New York University

Mechanical & Aerospace Engineering (MAE)

E2 - Project Report

Single-DOF Gesture Controlled Mechatronic Finger

Grishma Balgi, Tejas Mathur Sriganesh, Pavan Kushal Velagaleti

Affiliation: NYU Tandon School of Engineering, MAE

Course: ROB-GY 5103 Mechatronics

Prof: Nicholas Chbat

Date: 19 December, 2025

ABSTRACT

The paper describes the design, modeling, and verification of a vision-based, gesture-controlled robotic finger within the paradigm of human-in-the-loop control. Human finger motion is intercepted using a monocular camera and further processed by computer vision for generating a reference command for a motor-driven robotic finger. The robotic finger actuation makes use of a DC motor coupled to a tendon driven transmission and is controlled using closed-loop feedback provided by a rotary encoder. In turn, the system is modeled as a single-input single-output electromechanical plant and an appropriate PD controller has been designed which ensures stability with accuracy in tracking. A validation of the control design has been done by running MATLAB/Simulink simulations before actual hardware implementation. The project focuses on the system-level integration of perception with control, embedded firmware, and mechanical actuation-a stepping stone toward multi-finger robotic hand systems.

INDEX

1. INTRODUCTION

In medical and laboratory settings where accuracy, consistency, and sterility are crucial, robotic systems are being used more and more. Because of contamination risks, operator fatigue, and ergonomic limitations, direct human interaction with instruments is frequently restricted in such environments. Robotic manipulators enable safe, reliable, and repeatable task performance by separating human intent from physical execution.

By enabling natural human motion to directly control robotic systems, gesture-based teleoperation provides an intuitive method of human–robot interaction. Because vision-based sensing reduces system complexity and contamination risk by doing away with wearable sensors and physical contact, it is especially appealing in sterile settings.

The creation of a gesture-controlled robotic hand that can mimic human finger motion for medical and assistive applications is the project's long-term objective. However, a complete multi-finger robotic hand presents high-dimensional control complexity, substantial mechanical coupling, and sensing difficulties. This project focuses on a single robotic finger that is driven by a DC motor via a tendon-driven mechanism in order to maintain analytical depth and facilitate rigorous control design.

The robotic finger can be modeled as a single-input single-output (SISO) electromechanical system while still being representative of larger robotic hand architectures by limiting the system to a single degree of freedom. This method makes it possible to incorporate embedded implementation, simulation-based validation, controller design, and detailed modeling into a single academic project.

2. SYSTEM OVERVIEW

The suggested system combines vision-based perception, reference generation, motor control, mechanical actuation, and feedback sensing into a closed-loop, human-in-the-loop mechatronic control system.

2.1 Description

In front of a camera, the human operator makes finger movements. To estimate finger bending, these motions are recorded, processed, and then transformed into a reference motor angle. The robotic finger tracks the motion of the human finger in real time thanks to a feedback controller that drives the motor.

2.2 System Architecture

The system consists of the following subsystems:

- Vision-based gesture sensing
- Reference generation and signal conditioning
- Embedded motor control
- Mechanical actuation via tendon transmission
- Encoder-based feedback sensing

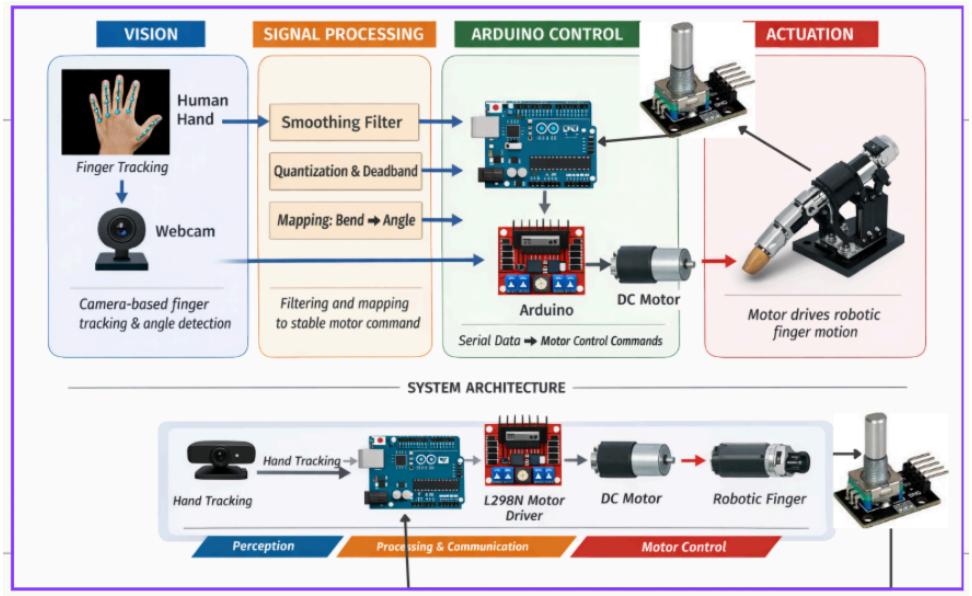

Figure 1. System Architecture

2.3 Signal Definitions

The key system signals are defined as:

$\theta_h(t)$: $Estimated\ Human\ Finger\ Bending\ Angle$

$\theta_r(t)$: $Reference\ Motor\ Angle\ Generated\ From\ Vision\ Processing$

$\theta_m(t)$: $Measured\ Motor\ Angle\ Obtained\ From\ Encoder\ Feedback$

$e(t) = \theta_r(t) - \theta_m(t)$: $Tracking\ Error$

3. HARDWARE ARCHITECTURE

3.1 Actuator

A 12 V DC motor is used to operate the robotic finger. A servo motor was considered as a potential actuator during early prototyping; however, the available servo could not produce enough torque to consistently overcome tendon friction and elastic restoring forces under load. Consequently, a DC motor was used in place of the actuator.

