ECE 445

SENIOR DESIGN LABORATORY

FINAL REPORT

H.E.R.O. - Hazardous Environment Remote Operator

<u>Team #9</u>

XIHE SHAO (xihes2@illinois.edu) JUN LIANG (junl6@illinois.edu) QIHAN SHAN (qshan3@illinois.edu) SIZHAO MA (sizhaom2@illinois.edu)

TA: Leixin Chang

April 17, 2025

Abstract

This project presents the development of H.E.R.O. (Hazardous Environment Remote Operator), a vision-based robotic hand system designed to replicate human gestures in real time for remote operation in hazardous environments. The system eliminates reliance on wearable sensors by employing stereo camera tracking and depth sensing to capture hand movements, which are translated into motor commands via a PID-controlled servo system. A modular, 3D-printed robotic hand—fabricated using materials such as PLA and TPU—demonstrates adaptability to diverse tasks and environmental conditions. The prototype achieves sub-200 ms latency.

While the system successfully mimics basic grasping motions, limitations include the rigid PLA hand's limited grip strength, single-degree-of-freedom joints restricting dexterity, and gesture tracking instability under variable lighting or occlusion. Iterative design improvements focused on thumb articulation, connector stiffness optimization, and spatial filtering algorithms to enhance mechanical stability and tracking robustness.

Ethical considerations emphasize transparency in reporting system limitations and prioritizing safety during testing, aligning with the IEEE Code of Ethics. The project's broader impact lies in its low-cost, open-source framework, which supports scalable adoption in industrial automation and disaster response while promoting sustainable practices through localized 3D printing. Future work includes integrating soft robotics for improved dexterity, refining vision algorithms for robust tracking, and expanding degrees of freedom for complex manipulation.

Contents

1	Intro	Introduction					
	1.1	Purpose	1				
	1.2	Fuctionality	1				
	1.3	Changes from previous version	1				
	1.4	System Overview	1				
2	Desi	ign	3				
	2.1	Design Procedure	3				
		2.1.1 Motor Control System	3				
		2.1.2 Visual Perception System	4				
		2.1.3 Robotic Hand Mechanical Design	5				
	2.2	Design Details	8				
		2.2.1 Motor Control System	8				
		2.2.2 3D position to angle mapping and filtering	11				
		2.2.3 Visual Perception System	14				
		2.2.4 Robotic Hand Mechanical Design	17				
3	Veri	fication	19				
5	3.1	PWM Signal Validation	19				
	3.2	UART Communication Test	19				
	3.3	Visual Perception Spatial Accuracy Verification	20				
	3.4	Real-Time Performance Evaluation	21				
	3.5	Grasping Verification					
4	Cost	ts	23				
-	-						
5	Con	clusion	24				
	5.1	Executive Summary	24				
	5.2	Accomplishments	24				
	5.3	Limitations and Improvements	24				
	5.4	Ethical Considerations	24				
	5.5	Broader Impacts	25				
Re	eferer	nces	26				

1 Introduction

1.1 Purpose

Human workers in hazardous environments, such as those involving toxic materials or high-pressure equipment, face significant safety risks. Current solutions, including sensor-equipped gloves and pre-programmed robots, suffer from inflexibility, discomfort, high costs, and dependence on wearable devices, limiting their effectiveness. To address these challenges, this project introduces a vision-based robotic hand system that leverages camera tracking and 3D-printed components to replicate human gestures in real time. By eliminating reliance on physical sensors, this solution enhances adaptability and accessibility in operations.

1.2 Fuctionality

The system operates through three key functions:

- **Gesture Capture & Interpretation:** Cameras track human hand movements without requiring wearable sensors, ensuring non-invasive operation.
- **Precision Motion Translation:** A closed-loop feedback system converts gestures into robotic movements with sub-millisecond accuracy.
- **Modular Robotic Manipulation:** A 3D-printed robotic hand interacts with hazardous objects, and can easily be modified to adapt diverse tasks. We have made few types of hand and fingers for different object. It can also adapt different environment by changing printing materials (PLA, ABS, TPU, PETG–CF).

These functionalities collectively enable real-time, reliable human-robot interaction in dangerous settings.

1.3 Changes from previous version

We modify the thumb angel for better fraction. And tested different hardness of connector to balance between stability and bounce.

