1 Introduction

1.1 Objective and Background

Humans working in hazardous environments, such as those involving toxic materials, high-pressure equipment, or explosives, face life-threatening risks. Current solutions, such as sensor-equipped gloves or pre-programmed robots, suffer from inflexibility, discomfort, high cost, and dependence on physically wearable devices, limiting their usefulness. To address these challenges, this project introduces a vision-based robotic hand system that utilizes camera tracking and 3D-printed components to enable real-time human gesture imitation. The solution moves away from reliance on wearable sensors and prioritizes adaptability. The system aims to redefine safety and accessibility in hazardous operations.

1.2 Goals

Its main goal is to reduce human exposure in hazardous environments by replacing rigid, sensor-dependent solutions with a flexible, vision-driven robotic system that replicates human gestures in real time.

1.3 Functions

The system (1) captures and interprets human hand gestures via cameras, (2) translates them into precise robotic movements using closed-loop feedback, and (3) employs a modular, 3D-printed robotic hand for safe manipulation of hazardous objects.

1.4 Benefits

Consumers gain a non-invasive, low-cost alternative to traditional methods, eliminating the need for restrictive wearables while improving operational safety, adaptability, and ease of deployment in high-risk scenarios.

1.5 Features

- Camera-driven gesture recognition (no physical sensors required)
- Real-time mimicry with closed-loop feedback for sub-millisecond precision
- Affordable 3D-printed construction (80% cost reduction vs. industrial robots)

1.6 High-Level Requirement List

- The system must achieve **gesture recognition accuracy ≥ 95%** under varying lighting and occlusion conditions.
- End-to-end latency between human gesture and robotic response must be **≤ 200 ms** to ensure real-time performance.

- The robotic hand system must demonstrate **zero unintended motions** during a **1-hour** stress test under error-inducing conditions (e.g., sensor noise, communication latency, or environmental disturbances).