Through pulse-width modulation (PWM), the DC motor offers continuous rotational motion and flexible torque control. This arrangement allows for controlled finger bending while preserving hardware simplicity when paired with the tendon-driven transmission. Control flexibility and robustness are given priority in this actuator selection, and closed-loop encoder feedback is used to achieve positional accuracy.

Figure 2. DC Motor

3.2 Sensors

To determine how far the tendon has been stretched, a rotary encoder has been installed on the shaft of the spool, as well as to provide feedback to the closed-loop system, allowing the

motor's position to be controlled. The motor-side sensing method was chosen because it was simple and reliable, however, it also means that the finger's joint angles cannot be measured directly.

Figure 3. Rotary encoder used for motor-side angular position feedback

3.3 Power Electronics

A DC motor is connected to the Arduino microcontroller through an L298N H-bridge. By using pulse-width modulation (PWM), the speed of the motor can be controlled while the direction is controlled by using digital logic signals.

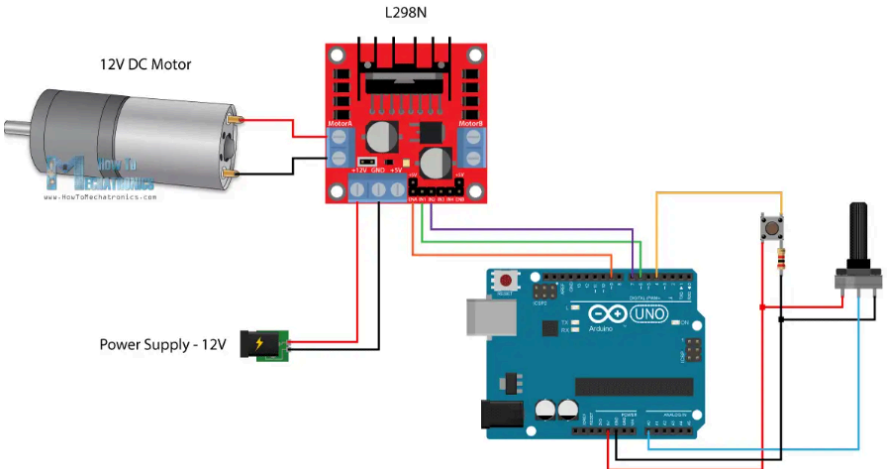

Figure 4. Electronics Systems

3.4 Mechanical Transmission

In order to convert the rotation of the motor into bending of the finger, a spool and tendon mechanism has been implemented, wherein the tendons move around and off of the spool, thereby causing a linear displacement of the finger, and the rubber bands provide a restoring force to extend the finger, allowing the finger to move in both directions with just one motor.

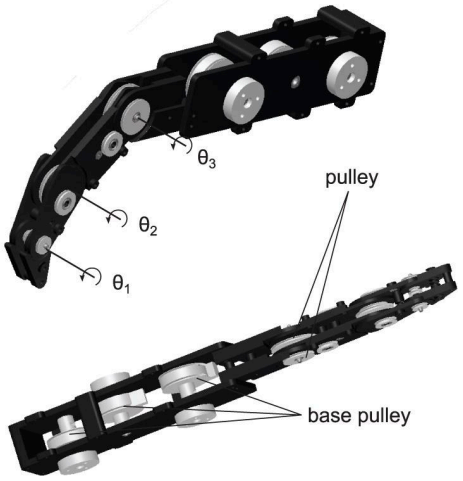

Figure 5. Mechanical Pulley

4. VISION PROCESSING AND GESTURE CONTROL

4.1 Vision Input Assumptions

A monocular webcam is used to capture real-time video of the human hand. The camera is assumed to be stationary with a clear field of view and sufficient lighting for reliable landmark detection.