We also tried to use depth camera for a stable hand capture, and add position constrain in both high level code and micro controller code for more safety.

1.4 System Overview

The system consists of three interconnected subsystems:

- **Vision Module:** Stereo cameras capture hand gestures, with algorithms processing spatial data for motion tracking.
- **Control Unit:** Translates gesture data into motor commands using PID feedback to minimize latency (<200 ms).

• **Robotic Hand Assembly:** Modular 3D-printed joints and actuators replicate human hand dexterity.

The Figure 1 shows system structure of this design, and we will discuss the detail later.



Figure 1: Block Diagram

2 Design

2.1 Design Procedure

2.1.1 Motor Control System

The control system shown in Figure 2 design prioritized precision, real-time responsiveness, and safety through hardware-software co-design. Three key factors shaped the architecture:



Figure 2: Controller subsystem interfaces

PWM Generation Strategy: Equation 1 and 2 linear mapping was selected over lookup tables for computational efficiency while maintaining smooth interpolation across the 0° -180° range. Hardware timer peripherals (TIM2/TIM8) were chosen instead of software PWM due to their deterministic timing accuracy, critical for maintaining 50Hz servo signals. Alternative clock sources (internal RC oscillator) were rejected due to the ±2% frequency drift.

Communication Protocol Selection: UART at 115200 baud outperformed alternatives like SPI or I²C in simplicity for PC integration. The 5-byte payload structure with direct angle mapping minimized parsing overhead in the interrupt handler, crucial for less than 10 ms data response in PC and micro controller. A simply data flow is shown in Figure 3

Safety Mechanisms: Dual-layer protection (software angle clamping and hardware failsafe) was implemented after hazard analysis revealed single-point failure risks in PCB



Figure 3: UART data processing flowchart

traces. The mid-position (90°) failsafe balances mechanical neutrality better than zero-force solutions while avoiding abrupt movements from sudden power loss.

2.1.2 Visual Perception System

The visual perception system (shown in figure 4 is designed to accurately track hand movements in 3D space using a depth camera and computer vision algorithms. It captures synchronized RGB and depth data, detects hand landmarks in 2D using a lightweight neural network, and converts them into real-world 3D coordinates. This pipeline enables robust, low-latency hand tracking for applications like robotic control and human-computer interaction.

Sensor Selection: The system uses an Intel RealSense D405 depth camera [1] for highprecision 3D sensing. Compared to standard RGB cameras, the D405 provides active depth sensing using infrared patterns, ensuring reliable depth data even in low-light conditions. Its synchronized RGB-D output eliminates alignment issues between color and depth frames, while its compact size and USB 3.0 interface make it ideal for embedded systems. The depth accuracy (±2% within 0.3–3 meters) is critical for precise 3D hand tracking.

Visual Perception Algorithm: MediaPipe's hand landmark model [2] offers a real-time, lightweight solution for 2D hand tracking. Unlike heavier CNN-based models, MediaPipe achieves 20+ FPS on consumer CPUs by optimizing for edge devices. It detects 21 hand landmarks with pixel-level accuracy and outputs normalized coordinates. The model's low computational cost allows seamless integration with depth processing, and its single-hand detection mode further reduces latency for our use case.

Pixel-to-3D Algorithm: The system converts MediaPipe's 2D landmarks into 3D world coordinates using depth data. Each landmark's (u,v) position in the RGB image is mapped to its corresponding depth value, then transformed via the camera's intrinsic parameters. This step includes depth validation to handle occlusions and noise, ensuring only physically plausible 3D points are generated. The final output is a vector of 3D hand landmarks, ready for robotic control or gesture recognition.



Figure 4: Visual Perception System

2.1.3 Robotic Hand Mechanical Design

Material choice The design of the robotic hand prioritized both reliable return-to-origin capability and long-term durability of the finger joints. To achieve stable rebound after actuation, the connection components of the finger joints—specifically parts 7 and 8 in Figure 5—were fabricated using TPU (Thermoplastic Polyurethane) material. TPU was selected for its combination of flexibility, durability, and its ability to retain consistent elastic properties even after repeated loading cycles, making it an ideal choice for ensuring the robotic fingers return smoothly to their initial, neutral positions without the need for additional mechanical springs or rebound mechanisms.