2 Design

2.1 Block Diagram

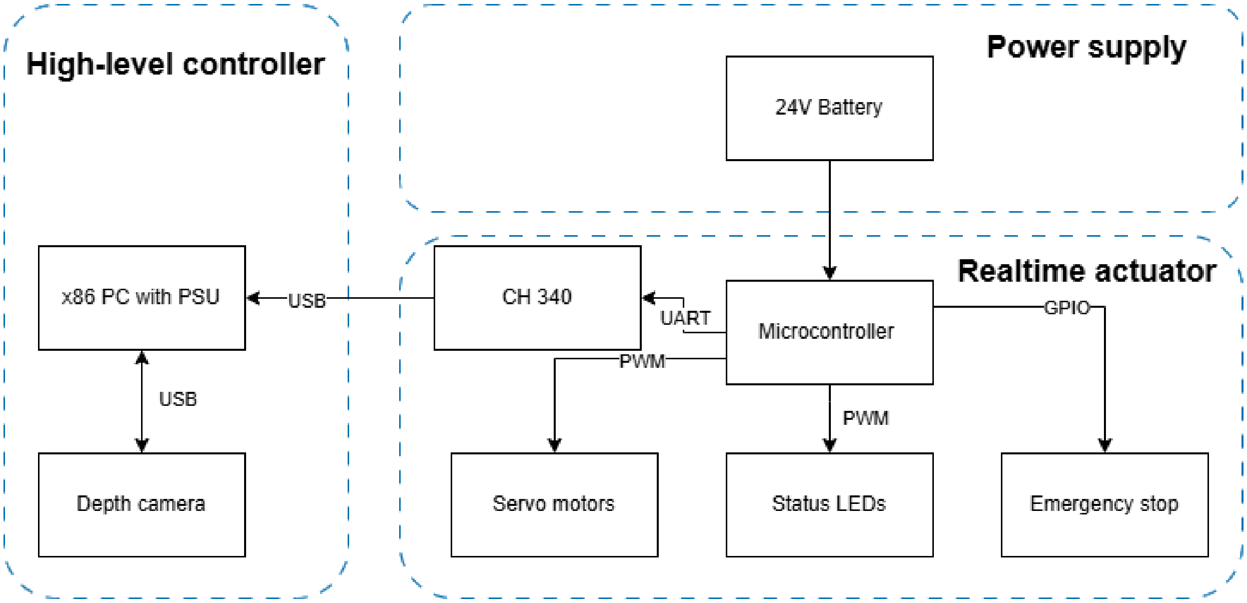
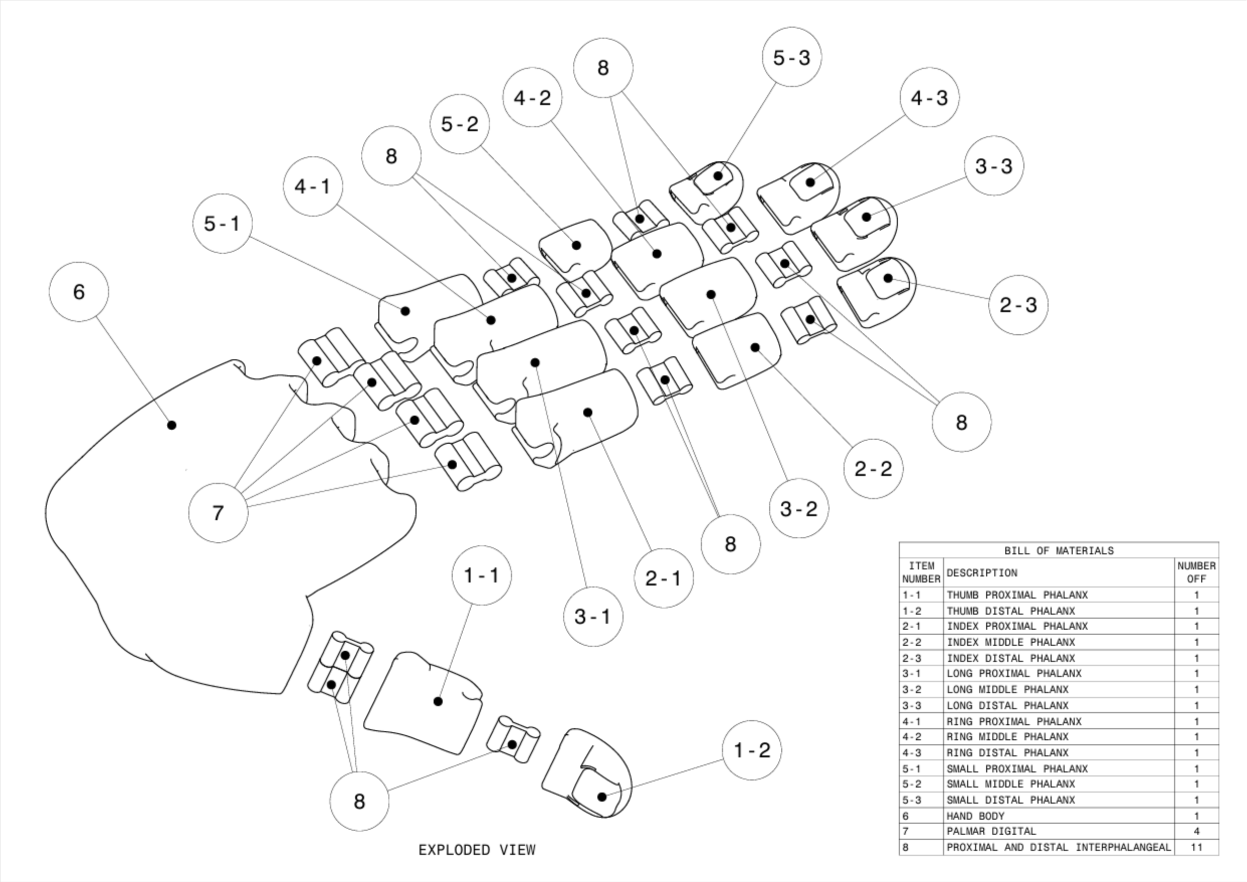


Figure 1: Block Diagram

2.2 Physical Design Diagram



BILL OF MATERIALS		
ITEM NUMBER	DESCRIPTION	NUMBER OFF
1-1	THUMB PROXIMAL PHALANX	1
1-2	THUMB DISTAL PHALANX	1
2-1	INDEX PROXIMAL PHALANX	1
2-2	INDEX MIDDLE PHALANX	1
2-3	INDEX DISTAL PHALANX	1
3-1	LONG PROXIMAL PHALANX	1
3-2	LONG MIDDLE PHALANX	1
3-3	LONG DISTAL PHALANX	1
4-1	RING PROXIMAL PHALANX	1
4-2	RING MIDDLE PHALANX	1
4-3	RING DISTAL PHALANX	1
5-1	SMALL PROXIMAL PHALANX	1
5-2	SMALL MIDDLE PHALANX	1
5-3	SMALL DISTAL PHALANX	1
6	HAND BODY	1
7	PALMAR DIGITAL	4
8	PROXIMAL AND DISTAL INTERPHALANGEAL	11

Figure 2: Exploded view of the robotic hand design.