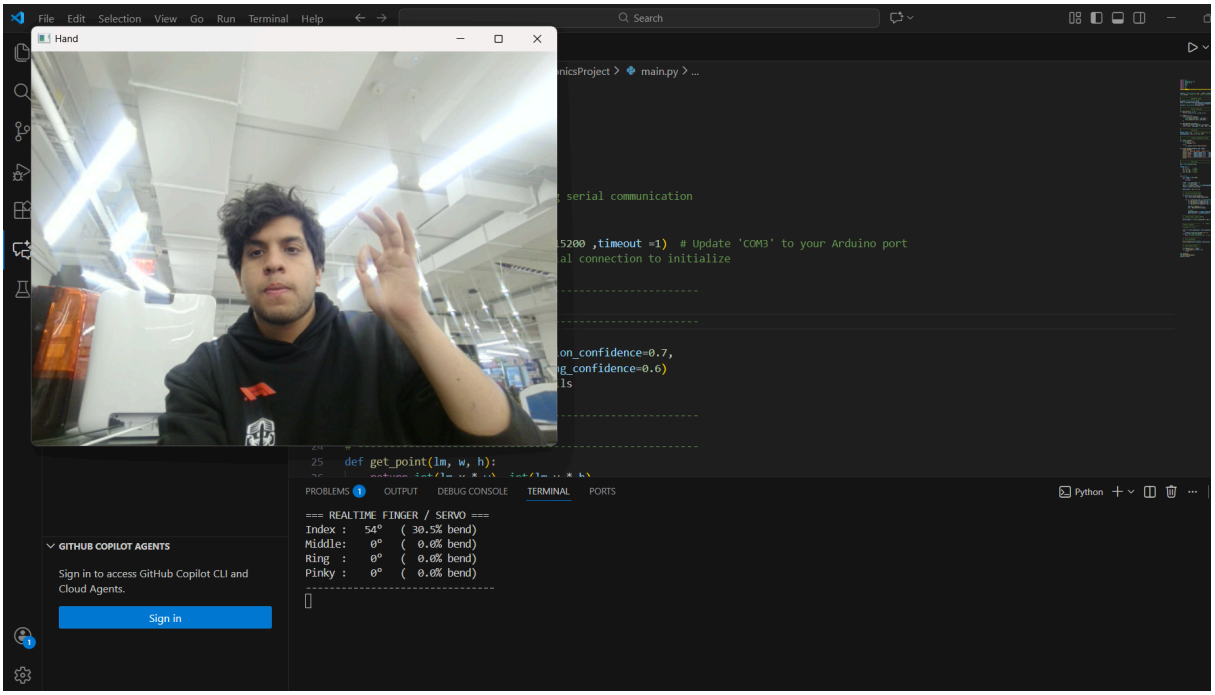

Figure 6. Camera

4.2 MediaPipe Hand Tracking

MediaPipe Hand Tracking is employed to detect twenty - one hand landmarks corresponding to finger joints and key anatomical features. These landmarks are expressed in normalized image coordinates and are updated at the camera frame rate.

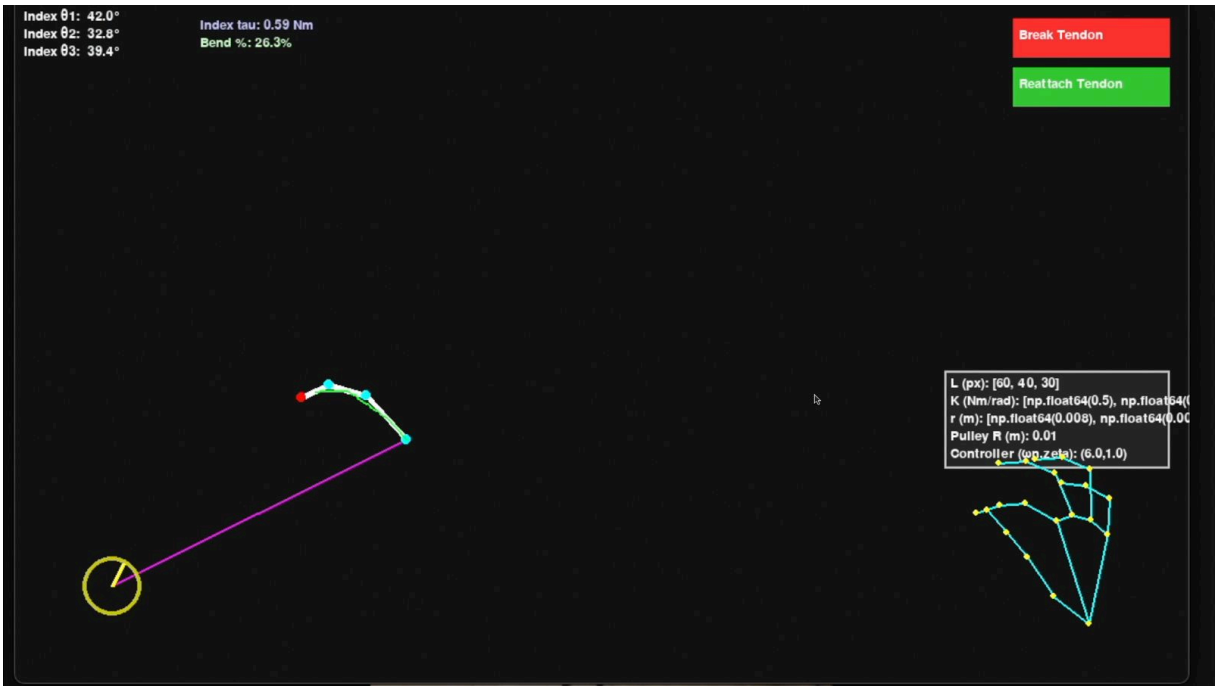

Figure 7. Media Pipe

4.3 Finger Angle Estimation

Finger bending is computed by forming vectors between adjacent landmarks and calculating the inter-segment angle using the dot product:

$$\theta_h = \cos^{-1}\left(\frac{\mathbf{v}_1 \cdot \mathbf{v}_2}{\|\mathbf{v}_1\|\,\|\mathbf{v}_2\|}\right)$$

This formulation provides a continuous and geometrically meaningful estimate of finger flexion.