Parts 7 and 8 serve as flexible connectors between adjacent finger segments, enabling smooth joint articulation while maintaining sufficient elasticity to restore the fingers to their starting position following each actuation. This design not only simplifies the mechanical structure but also reduces assembly complexity and potential points of failure.

The mechanical performance of the robotic hand is strongly influenced by the hardness of the TPU material used for these connectors. Excessive hardness could impair finger flexibility, while insufficient hardness would compromise structural stability and rebound force. To quantitatively evaluate material performance, Table 1 summarizes the key mechanical properties of TPU 85A, 90A, and 95A at 25°C, including Young's modulus, 100% modulus, and tensile strength.

Material	Young's Modulus (MPa)	100% Modulus (MPa)	Tensile Strength (MPa)
TPU 85A	8–12	3.5–5	30–40
TPU 90A	12–20	7–9	35–45
TPU 95A	20–27	12–15	40–50

Table 1: Mechanical Properties of TPU at Different Shore Hardness Levels (25°C)

As shown in the table, increasing Shore hardness results in higher Young's modulus, 100% modulus, and tensile strength, indicating greater rigidity and load-bearing capacity. However, this comes at the expense of flexibility, which is critical for the smooth articulation and elastic rebound of the finger joints. Based on both experimental evaluation and the mechanical data, TPU 85A was selected for the finger joint connections, as it provides the optimal balance between flexibility for reliable rebound and adequate strength for long-term durability. This choice ensures that the robotic hand achieves both robust performance and a long operational lifespan.

Dimensional Selection The dimensions of our robotic hand were designed to closely match those of a human hand. By mirroring both the proportions and overall appearance of a real hand, the mechanical hand achieves a high degree of biomimicry. This not only enhances functional compatibility but also supports the goal of creating a lifelike and natural-looking robotic hand.



Figure 5: Exploded view of the robotic hand design.

2.2 Design Details

2.2.1 Motor Control System

Motor choice We choose MG995 servo motor [3] as actuator. Because it provides 90 N*cm torque and can be controlled by PWM.

PWM Timing Parameters Timer configurations in MX_TIM2_Init() establish precise timing:

htim2.Init.Prescaler = 899; // 90MHz/(899+1) = 100kHz htim2.Init.Period = 1999; // 100kHz/2000 = 50Hz

These values satisfy:

$$T_{\rm PWM} = \frac{(Period+1)}{f_{\rm TIM2}} = \frac{2000}{100 \rm kHz} = 20 \rm ms$$
(1)

Angle-to-Pulse Conversion The core equation (1) is implemented with bounds checking:

Pulse Width =
$$\left(\frac{\theta}{180^{\circ}} \times 2000 \mu s\right) + 500 \mu s$$
 where $\theta \in [0^{\circ}, 180^{\circ}]$ (2)

Code excerpt from PWM_SetFromAngle():

```
angle = (angle > 180) ? 180 : angle; // Safety clamp
pulse_width = (angle * 2000)/180 + 500;
```

UART Command Processing We designed a USB to serial tool for UART communication (shown in Figure 67). It use CH340, which is widely used in USB to serial.

The interrupt-driven handler processes 5-byte packets at 115200 baud:

```
void HAL_UART_RxCpltCallback() {
    // Parallel actuation + echo
    PWM_SetFromAngle(&htim2, TIM_CHANNEL_1, rx_data[0]);
    HAL_UART_Transmit_IT(&huart6, rx_data, 5); // echo motor angel
}
```

DMA (direct memory access) were considered but rejected for requiring additional buffer management overhead. And for this simply data transfer it is too complex.



Figure 6: schematic for USB to serial board





Figure 7: Layout

Pinout Configuration According to datasheet[4], we assignment pinout. As shown in the STM32 pinout Figure 8:

- UART6: TX(PA11), RX(PA12) for host communication
- PWM: PE9(Ch1), PE11(Ch2), PE13(Ch3), PE14(Ch4)(All use TIM2), PC6(TIM8 Ch1)
- LEDs: PG0–PG4 for channel



Figure 8: STM32 pinout diagram

2.2.2 3D position to angle mapping and filtering

The core functionality of the system involves converting 3D hand landmark positions into finger joint angles. This section details the mathematical foundations and implementation specifics of this mapping process. The whole process can be summarized in Figure 9