As illustrated in Figure 2, the design is inspired by the human hand and consists of modular finger segments (proximal, middle, and distal phalanges). Most components are made of rigid PLA material for structural stability, while parts 7 and 8 utilize flexible TPU material (85A hardness) to enable adaptive joint motion. The modular architecture allows for easy assembly and customization, with internal integration potential for tendon-driven mechanisms or sensors. This design supports precise human gesture replication and is ideal for hazardous environment applications, robotics research, and educational purposes.

2.3 Realtime Actuator

2.3.1 Microcontroller

The DJI A Board is a high-performance microcontroller that serves as the central control unit for the robotic hand system. It is built with an ARM Cortex-A processor, offering substantial computational power to handle real-time processing tasks such as gesture recognition, motor control, and sensor data integration. The hardware is optimized for robotics applications, ensuring smooth and efficient operation even under complex workloads.

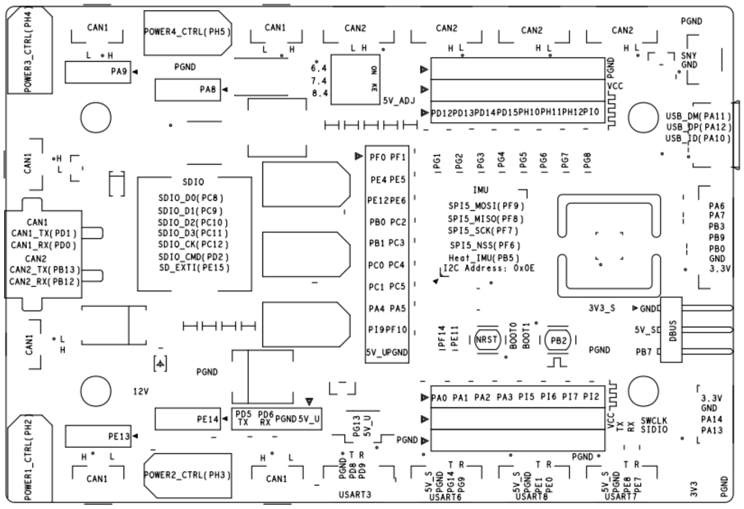


Figure 3: DJI A Board schematic diagram

This microcontroller enables seamless communication with other subsystems through its support for multiple interfaces, including UART, SPI, I2C, and PWM. These communication protocols make it highly adaptable, allowing integration with various sensors, actuators, and peripherals. Additionally, the DJI A Board has onboard hardware acceleration for AI processing, which is particularly valuable in vision-based systems that require low latency and high accuracy.

The DJI A Board's modular design and compact size make it ideal for embedding into the robotic hand system. It offers robust SDK support, simplifying the development of control algorithms and ensuring compatibility with a wide range of software tools. Its reliability and flexibility make it a key component in achieving precise and responsive robotic hand movements.

Table 1: Requirements and Verification for Microcontroller

Requirements	Verification
1. The microcontroller must execute motor control algorithms with low latency.	Test the system with motors under load and measure response time to ensure real-time performance.
2. It must support PWM output for driving servo motors.	Validate PWM signal generation using an oscilloscope and verify its compatibility with actuators.
3. The microcontroller should process data from the depth camera in real-time.	Conduct tests by simulating human gestures and evaluating the response time of the robotic hand.
4. The microcontroller should maintain stable operation under high computational loads.	Stress-test the controller with simultaneous motor actuation and sensor data processing tasks.

2.3.2 Servo Motor

The MG995 Servo Motor is a high-torque servo widely used in robotics applications due to its durability, precision, and ability to handle significant mechanical loads. In the robotic hand system, the MG995 plays a crucial role in controlling the motion of finger joints, enabling smooth and realistic replication of human hand movements.