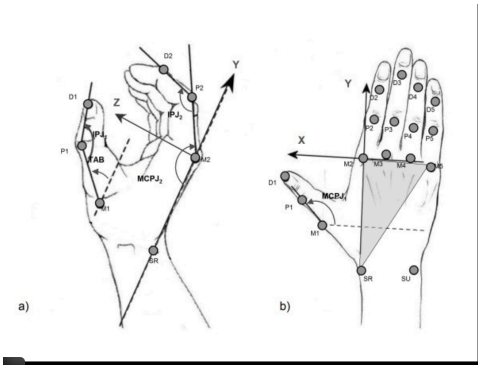

Figure 8. Finger Angle Estimation

4.4 Signal Conditioning

To reduce measurement noise and suppress high-frequency jitter in the vision-based finger angle estimate, the raw signal is processed through a low-pass filter and a deadband. The filtered signal is given by:

$$\theta_{h,f}(k) = \alpha\theta_h(k) + (1-\alpha)\theta_{h,f}(k-1)$$

where $\alpha \in (0,1)$ is the smoothing coefficient.

A deadband is applied to prevent small unintentional finger movements from generating actuator commands, improving closed-loop stability.

5. REFERENCE GENERATION AND MAPPING

The filtered human finger angle is mapped to a motor reference angle using a linear transformation:

$$\theta_r(t) = k_s\,\theta_{h,f}(t) + \theta_0$$

where:

- k_s is a scaling factor that accounts for the spool radius and tendon-driven transmission geometry, mapping human finger motion to motor-side angular motion, and

- θ_0 defines the neutral motor position corresponding to a relaxed finger posture.

Saturation limits are applied to the reference angle to ensure that commanded motor positions remain within safe mechanical and electrical operating bounds.

6. EMBEDDED FIRMWARE DESIGN

The embedded subsystem is responsible for real-time motor actuation and feedback acquisition. An Arduino microcontroller is used as the low-level controller, interfacing between the high-level vision-based reference generation and the physical actuator. The firmware receives reference motor angles from the host computer, measures motor position using a rotary encoder, and generates motor control signals through an H-bridge motor driver. The embedded software is designed to be lightweight and deterministic, ensuring reliable operation despite asynchronous serial communication and sensor noise.

The embedded firmware executes this control logic continuously in a real-time loop, updating motor commands at each iteration based on the latest reference and encoder feedback.

6.1 Firmware Responsibilities

During each control cycle, the Arduino firmware performs the following tasks:

• Receives the reference motor angle $\theta_r(t)$ via serial communication

• Acquires encoder measurements and computes the measured motor angle $\theta_m(t)$

• Computes the tracking error: $e(t) = \theta_r(t) - \theta_m(t)$

• Determines motor direction and PWM magnitude based on the tracking error

• Enforces safety constraints such as saturation limits and communication timeouts

This partitioning ensures that computationally intensive vision processing is handled by the host computer, while the embedded controller focuses exclusively on reliable low-level motor actuation.

6.2 Serial Communication Protocol

Reference angles are transmitted from the host computer to the Arduino using a UART serial interface operating at 115200 baud. Each reference command consists of a single numeric

angle value encoded in ASCII format and terminated by a newline character to simplify parsing and debugging.

Upon receiving a complete message, the firmware updates the current reference angle θr(t). To ensure safe operation in the event of communication loss, a command timeout mechanism is implemented. If no new reference command is received within a specified time interval, the motor command is set to zero and the actuator is disabled.

6.3 Encoder Processing and Position Estimation

A rotary encoder mounted on the motor shaft provides incremental position feedback. Encoder counts are accumulated in software and converted to an angular position using the encoder resolution:

$$\theta_m(t) = \frac{2\pi}{N_{cpr}} * N(t)$$

where N(t) is the accumulated encoder count and Ncpr is the number of counts per revolution. This conversion provides a continuous estimate of the motor shaft angle for closed-loop control. Optional debouncing and basic filtering are applied to reduce noise-induced miscounts.

6.4 Motor Driver Interface and Control Logic

The Arduino controls the DC motor through an H-bridge motor driver using digital direction pins and a PWM output pin. Motor rotation direction is determined based on the sign of the tracking error:

If e (t) > 0, the motor rotates in the positive direction.
If e (t) < 0, the motor rotates in the negative direction.

The PWM duty cycle is computed as a function of the error magnitude and is limited by a maximum allowable value to protect the motor and driver circuitry. A small deadband around zero error is implemented to prevent motor jitter and oscillatory behavior near the desired position.

6.5 Real-Time Execution and Safety Constraints

The firmware executes within a periodic loop to maintain consistent timing. Several safety constraints are enforced at the embedded level:

- PWM saturation to prevent excessive motor current and overheating.
- Reference angle bounds to prevent mechanical over-travel
- Command timeout to safely stop the motor if serial communication is lost
- Deadband near the setpoint to avoid chatter and audible oscillations

These constraints improve system robustness and ensure safe operation under noise, disturbances, and transient communication faults.

6.6 Embedded Control Logic Summary

The high-level embedded control logic can be summarized as follows:

- Read the latest reference angle from serial communication
- Measure the current motor angle using encoder feedback
- Compute the tracking error
- Determine motor direction and PWM command
- Apply saturation and safety constraints
- Command the motor driver

This firmware architecture provides a reliable interface between high-level vision-based reference generation and low-level motor actuation.

7. MATHEMATICAL MODEL

7.1 Electrical Model

The DC motor electrical dynamics are modeled as:

$$v(t) = L\frac{di(t)}{dt} + Ri(t) + K_e\omega(t)$$

$$J\frac{d\omega(t)}{dt} + b\,\omega(t) = K_t i(t)$$

Armature inductance is neglected due to its minimal effect at the operating frequencies of interest.