Figure 9: Flowchart of 3D Position to Angle Mapping Process

Landmark Position Extraction: The MediaPipe hand tracking solution provides 21 3D landmarks for each detected hand. We extract specific landmarks for each finger:

- **Thumb**: Landmarks 2 (MCP), 3 (PIP), and 4 (TIP)
- Index finger: Landmarks 5 (MCP), 6 (PIP), and 8 (TIP)
- Middle finger: Landmarks 9 (MCP), 10 (PIP), and 12 (TIP)
- Ring finger: Landmarks 13 (MCP), 14 (PIP), and 16 (TIP)
- Pinky finger: Landmarks 17 (MCP), 18 (PIP), and 20 (TIP)

The positions are obtained as normalized coordinates (x, y, z) relative to the hand's bounding box.

Mathematical Formulation: For three points P_1 , P_2 (joint), and P_3 , we compute the angle at P_2 as follows:

- 1. Create vectors $\vec{a} = P_1 P_2$ and $\vec{b} = P_3 P_2$
- 2. Calculate the cosine of the angle using the dot product:

$$\cos\theta = \frac{\vec{a} \cdot \vec{b}}{||\vec{a}|| \cdot ||\vec{b}||} \tag{3}$$

3. Convert to angle in degrees:

$$\theta = \arccos(\cos\theta) \times \frac{180}{\pi} \tag{4}$$

4. Apply a lower bound of 90° to prevent over-flexion:

$$\theta_{\text{final}} = \max(\theta, 90^{\circ}) \tag{5}$$

Calibration and Scaling: The raw angles require calibration to match the target servo range. Each finger has unique scaling parameters:

Finger	Scale Factor	Offset
Thumb	2.0	150°
Index	1.5	180°
Middle	1.5	180°
Ring	1.7	180°
Pinky	1.4	180°

Table 2: Finger Calibration Parameters

The calibration formula for each finger is:

$$\theta_{\text{calibrated}} = \max\left(-(180 - \theta_{\text{raw}}) \times \text{scale} + \text{offset}, 70^\circ\right)$$
(6)

Special handling for the thumb ensures it stays within a practical range (120°-180°).

Filtering and Smoothing: An Exponential Moving Average (EMA) filter is applied to reduce jitter:

$$\theta_{\text{filtered}} = \alpha \times \theta_{\text{new}} + (1 - \alpha) \times \theta_{\text{prev}} \tag{7}$$

- $\alpha = 0.7$ provides a balance between responsiveness and smoothness
- The filter operates on a window of 3 consecutive values
- Integer conversion is applied for final servo commands

Finger State Tracking: A hysteresis-based state machine ensures stable state transitions, as described in Algorithm 1. The scaling factors and the offsets are om Table 2

```
Algorithm 1 Finger State Tracking
```

```
1: Initialize with upper_thresh = 0.95, lower_thresh = 0.85
2: consec\_upper = 5, consec\_lower = 5
3: current_state \leftarrow 'state0' (straight)
4: for each new measurement v do
5:
      if current_state == 'state1' (bent) then
         if v \geq upper_{thresh} then
6:
7:
           upper_counter \leftarrow upper_counter + 1
           if upper_counter ≥ consec_upper then
8:
              current_state \leftarrow 'state0'
9:
           end if
10:
         else
11:
12:
           upper_counter \leftarrow 0
         end if
13:
      else if current_state == 'state0' (straight) then
14:
        if v < \text{lower_thresh then}
15:
           lower_counter \leftarrow lower_counter + 1
16:
           if lower_counter \geq consec_lower then
17:
              current_state \leftarrow 'state1'
18:
           end if
19:
         else
20:
21:
           lower_counter \leftarrow 0
22:
         end if
      end if
23.
24: end for
```

Data Transmission: The final processed angles are sent via serial communication as unsigned 8-bit integers:

- Baud rate: 115200
- Data format: [Thumb, Index, Middle, Ring, Pinky]
- Special handling for ring and pinky fingers (180° angle)

2.2.3 Visual Perception System

Sensor choice : We choose Intel D405 depth camera 10 as the visual sensor, because the Intel D405 provides accurate depth sensing for 3D tracking, while standard webcams only capture 2D images.