This servo motor operates on a 5–7V AC power supply and is driven by a 50 Hz PWM signal with an adjustable duty cycle, which determines the position of the servo. The MG995 is designed to deliver reliable performance under varying conditions, with a stall torque of up to 10 kg/cm, making it suitable for applications requiring precise and strong actuation.

The MG995 features a metal gear train, which enhances its longevity and ability to handle high torque without deformation. Its fast response time and minimal backlash ensure accurate and repeatable motion, which is essential for the robotic hand to achieve natural gestures. Additionally, its compact size allows for easy integration into the robotic hand's modular design.

Table 2: Requirements and Verification for Servo Motor

Requirements	Verification
1. 50 Hz PWM signal output with adjustable duty cycle and 5–7V AC power supply.	Use an oscilloscope to check whether the output signal is correct. Use a voltage meter to check power supply
2. Accurate positional control with minimal backlash.	Perform repeatability tests by commanding the servo to specific positions and measuring its accuracy.
3. High torque output of at least 10 kg/cm for reliable actuation.	Test the servo underload to ensure it can handle the required torque without stalling.
4. Stable operation under continuous use.	Conduct endurance tests by running the servo continuously for extended periods under load.

2.3.3 UART to USB

The CH340 is a USB-to-UART bridge chip used to facilitate communication between the robotic hand system's microcontroller (DJI A Board) and external devices such as the x86 PC. The CH340 enables seamless data transmission by converting UART signals to USB, allowing the microcontroller to interface with devices that do not have native UART support. This component is essential for transferring control commands, debugging data, and real-time feedback between the high-level controller and the rest of the system.

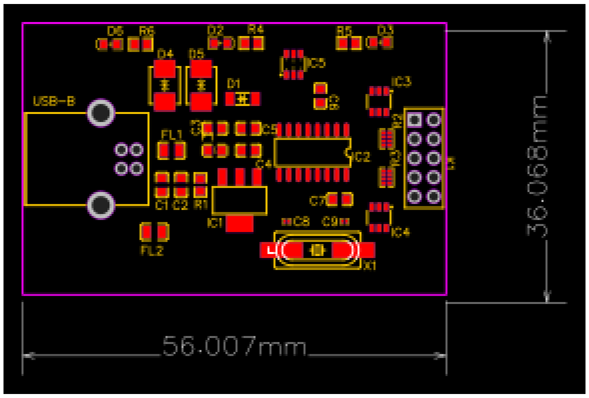


Figure 4: CH 340 PCB schematic diagram

The CH340 is designed for simplicity and reliability, supporting standard baud rates from 50 bps to 2 Mbps, which makes it highly compatible with various robotic applications. It features low power consumption, making it ideal for embedded systems like the robotic hand. Additionally, its compact form factor ensures easy integration into the overall hardware architecture.

The CH340 supports plug-and-play functionality, requiring minimal configuration for use. Its robust driver support across major operating systems (Windows, macOS, and Linux) ensures compatibility and ease of use during development and operation. This makes the CH340 a critical component for establishing a stable and efficient communication link in the robotic hand system.

Table 3: Requirements and Verification for CH 340

Requirements	Verification
1. Full-Duplex communication between microcontroller and PC at115200 baud rate.	Send test message in full speed and keep 10 min stress test. Check if there is any error message.
2. Seamless USB-to-UART conversion without data loss or corruption.	Perform data loopback tests to ensure accurate signal conversion.
3. Low power consumption suitable for embedded systems.	Measure the current draw of the CH340 module under normal operation using a multimeter.
4. Stable operation during continuous data transmission.	Conduct stress tests by transmitting large volumes of data over an extended period and monitoring for errors.

2.4 Power Supply

2.4.1 24V Battery

The DJI TB47 Intelligent Flight Battery serves as the power supply for the robotic hand system, providing a stable and reliable source of energy for all the subsystems, including the microcontroller, servo motors, and peripherals. This battery is specifically designed for high-performance applications, delivering a nominal voltage of 22.2V and a capacity of 4500mAh, which ensures sufficient power for prolonged operation.

The TB47 features advanced battery management capabilities, including over-voltage, over-current, and temperature protection. These features ensure the safety and longevity of the battery while maintaining stable performance under varying load conditions. Its smart design includes real-time monitoring of battery status, such as remaining charge and health, which can be accessed through the system interface for effective energy management.