7.2 Mechanical Model

The mechanical dynamics are given by:

$$J\dot{\omega}(t) + \left(b + \frac{K_t K_e}{R}\right)\omega(t) = \frac{K_t}{R}v(t)$$

where J represents the combined inertia of the motor, spool, and finger.

7.3 Transfer Function

Combining the electrical and mechanical models yields a second-order transfer function with the use of Laplace Transform.

$$G(s) = \frac{\Theta(s)}{V(s)} = \frac{K}{s(\tau s + 1)}$$

where,

$$K = \frac{K_t}{R\left(b + \frac{K_t K_e}{R}\right)},$$

$$\tau = \frac{J}{b + \frac{K_t K_e}{R}}$$

8. CONTROL SYSTEM DESIGN

8.1 Design

A proportional-derivative (PD) controller is used to ensure stable tracking of the reference

angle:

$$C(s) = \frac{K_d s + K_p}{\tau s + 1}$$

where,

$K_d = 0.2, K_p = 4, \tau = 0.02$

The proportional term provides stiffness, while the derivative term adds damping to suppress oscillations caused by tendon compliance.

$$L(s) = \frac{K(K_p + K_d s)}{s(\tau s + 1)}$$

$$T(s) = \frac{K(K_p + K_d s)}{\tau s^2 + (1 + K K_d)s + K K_p}$$

$K_d = 0.2, K_p = 4, \tau = 0.02, K = 0.1$

Thus, the final equations that we get are as follows:

$$G(s) = \frac{1}{s(0.1s + 1)}$$

Figure 9. Plant Function

$$C(s) = K_p + K_d s = 4 + 0.2s$$

Figure 10. Controller Equation

$$L(s) = \frac{4 + 0.2s}{s(0.1s + 1)}$$

Figure 11. Open Loop Transfer Function

$$T(s) = \frac{0.2s + 4}{0.1s^2 + 1.2s + 4}$$

Figure 12. Closed Loop Transfer Function

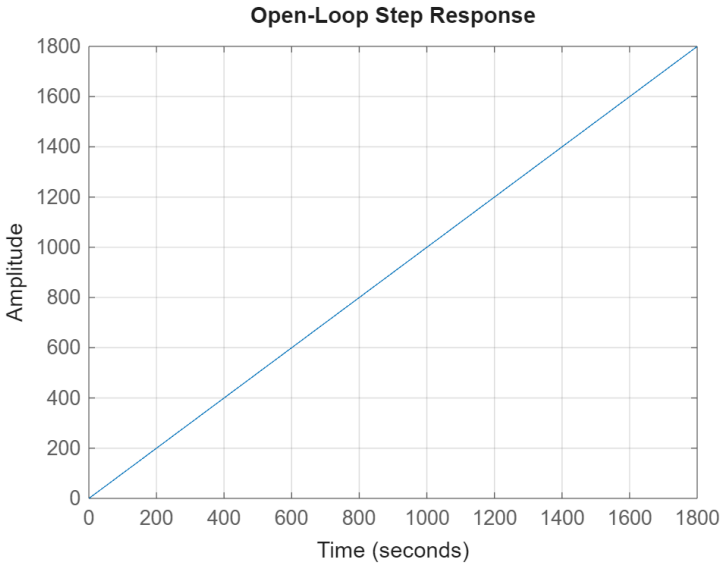

Figure 13. Open Loop Step Response

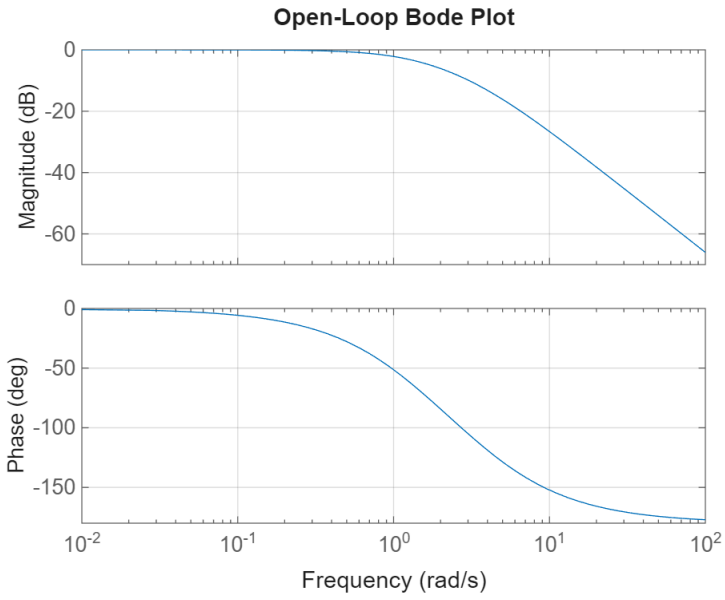

Figure 14. Open Loop Bode Plot

The output in the open-loop case is the response without feedback, implying that the controller cannot adjust to remove any errors or disturbances in the control output. The step response of the open-loop system reveals a continuously increasing output, which is expected to settle to a certain value. However, this settling value in the DC motor model is infinity because the plant transfer function has the integrator 1/s. The resulting output means that the motor is physically incapable of stopping at any angle once the voltage is applied to it because the output position will continue to increase. The Bode plot of the open-loop system further

confirms the instability because the magnitude falls off at lower frequencies and the phase approaches - 180 degrees.