Figure 10: Intel D405 Depth Camera

Hardware Configuration Parameters :

- Resolution: 640×480 @30FPS
- Data format: RGB (BGR8) + Depth (Z16)
- Alignment: Depth-to-color alignment (rs.align(rs.stream.color))

Spatial-Temporal Filtering Algorithms

• Spatial Filter:

$$D_{filtered}(x,y) = \frac{\sum_{i=-r}^{r} \sum_{j=-r}^{r} w(i,j) \cdot D(x+i,y+j)}{\sum w(i,j)}$$
(8)

where the kernel weights are:

$$w(i,j) = e^{-\frac{i^2+j^2}{2\sigma_s^2}} \cdot e^{-\frac{|D(x,y)-D(x+i,y+j)|}{2\sigma_r^2}}$$
(9)

Parameter Settings:

- Filter strength (filter_magnitude=2): $\sigma_s = 1.5$
- Hole filling (holes_fill=1): 3×3 neighborhood repair
- Temporal Filter:

$$D_t = \alpha \cdot D_{current} + (1 - \alpha) \cdot D_{t-1} \tag{10}$$

Parameter Settings:

- $\alpha = 0.4$ (filter_smooth_alpha)
- Motion threshold $\delta = 50 \text{ mm}$ (filter_smooth_delta)

MediaPipe Hand Landmark Detection

• Palm Detector (SSD-based)

$$P_{hand} = \text{Sigmoid}(W_{hand}^T \cdot \phi_{RGB}(I) + b_{hand})$$
(11)

where ϕ_{RGB} is the MobileNetV3 feature extractor.

• Landmark Regression (21 points)

$$\mathbf{L}_{2D} = \begin{bmatrix} u_0 & v_0 \\ \vdots & \vdots \\ u_{20} & v_{20} \end{bmatrix} = f_{CNN}(I_{crop})$$
(12)

Parameter Settings:

- Input size: 256×256 (internal model scaling)
- Complexity level: model_complexity=1 (8 CPU inference threads)
- Confidence Mechanism

Tracking Score =
$$0.7 \cdot \text{Detection} + 0.3 \cdot \text{Optical Flow}$$
 (13)

Pixel-to-3D Coordinate Conversion

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = Z \cdot \mathbf{K}^{-1} \cdot \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}, \quad \mathbf{K} = \begin{bmatrix} f_x & 0 & ppx \\ 0 & f_y & ppy \\ 0 & 0 & 1 \end{bmatrix}$$
(14)

Algorithm 2 3D Hand Landmark Detection using RealSense and MediaPipe 1: Initialize RealSense camera pipeline with color and depth streams 2: **Start** stream alignment to color frame 3: **Retrieve** depth camera intrinsics 4: Initialize MediaPipe hand landmark model 5: while True do Acquire aligned color and depth frames 6: 7: if frames are not valid then Continue 8: 9: end if Apply spatial and temporal filters to depth frame 10: 11: Convert filtered depth and color frames to NumPy arrays 12: Convert BGR image to RGB 13: Run MediaPipe hand landmark detection if hand landmarks are detected then 14: for each hand do 15: for each landmark do 16: 17: Convert normalized coordinates to pixel (u, v)18: if (u, v) inside image bounds and valid depth exists then Compute 3D coordinates using: 19: $X = \frac{(u - p_x) \cdot Z}{c}$ 20: f_x $Y = \frac{(v - p_y) \cdot Z}{2}$ 21: Output 3D landmark: (X, Y, Z)22: end if 23: 24: end for 25: end for end if 26: 27: Display annotated color and depth images if ESC is pressed then 28: 29: Break end if 30: 31: end while 32: Stop RealSense pipeline and close all windows

The Pseudocode for hand landmark detection system

2.2.4 Robotic Hand Mechanical Design

Reel design Converting the servo motor's rotational motion into the linear reciprocating motion of the tendon posed a significant design challenge. Initially, several mechanisms were explored, such as gear racks, cam systems, and linkage structures. However, these designs introduced structural complexity, reduced the mechanical stability of the robotic hand, and resulted in a bulky connection between the servo motor and the tendon.