In addition, the TB47's lightweight and compact design make it ideal for integration into the robotic hand system, where weight and space are critical considerations. The battery's high discharge rate ensures that it can handle peak loads, such as those experienced during the simultaneous actuation of multiple servo motors. With its proven reliability in demanding environments, the TB47 is a robust solution for powering the robotic hand system.

Table 4: Requirements and Verification for 24V Battery

Requirements	Verification
1. Nominal voltage of 22.2V with a capacity of 4500mAh.	Measure the output voltage and capacity using a multimeter and battery tester.
2. Must provide a stable power supply to all subsystems under varying loads.	Test the system under full load and monitor for voltage drops or instability.
3. Support for peak current draw during simultaneous operation of multiple motors.	Measure current draw during peak load scenarios and ensure the battery can handle the demand.
4. Real-time monitoring of battery status (charge, health, etc.).	Access battery status through the system interface and verify the accuracy of reported metrics.

2.5 High-level Controller

2.5.1 Depth Camera

The Intel RealSense D405 Depth Camera serves as the core 3D vision sensor for the hand-tracking system, providing high-precision depth perception and RGB imaging for accurate real-time gesture recognition. This camera is specifically optimized for close-range applications, offering an ideal working range of 7cm to 50cm and a theoretical depth resolution down to 0.1mm, which ensures exceptional accuracy in capturing fine hand movements and spatial details.

The D405 features advanced hardware capabilities, including a 90fps RGB frame rate and 1280×720 resolution, enhanced by an integrated Image Signal Processor (ISP) for superior image quality in varying lighting conditions. Its compact design (42×42×23mm) and USB 3.0 connectivity make it ideal for embedded systems, robotic vision, and augmented reality applications.

Equipped with active stereo infrared (IR) sensing, the D405 delivers robust depth data even in low-texture environments, while its factory-calibrated optics eliminate the need for manual recalibration. The camera also supports real-time depth-RGB alignment, ensuring seamless integration with vision-based AI models like MediaPipe Hands for stable 3D skeletal tracking. Additionally, the Intel RealSense SDK provides comprehensive tools for depth filtering, noise reduction, and system diagnostics, enabling efficient deployment in dynamic interactive systems.

Table 5: Requirements and Vertification for Depth Camera

Requirement	Verification
Operating temperature must be between 0°C and 35°C to ensure stable performance.	Test image and depth output quality at 0°C, 20–25°C, and 35°C, ensuring no frame drops or data errors.
Camera must be powered via USB 3.1 Type-C at 5V, max current 1A.	Use USB power meter to measure voltage/current and confirm stable operation over extended periods.
Effective depth sensing range must be between 7 cm and 50 cm.	Place objects at key distances and verify depth data output is valid and accurate.
Camera must function under natural and low indoor lighting conditions (≥10 lux).	Test camera in normal and low-light (≥10 lux) conditions to evaluate noise and recognition reliability. Requirement

Figure 5: Intel D405 Depth Camera

…

2.6 Schematics, software flow charts, calculations, and simulation

2.6.1 Visual perception algorithm of hands pose estimation based on Mediapipe

The MediaPipe Hands model is a real-time hand tracking solution developed by Google. It detects and tracks 21 3D landmarks of a human hand from a single RGB image or video frame. The model combines palm detection and landmark regression to achieve high accuracy and low latency.

It consists of two stages:

1. Palm Detection – Identifies the presence and region of a hand in the image.
2. Hand Landmark Model – Predicts 21 3D landmarks within the detected region, representing key finger joints and wrist positions.

MediaPipe Hands is highly optimized for mobile and embedded platforms, making it suitable for real-time gesture recognition in interactive systems.

Table 6: Requirements and Vertification for Mediapipe

Requirement	Verification Method
The model must detect and return 21 3D hand landmarks in each frame.	Use sample video frames to verify that the model returns 21 landmark points consistently for a variety of hand poses.
The model must run in real-time (≥15 FPS) on the target platform.	Measure the model's runtime performance using profiling tools (e.g., OpenCV, MediaPipe built-in logs) and ensure average FPS ≥ 15.
The model must detect hands with high accuracy (≥90%) under various hand poses.	Conduct testing with a dataset containing different hand poses and calculate detection success rate. Accuracy must be ≥ 90%.
The model must tolerate minor occlusions (e.g., partially hidden fingers) and still provide stable landmark tracking.	Simulate common occlusion scenarios (e.g., hand partially outside frame or overlapping fingers) and verify landmark consistency across frames.
The model must operate under varying lighting conditions, including indoor and natural light.	Test the model in environments with different light intensities and evaluate detection stability and landmark precision.