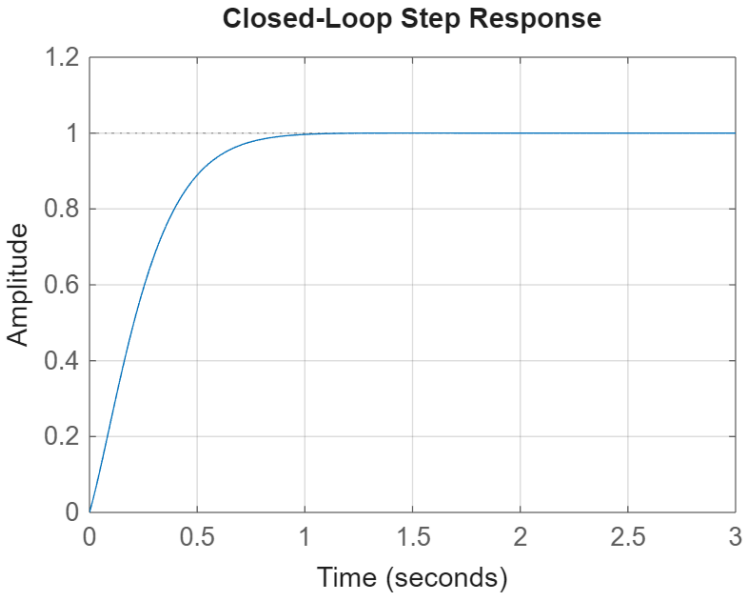

Figure 15. Closed Loop Step Response

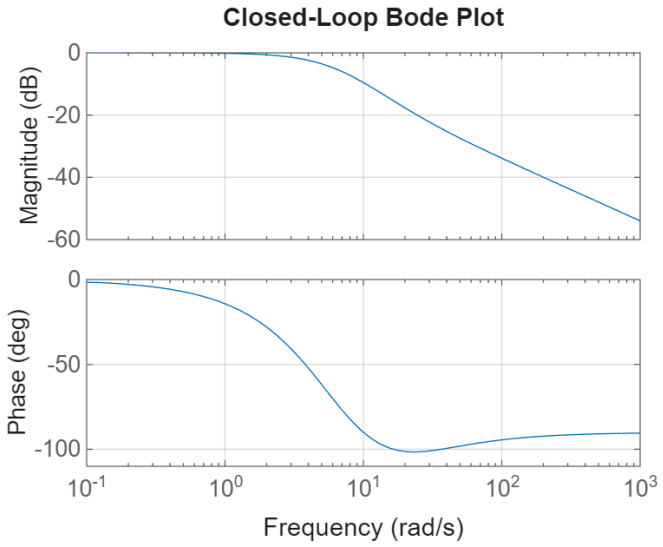

Figure 16. Closed Loop Bode Plot

When the feedback from the PD controller and encoder is added into the system, the latter enters a closed-loop mode where the motor position is properly controlled and maintained. The closed-loop step response plot clearly shows that the system behaves in a stable and damped manner, with the output converging promptly to the desired position without any oscillations and overshooting. This confirms that the proportional component provides

stiffness and facilitates high-speed dynamics, whereas the derivative component adds damping properties and increases stability. The Bode plot analysis of the closed-loop system also supports these observations. The magnitude plot reveals a smooth roll-off characteristic at high frequencies, signifying an attenuation of noise and disturbances. On the other hand, the phase plot exhibits less steep tendencies without precipitous falls, signifying improvements in phase margin and stability. The closed-loop analysis clearly supports the fact that the controller effectively stabilizes the motor position and facilitates precise positioning and reliable functioning of the robotic finger amidst disturbances.

8.2 Simulation and Validation

The complete system is simulated in MATLAB/Simulink, including the plant model, controller, encoder feedback, and disturbances.

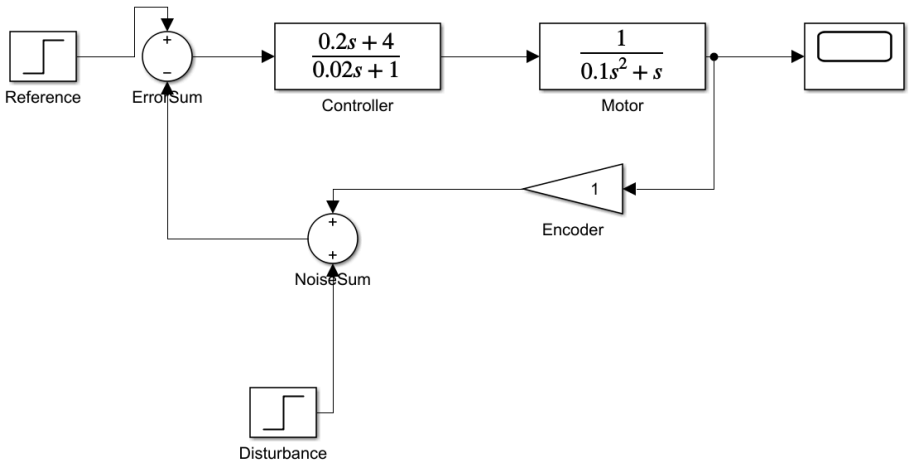

Figure 17. Simulink Block Diagram

This block diagram describes the closed loop position control system of the robotic finger. The reference input is the desired position of the finger, which is calculated using the vision-based gesture recognition system. The difference, also called the error, between the reference and the physical position, after being filtered by a filtered PD controller, provides the control signal for the DC motor modelled using a second-order system based on the equations of the DC motor. The system includes feedback from the encoder and disturbance signals for robustness testing.

