

UNIMORE
UNIVERSITÀ DEGLI STUDI DI
MODENA E REGGIO EMILIA

UNIVERSITY OF MODENA AND REGGIO EMILIA

Department of Sciences and Method for Engineering

Master's Degree in Mechatronic Engineering

Design and Sim-to-Real Learning of a Tactile and ToF-Sensorized Two-Finger Gripper: From Embedded Admittance Control to Reinforcement-Learned Grasping

Progettazione e apprendimento Sim-to-Real di un gripper a due dita dotato di sensori tattili e ToF: dal controllo di ammettenza embedded alla presa con Reinforcement Learning

Supervisor:
Prof. Fabrizio Pancaldi
Co-supervisors:
Dr. Luca Dibiase
Reviewer:
Prof. Alessandro Bertacchini

Candidate:
Nicola Montanari
Student ID 206955

ACADEMIC YEAR 2025/2026

Abstract

Design and Sim-to-Real Learning of a Tactile and ToF-Sensorized Two-Finger Gripper: From Embedded Admittance Control to Reinforcement-Learned Grasping

This thesis presents the design, implementation, and preliminary validation of an end-to-end mechatronic and learning pipeline for a sensorized two-finger robotic gripper. Robust autonomous manipulation across varied object geometries and mechanical properties remains a fundamental challenge in robotics. While classical control methods struggle to generalize across diverse object profiles, learned grasping strategies often rely on idealized simulation environments that fail to mirror the physical realities of real-world tactile sensing. To bridge this gap, this work establishes a hardware and simulation platform where classical and learning-based control strategies can be realized, compared, and deployed using identical sensor modalities.

The physical platform comprises a custom two-finger parallel gripper equipped with a dual Force-Sensing Resistor (FSR) matrix for tactile feedback and a multi-zone Time-of-Flight (ToF) sensor (VL53L7CX) for pre-grasp spatial localization. Low-level processing and control are executed on an STM32F446 microcontroller, integrating a real-time admittance control loop to enable compliant grasping behavior. High-throughput multi-sensor telemetry and motor commands are structured via a CANopen communication, decoupling local real-time control constraints from off-board logging. In parallel, a dedicated electronic hardware architecture was designed and fabricated as a custom PCB to consolidate sensing, power regulation, and signal processing into the gripper body.

To complement the physical hardware, a sensor-faithful virtual model was developed in NVIDIA Isaac Sim. The virtual environment models the structural kinematics of the end effector while replicating the response characteristics of both the tactile FSR matrices and the multi-zone ToF distance sensors. Utilizing this simulated framework, a reinforcement learning (RL) policy was formulated and trained in NVIDIA Isaac Lab. The policy operates over combined tactile and proximity observation spaces to learn compliant grasping behaviors.

The primary contributions of this work are fourfold:

- Mechatronic Co-Design: Mechanical, electronic, and firmware co-design of a two-finger sensorized end effector featuring onboard real-time tactile and ToF processing.

- Real-Time Communication Architecture: A deterministic CANopen communication framework balancing real-time motor control with high-rate sensor streaming.
- Embedded Compliant Control: An embedded admittance control implementation evaluated on prototype hardware to demonstrate compliant grasping dynamics.
- Sensor-Faithful Virtual Model and RL Pipeline: Construction of a sensor-matched virtual model in Isaac Sim along with an RL policy training pipeline in Isaac Lab.

Preliminary evaluations confirm real-time closed-loop admittance performance and functional sensor streaming on the prototype platform, alongside stable simulation dynamics during RL training. A structured roadmap outlines immediate follow-up milestones, including hardware bring-up of the custom PCB, quantitative admittance benchmarking, converged policy optimization, and eventual transfer to the physical system.