2.6.2 Filtering algorithm of depth graph

To improve the reliability and accuracy of depth data captured by the Intel RealSense D405 camera, both spatial and temporal filtering techniques are applied to the raw depth frame.

1. Sptial Filtering

The spatial filter smooths pixel-level noise using a weighted average of neighboring pixels. It can be expressed as:

$$D_s(x,\ y)\ =\ \left(\frac{1}{W}\right)\cdot\ \sum_{(i,j)\in N(x,y)} w_{\{i,j\}}\cdot D(i,\ j)$$

where:

- $D(i,\ j)$ is the raw depth value at pixel (i, j),
- $N(x,\ y)$ is the neighborhood of pixel (x, y),
- $w_{\{i,j\}}$ is the spatial weight (based on distance and/or depth similarity),
- W is the normalization factor, $W\ =\ \sum w_{\{i,j\}}$.

The filter also performs hole filling, which interpolates missing depth data (usually marked as 0) using surrounding valid pixels.

2. Temporal Filtering

The temporal filter reduces flickering by combining the current frame with past depth data using an exponential moving average:

$$D_t\ =\ \alpha\ \cdot\ D_{current}\ +\ (1-\alpha)\cdot D_{prev}\ ,\ \text{if}\ \left|D_{current}-D_{prev}\ \right|<\delta$$

where:

- $D_{current}$ is the current depth value,
- D_{prev} is the value from the previous filtered frame,
- α ∈ [0, 1] controls the smoothing strength (set to 0.4),
- δ is the maximum allowable depth jump for filtering (set to 50 mm).

If the depth change exceeds δ, the filter favors the current value to preserve responsiveness.

This combination ensures both spatial consistency (smooth surfaces) and temporal stability (minimal flicker) in the depth output, while keeping latency low enough for real-time applications.

Table 7: Requirements and Vertification for Filtering Algorithm

Requirement	Verification Method
The system must apply spatial filtering to reduce pixel-level noise in depth maps.	Visually compare raw and filtered depth maps using test scenes; noise level should visibly decrease in flat surfaces.

The filter must support hole-filling to interpolate missing depth values in small regions.	Compare regions with missing depth (black holes) before and after filtering; measure pixel fill rate improvement.
The system must apply temporal filtering to stabilize depth data over time.	Record depth frames of a static object and check for fluctuations before and after filtering; depth value variance should be reduced.
Filtering must run in real-time (≥15 FPS) on the target hardware.	Use profiling tools to measure frame processing speed after filtering; ensure processing rate stays above 15 frames per second.

2.6.3 Pixel-to-3D algorithm

The system uses a standard pinhole camera model to convert 2D pixel coordinates into real-world 3D space. Given a pixel (u,v)(u, v)(u,v) from the depth image and its corresponding depth value ZZZ (in meters), the coordinates in 3D camera space (X,Y,Z)(X, Y, Z)(X,Y,Z) are computed using the intrinsic parameters of the depth camera:

$$X = \frac{(u - p_x) \cdot Z}{f_x}$$

$$Y = \frac{\left(v - p_y\right) \cdot Z}{f_y}$$

$Z =$ Depth_frame.get_distance(u, v)

where:

$\left(p_x,\, p_y\right)$a re the principal point offsets (optical center),

$\left(f_x,\, f_y\right)$are the focal lengths (in pixels),

Z is the depth value at pixel$(u,\, v)$from the depth sensor.

The implementation includes boundary checking for pixel validity and ignores points with invalid or zero depth. This conversion is essential for mapping image-based hand gestures into spatial control commands for robotic applications.

Table 8: Requirements and Vertification of Pixel-to-3D algorithm

Requirement	Verification Method

The algorithm must correctly convert valid 2D pixel coordinates with depth into 3D coordinates using intrinsic parameters.	Test with known calibration parameters and check that 3D points lie within expected spatial range.
The function must return a sentinel value (e.g., Z = -1) when pixel coordinates are out of bounds.	Provide out-of-bound pixel inputs and verify that output contains Z = -1 for invalid data.
The function must return a sentinel value when depth value is zero or invalid.	Input pixels with zero depth and verify returned Z = -1.
The conversion must use camera intrinsics including focal length and principal point.	Check that the implementation references `fx`, `fy`, `px`, `py` from the camera intrinsics.
The conversion accuracy must be within ±2 cm at 30–50 cm distance.	Compare output of pixel_to_3d against ground truth positions measured in real-world test setup.
The conversion must process at least 1000 pixels per second to meet real-time requirements.	Benchmark the function by running pixel-to-3D conversion on 1000 random pixels and ensure total time < 1 second.