9. MECHANICAL DESIGN

9.1 Introduction

The mechanical design was also an essential part of developing the vision-based gesture-controlled robotic finger system. The system required attention to mechanical stability as it utilized tendon actuation as well as elastic return mechanisms. The mechanical components were designed using Fusion 360 software, whose main objective was to create rapid prototyping using 3D printing. This was more cost-effective than other methods when this project was carried out.

CAD modeling involved a number of activities such as model validation for mechanical feasibility, iterative improvements, and accurate fabrication file generation. The systems that were modeled and designed for the mechanical aspect include the finger mechanism, the mount and actuation system, as well as the spool tendon transmission system. These systems had a number of iterations based on mechanical failure.

9.2 Finger Mechanism Design

9.2.1 Design Significance

The finger mechanism is responsible for converting tendon displacement into visible bending motion and is the primary interface between the actuation system and the robotic output. Its design directly influences motion reliability, repeatability, and resistance to unwanted deformation under elastic loading.

9.2.4 Design Modifications

The finger mechanism was redesigned using a direct tendon-driven architecture, significantly simplifying the mechanical structure. However, further issues emerged during testing:

1. Lack of Passive Finger Extension: Initially, the finger did not return to an upright position when tendon tension was released. Rubber band hooks were incorporated into the CAD model, allowing elastic elements to provide passive extension.
2. Backward Joint Bending Under Elastic Load: The rubber bands caused the finger joints to bend backward beyond their intended range of motion. Physical joint guards

were added to each finger segment to mechanically limit backward rotation. This ensured that, at zero tendon tension, the finger remained upright rather than collapsing backward.

3. Tendon Routing and Stability: Unguided tendons led to inconsistent motion and occasional slippage. Holes were added along the finger body to guide tendon routing, and a groove at the fingertip was designed to securely anchor the tendon.

These design changes resulted in a stable finger mechanism capable of repeatable bending and extension.

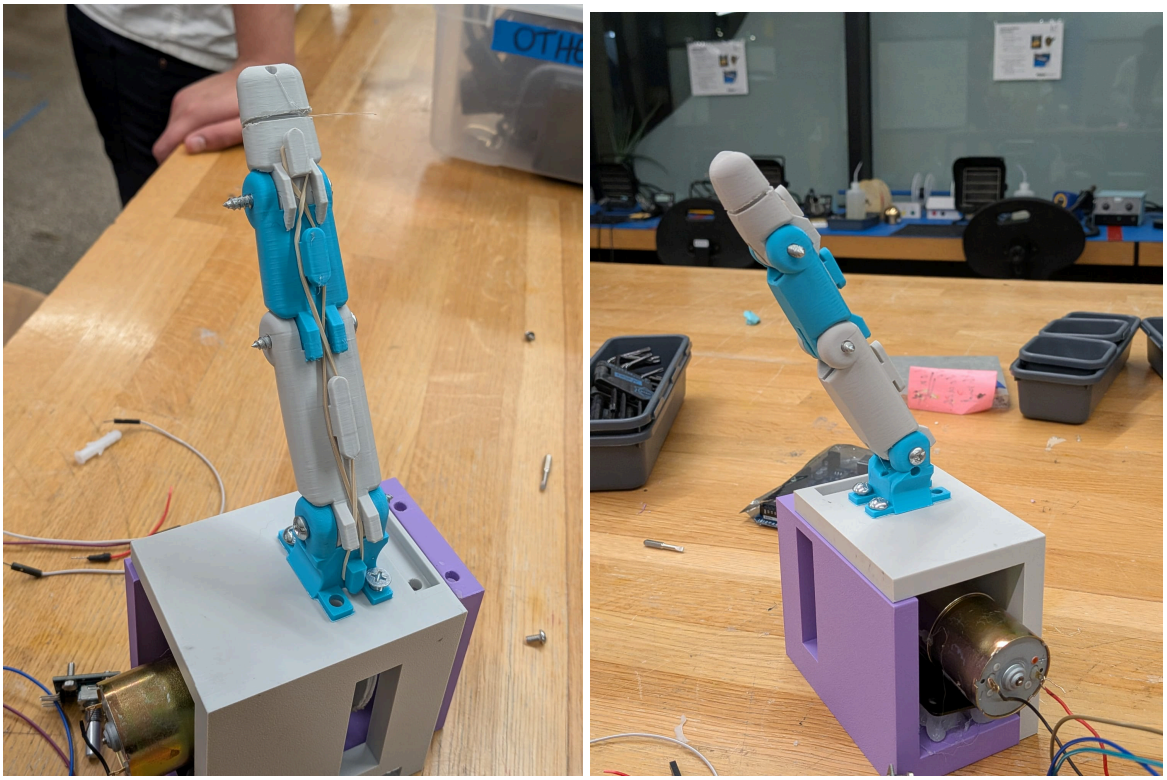

Figure 19. Finger Design

9.3. Motor Mount and Actuation System

9.3.1 Design Significance

The motor mount ensures rigid alignment between the actuator and the tendon transmission system. Proper alignment is essential for efficient torque transfer and consistent finger motion.

