, 47–52.
30. Lyu, W. Fatigue Mechanics Analysis of the Shaft Head of High-speed Shear Single-shaft Shredder. *Mech. Eng.* **2025**, *8*, 119–122.

**Disclaimer/Publisher’s Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.