2.6.4 3D position to angle mapping

Finger flexion angles are computed using a vector-based approach. For each finger, three key joints (proximal, intermediate, and distal) define two vectors: one from the proximal to the intermediate joint and another from the intermediate to the distal joint. The angle between these vectors is calculated using the dot product formula:

$$\cos(\theta) = \frac{\vec{a} \cdot \vec{b}}{||\vec{a}|| \cdot \left||\vec{b}|\right|}$$

which is converted from radians to degrees for intuitive interpretation. To handle tracking instability, the system retains the previous valid angle if any joint’s depth data is invalid, preventing sudden jumps in the output.

To further reduce noise, an Exponential Moving Average (EMA) filter smooths the computed angles across frames. The EMA applies a smoothing factor (α = 0.2), balancing responsiveness and stability. Additionally, a threshold-based state machine classifies each finger as either straight (state0) or bent (state1). Hysteresis is implemented to avoid rapid toggling between states: a finger must remain above a 0.95 cosine threshold for five consecutive frames to be considered straight, or below 0.85 for five frames to be considered bent. This debouncing mechanism ensures reliable state transitions. The whole process of this part can be summarized by Figure 6.

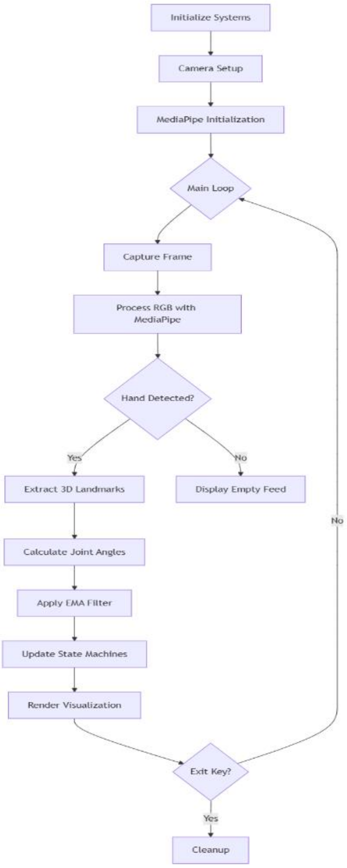


Figure 6: 3D position to angle mapping algorithm

The requirement and verification for this algorithm are summarized in Table 9 and Table 10 below:

Table 9: Functional Requirements

Requirement	Verification Method
Real-time 3D Hand Tracking	The system must detect and track 21 hand landmarks at ≥30 FPS.
Depth-Enhanced 3D Mapping	Each landmark must be converted to 3D world coordinates using depth data.
Finger Flexion Angle Calculation	Compute angles between finger joints (e.g., MCP, PIP, DIP) using vector math.
Noise-Robust Angle Smoothing	Apply EMA filtering to stabilize angle outputs.
Finger State Classification	Classify fingers as "straight" or "bent" using hysteresis thresholds.

Table 10: Functional Verification

Requirement	Verification Method

Run MediaPipe on a test video with known hand poses.	All 21 landmarks detected with ≥95% accuracy.
Compare RealSense depth-based 3D coordinates against a motion-capture system.	Mean error <5mm in XYZ coordinates.
Manually measure finger angles with a goniometer and compare with algorithm output.	Computed angles within ±5° of ground truth.
Introduce synthetic noise to angle data and verify EMA reduces jitter.	Filtered angles show ≤2° fluctuation under noise.
Simulate finger bending/straightening and check state transitions.	Correct classification with no false triggers.

2.7 Tolerance Analysis

2.7.1 MediaPipe Tracking Accuracy

Issue: Poor lighting or occlusions may reduce tracking accuracy.

Mitigation: Apply Kalman filtering and ensure controlled lighting in the environment.

2.7.2 Motor Overload

Issue: Risk of overload in Motors 2 and 3.