To address these challenges, a more streamlined structure was developed, as shown in Figure 11. In this design, the tendon first wraps around a circular pulley via a guide pipe, completes one loop around the pulley, and then passes through holes with diameters of 2 mm and 5 mm. The end of the tendon is anchored at the 5 mm hole. The pulley ensures smooth guidance of the tendon while minimizing friction and wear. The pipe serves as a guiding channel for the tendon, preventing it from slipping off the pulley and ensuring consistent motion. The fixation point securely anchors the tendon, eliminating any risk of slippage during operation. This configuration ensures that the tension in the tendon is evenly distributed, reducing wear and improving the overall stability of the motion.

During installation, the servo motor is rotated to its maximum limit in one direction. The tendon emerging from the pipe is then tangentially aligned with the pulley's rotation and connected to the fixed position on the finger. By precisely controlling the rotation angle of the servo motor, accurate control over the tendon's movement length is achieved, which directly translates to the finger's motion. Additionally, the mounting interface allows for easy and secure installation of the servo motor, ensuring proper alignment with the tendon mechanism.

This design ensures both compactness and mechanical efficiency, addressing the challenges of motion conversion while maintaining the overall stability and reliability of the system. Furthermore, this approach minimizes the size of the tendon-motor connection structure, making it easier to integrate into the robotic hand without compromising its performance. By optimizing the interaction between the components, the system achieves high precision and durability, making it suitable for various applications.

Motor Base Design The design of the motor base was carefully considered to ensure both structural compatibility with the robotic hand and ease of integration with existing robotic arms. As illustrated in Figures 12 and 13, both the engineering drawing and the conceptual sketch of the motor base are provided.

In the sketch, the five dashed circles indicate the positions designated for the reels, while the five solid rectangles correspond to the mounting locations for MG 995 servos. The five dashed lines extending to the origin represent the cable pathways. Additionally, the four holes on the right side, each with a diameter of 4 mm, are used for mounting the main control board. The dimensions of the motor base have been precisely matched to the size of the designed robotic hand, ensuring a harmonious appearance and proper mechanical fit.



Figure 11: Engineering drawing of the reel.

Beyond securing the relative positions of the motors and the robotic hand, the motor base also serves as the "forearm" of the system, enabling straightforward attachment to existing robotic arms such as the ARX5. This dual functionality simplifies assembly and enhances the modularity of the overall robotic system.



Figure 12: Engineering drawing of the motor base.



Figure 13: Sketch of the motor base.

3 Verification

3.1 PWM Signal Validation



Figure 14: PWM waveform validation using SIGLENT SDS 2304X

Test Procedure:

- 1. Initialized TIM2/TIM8 with prescaler 899 (72MHz/900 = 80kHz clock)
- 2. Programmed servo angles from 0-180° using <code>PWM_SetFromAngle()</code>
- 3. Measured pulse width using oscilloscope

Results:

- 1. Pulse width error: $\pm 0.8\%$ (0.5-2.5ms range)
- 2. Frequency stability: $50Hz \pm 0.1Hz$
- 3. Code validation: Confirmed timer configuration matches Equation 1

3.2 UART Communication Test

Test Setup:

HAL_UART_Receive_IT(&huart6, rx_data, 5); // 5-byte payload

Validation Metrics:

Parameter	Measured	Requirement	
Baud rate	$115200 \pm 2\%$	3% tolerance	
Latency	10ms	less than 200ms	
Packet error rate	0.1%	less than 1%	

Table 3:	UART	performance	metrics
----------	------	-------------	---------

3.3 Visual Perception Spatial Accuracy Verification

Test Procedure

- Conducted under varied conditions including:
 - Open-hand postures (full finger extension)
 - Hand-object interactions (smartphone grasping)
- Measured Mean Per Joint Position Error (MPJPE) for 21 landmarks

Results

- Achieved 4.3mm average MPJPE @50cm working distance
- Maintained spatial consistency during dynamic gestures
- Successful landmark tracking during open-hand postures (Figure 15) object occlusion (Figure 16)



Figure 15: 3D hand landmark output during open-hand gesture



Figure 16: 3D hand landmark output during object grasp

3.4 Real-Time Performance Evaluation

Test Procedure

- Measured end-to-end processing latency:
 - Frame capture to 3D coordinate output
 - 60-second continuous operation tests
- Evaluated under CPU load conditions (simulating full system integration)

Results

- Sustained 30 FPS throughput (200% above 15 FPS requirement)(Figure 17)
- Maximum latency: 38.2ms



Figure 17: Measured runtime frame rate of 30 FPS

3.5 Grasping Verification

Our robotic hand is capable of effectively grasping test objects such as cuboid boxes and paper cups. The following images are screenshots taken during the testing process.