9.3.2 DC Motor Integration and Encoder Placement

The system was redesigned to use a DC motor, providing higher torque at the cost of losing inherent positional feedback. To compensate for this, a rotary encoder was introduced.

Rather than mounting the encoder directly on the motor shaft, it was positioned on the opposite side of the spool, ensuring that encoder rotation directly corresponded to tendon displacement. The motor mount was redesigned to securely hold the motor, spool, and encoder in coaxial alignment while remaining compact and 3D printable.

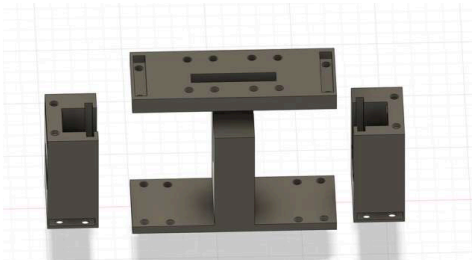

Figure 20. Finger and Robot Mount

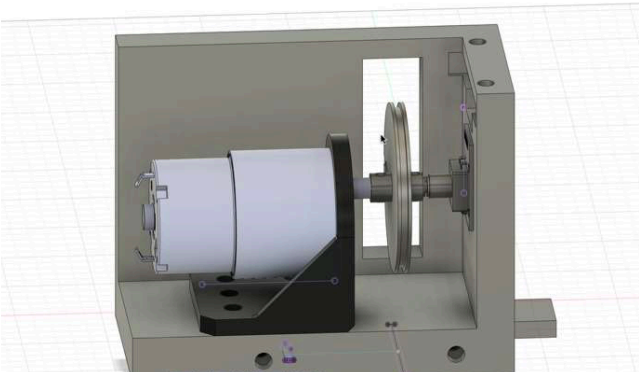

Figure 21. Motor and Spool Assembly

Figure 22. Final Assembly

9.4. Spool and Tendon Transmission System

9.4.1 Design Significance

The spool converts motor rotation into linear tendon displacement. Its reliability directly affects finger positioning accuracy and system robustness.

9.4.2 Spool Design

A commercial servo attachment was temporarily used to improve coupling; however, this solution became obsolete after transitioning to DC motors.

The final spool design included two axial ports—one for the motor shaft and one for the encoder shaft—secured using set screws. Despite this improvement, PLA deformation under torque still caused slippage. As a time-constrained solution, hot glue was applied between the shafts and the spool, providing temporary adhesion. While effective in the short term, this highlighted the need for metal or reinforced spool designs in future iterations. Additional features such as tendon guards and tie holes were added to improve reliability.

Figure 23. Dual- Shaft Spool with Tendon Guards

10. LIMITATIONS

The current system emphasizes controlling one joint of the robotic finger and does not attempt to directly measure the joint angles of the finger. Instead, motor shaft position as interpreted by its encoder is utilized as an estimate, which has associated models involving tendon compliance, friction, and backlash. While in reality, all such values could be nonlinear, their models in this system framework are kept simple and are not explicitly compensated for in the

controller. The vision system is also indirect in terms of its measurements, giving only relative gestures and not absolute measurements of the human finger. Yet, it does manage to convey stable and effective control of gestures for a proof-of-concept setup.

11. FUTURE WORK

Future developments of this technology could be a multi-finger, multi-_degree-of-freedom robotic hand. Adding sensors at a joint level within the finger would increase precision and allow closed-loop control at a joint-by joint level rather than at a motor position. Sensors measuring either forces/torques could allow a compliant grasping strategy. Sensor fusion algorithms integrating vision, encoder, and force sensors could greatly enhance performance in real-world environments. More sophisticated control algorithms, like impedance control, adaptive control, and learning controllers, could offer further improvements in environments with uncertainty and non-linearities.

12. CONCLUSION

This project demonstrates a rigorously engineered vision-based gesture-controlled robotic finger that integrates perception, system modeling, control design, embedded firmware, and mechanical implementation into a unified mechatronic system. A physically motivated DC motor model was derived, and a PD controller was designed and validated through analytical modeling and Simulink simulations before hardware implementation. The system successfully translates human finger gestures into stable robotic motion while rejecting disturbances and sensor noise. As a reduced-order prototype, this work provides a strong foundation for future robotic hand systems aimed at assistive, rehabilitative, or medical applications.

REFERENCES

1. R. Cutkosky, *"On Grasp Choice, Grasp Models, and the Design of Hands for Manufacturing Tasks,"* IEEE Transactions on Robotics and Automation, vol. 5, no. 3, pp. 269–279, 1989.
2. S. Schaal, *"Is Imitation Learning the Route to Humanoid Robots?"* Trends in Cognitive Sciences, vol. 3, no. 6, pp. 233–242, 1999.
3. M. Tavakoli, A. Aziminejad, R. Patel, and M. Moallem, *"High-Fidelity Bilateral Teleoperation Systems and the Effect of Multimodal Haptics,"* IEEE Transactions on

Systems, Man, and Cybernetics, vol. 37, no. 6, pp. 1512–1528, 2007.

4. T. B. Sheridan, *"Human–Robot Interaction: Status and Challenges,"* Human Factors, vol. 58, no. 4, pp. 525–532, 2016.
5. K. Ogata, *Modern Control Engineering*, 5th ed., Prentice Hall, 2010.
6. B. Siciliano and O. Khatib, *Springer Handbook of Robotics*, Springer, 2016.