Mitigation: Implement torque limits and use mechanical slip clutches.

2.7.3 Battery Safety

Issue: Potential fire hazard from LiPo batteries.

Mitigation: Integrate a battery management system (BMS) with voltage cutoff and fuse protection.

3 Cost analysis (parts and labor)

Parts	Description	Price (RMB)	Qty	Total
MG 995	Servo motor	10	5	50
DJI A Board	Microcontroller intergrade part	495	1	495
DJI TB47	24V Battery	200	1	200
CH 340	UART TO USB	20	1	20
Finger Hinge	85A TPU	50	1	50
Intel Camera		500	1	500

4 Schedule

Week (Starting)	Task Description
2/24	Low-level control module development (Completed)
3/3	Vision identification implementation (Completed)
3/10	Mechanical design and 3D printing of robotic hand (Completed)
3/17	Buffer period / Integration planning
3/24	Begin unit testing for individual modules
3/31	Continue unit testing and correct any functional issues
4/7	Finalize unit testing for all modules
4/14	Start assembling robotic hand and perform basic functionality checks
4/21	Improve system stability and control accuracy
4/28	Final integration and stress testing
5/5	Develop demo code for final presentation
5/12	Refine demo functionality and test under presentation scenarios
5/19	Complete remaining sections of the thesis
5/26	Finalize thesis and submit full project documentation

5 Ethics and Safety

5.1 Ethics

Privacy and Data Protection:

Since the system relies on vision-based tracking, it must ensure that user data is handled securely. Potential concerns include:

– Unauthorized data collection or storage.

– Risk of surveillance or misuse of visual data.

– Measures such as local processing, encryption, or anonymization should be implemented to mitigate these risks.

- Reliability and Safety in Hazardous Environments:

– Errors in real-time mimicry could result in dangerous situations, such as mishandling toxic materials or explosives.

– Fail-safe mechanisms and rigorous testing are necessary to prevent malfunctions that could endanger workers.

5.2 Safety

Our project follows ECE 445 Safety Guidelines to ensure a safe working environment for both developers and end-users. Below, we address potential safety concerns related to electrical, mechanical, and lab safety while justifying areas with minimal safety risks.

5.2.1 Electrical Safety

Our system primarily uses low-voltage electronics for gesture tracking and robotic hand control. Since it does not involve high voltage, the risk of electrical hazards is minimal.

If modifications introduce high-voltage components in the future, we will complete the required high-voltage safety training and follow safe electrical handling procedures.

The system does not involve direct electrical contact with human users, eliminating risks related to electric current exposure.

5.2.2 Mechanical Safety

The robotic hand is 3D-printed and designed to operate with low force and torque, minimizing the risk of injury.

Moving parts could pose a pinching hazard during testing. To mitigate this:

Team members will keep their hands away from moving parts during testing and use tools for assembly or adjustments.

5.2.3 Lab Safety

Lab Presence Requirement: At least two team members will always be present in the lab when working on the project.

Mandatory Safety Training: All team members will complete the online safety training and submit certificates on Blackboard before starting lab work.

5.2.4 End-User Safety

The robotic hand system is designed for remote teleoperation in hazardous environments, reducing risks to human workers.

Software reliability is crucial to prevent misinterpretation of gestures, which could lead to incorrect robotic actions. To address this:

We will implement closed-loop feedback mechanisms to ensure accurate mimicry.

Thorough testing will be conducted before deployment in real-world hazardous environments.

5.2.5 Safety Plan

Prevention: Strict hardware and software testing to identify failure points.

Fail-Safe Mechanisms: Emergency stop features to halt movement in case of errors.

Protective Measures: Ensure non-harmful force output from robotic movements.

Training & Documentation: Team members will review safety protocols before operation.

5.2.6 Justification for Minimal Safety Concerns

The project does not involve hazardous chemicals, explosive materials, or biological risks.

It operates at low voltage and does not require direct human-electrical contact.

The robotic hand has low mechanical force, and risks such as pinching are mitigated through design and controlled movement.

6 Citations and References

- TB47MSDS(Amperex Technology). https://www.amperextechnology.com/tb47-msds

- MediaPipe Hand Tracking Documentation. https://google.github.io/mediapipe/solutions/hand_tracking

- IEEE Code of Ethics. https://www.ieee.org/about/corporate/governance/p7-8.html