Figure 18: Snapshots of the robotic hand grasping test objects during the evaluation process. (a) Grasping a paper cup. (b) Grasping a cuboid box.

4 Costs

Here is the BOM of our design.4

Parts	Description	Price (RMB)	Qty	Total (RMB)
MG 995	Servo motor	10	5	50
DJI A Board	Microcontroller	borrow from lab	1	0
DJI TB47	24V Battery	borrow from lab	1	0
CH 340	UART TO USB	20	1	20
Finger Hinge	85A TPU	50	1	50
Intel D405 Depth Camera		500	1	500
РСВ	USB-TTL	30	1	30
Xihe Shao labor		10	240 hours	2400
Qihan Shan labor		10	240 hours	2400
Sizhao Ma labor		10	240 hours	2400
Jun Liang labor		10	240 hours	2400
TOTAL				10250

Table 4: Bill of Materials

5 Conclusion

5.1 Executive Summary

This project explores the feasibility of a vision-based robotic hand system designed to replicate human gestures without the use of wearable sensors. While the current proto-type demonstrates the potential for remote operation in hazardous environments, it also highlights several technical limitations that must be addressed before such a system can be considered robust or deployable.

5.2 Accomplishments

The system successfully captures basic hand movements using stereo camera input and translates them into motor commands with a response time under 200 ms. A modular, 3D-printed robotic hand was developed and tested with different materials and connector stiffness levels to improve mechanical performance. Several design iterations were implemented to optimize thumb positioning and increase joint stability.

These efforts resulted in a working prototype that is capable of mimicking simple grasping and finger-opening motions, demonstrating the core concept behind vision-guided robotic manipulation.

5.3 Limitations and Improvements

Despite these achievements, the current prototype has some limitations. The robotic hand is constructed primarily from rigid PLA, which limits its ability to firmly grip objects. Additionally, each finger has only one degree of freedom, restricting the range of motion and dexterity.

The gesture tracking system also lacks consistent reliability, often being affected by lighting conditions, hand orientation, and background interference. These issues result in occasional misinterpretation of gestures, limiting the system's usability outside of controlled environments.

To address these limitations, several improvements are recommended for future development. Exploring soft robotics or multi-joint finger designs could significantly enhance grip adaptability and dexterity. Upgrading the vision system, such as by integrating more advanced pose estimation models, could lead to more stable and accurate gesture tracking.

5.4 Ethical Considerations

In alignment with the IEEE Code of Ethics[5], this project aims to support safer humanrobot interaction by reducing exposure to hazardous environments. However, as a prototype, it is not yet suitable for real-world deployment. Continued development should emphasize transparency regarding system capabilities and prioritize user safety during testing and iteration. And also consider accessibility and responsible use when developing assistive or industrial robotic systems.

5.5 Broader Impacts

Despite limited functionality, this project advances low-cost robotics through a modular design and accessible components, offering potential for education and use in resource-constrained settings. The use of 3D printing supports localized production and reduces material waste. By eliminating wearable sensors and utilizing off-the-shelf parts, the system lowers costs compared to existing solutions. Further development could enable safer automation in industrial and assistive applications.

References

- [1] *Intel*® *realsense*[™] *depth camera d400 series product family datasheet*, Document Number: 337029-002, Intel Corporation, 2022. [Online]. Available: https://www.intelrealsense. com/depth-camera-d405/ (visited on 04/15/2023).
- [2] Google. "Mediapipe hands." (2021), [Online]. Available: https://google.github.io/ mediapipe/solutions/hands (visited on 04/15/2023).
- [3] TowerPro, Mg995 servo motor datasheet, 2019.
- [4] *Stm32f4xx hal library user manual*, STMicroelectronics, 2021.
- [5] IEEE. ""IEEE Code of Ethics"." (2016), [Online]. Available: https://www.ieee.org/ about/corporate/governance/p7-8.html (visited on 02/08/2020).