Keywords: PPO, RL, AI, Embedded Systems, End Effector

Italian Summary

Il presente lavoro tesi tratta la progettazione, la realizzazione e la validazione preliminare di una piattaforma integrata, per un gripper robotico a due dita dotato di sensori tattili e di prossimità. Il gripper integra una doppia matrice di sensori di forza resistivi (FSR) e un sensore *Time-of-Flight* (ToF) multi-zona. Un microcontrollore STM32 esegue a bordo un controllo di ammettenza in tempo reale, mentre la comunicazione con il rover avviene su bus CANopen. Parallelamente, un modello virtuale fedele ai sensori è stato costruito in NVIDIA Isaac Sim e usato in Isaac Lab per addestrare una policy di reinforcement learning di pick and place. L'obiettivo è poter realizzare, confrontare e rendere trasferibili su hardware fisico strategie di controllo classiche e apprese sulla stessa piattaforma, con gli stessi sensori.

Contesto, motivazioni e obiettivi

Il lavoro trae origine da un'esigenza operativa concreta di ProjectRED, il team studentesco del Dipartimento di Scienze e Metodi dell'Ingegneria (DISMI) dell'Università di Modena e Reggio Emilia, che progetta e costruisce un rover per l'esplorazione planetaria destinato alla partecipazione alla European Rover Challenge (ERC). L'ERC è una competizione annuale tra squadre universitarie, nella quale i rover svolgono, su un terreno artificiale che riproduce la superficie marziana (il *Mars Yard*), una serie di compiti ispirati a scenari reali di missioni spaziali: esplorazione scientifica, navigazione, manutenzione, raccolta di campioni.

Due di questi compiti sono direttamente rilevanti per la tesi, poichè richiedono al rover una manipolazione di fino e contact rich tramite il braccio robotico e il relativo end effector. Nella *Maintenance Task*, il rover deve azionare un pannello progettato per l'uso umano, commutando leve e selettori rotativi, inserendo una spina in una presa e posizionando una piastra metallica su un elettromagnete, senza alcun intervento fisico da parte del team. Diversi elementi del pannello sono sensibili a forze e coppie eccessive e non devono essere danneggiati durante l'operazione. Nella *Probing Task*, il rover deve individuare tre sonde cilindriche disperse sul terreno, afferrarle e trasportarle a bordo nel minor tempo possibile, poichè il punteggio dipende sia dal successo

della raccolta sia dal tempo impiegato. In entrambe le prove pesa anche il livello di automazione raggiunto. Il gripper sviluppato è l'end effector che il team intende impiegare in queste due prove. La parte di apprendimento è pensata soprattutto per la probing task, dove servono presa rapida e affidabile con un budget di tempo stretto.

Da tali compiti discende la motivazione scientifica del lavoro. La presa robusta di oggetti eterogenei per rigidezza, geometria e attrito resta un problema aperto in robotica, infatti, il solo feedback visivo risulta insufficiente al momento del contatto, poichè le dita stesse occludono l'oggetto e fenomeni come deformazione o scivolamento non sono così osservabili a distanza. Il feedback tattile fornisce invece informazione locale di forza proprio nella regione in cui essa è necessaria. Il proximetry sensing mediante ToF risulta complementare al tatto, poichè fornisce informazione geometrica prima del contatto (presenza dell'oggetto, distanza, forma approssimativa), laddove il tatto fornisce informazione durante il contatto stesso (distribuzione della pressione, forza applicata, scivolamento incipiente). Infine, un controllore di ammettenza a guadagni fissi richiede una ritaratura per ciascuna classe di oggetti e generalizza limitatamente, mentre una politica addestrata tramite RL può, in linea di principio, apprendere una strategia di presa più generale, a condizione che i sensori simulati riproducano con sufficiente fedeltà quelli reali. Il divario tra simulazione e realtà (*reality gap*) risulta infatti particolarmente marcato per il sensing tattile rispetto, ad esempio, alla lettura di altri sensori.

Da queste considerazioni discendono le tre domande di ricerca che hanno guidato lo sviluppo del progetto.

- un insieme di sensori FSR a matrice e ToF a basso costo, elaborato localmente su un microcontrollore STM32F446, può supportare un ciclo di controllo di ammettenza in tempo reale adeguato alla presa stabile di oggetti con rigidezza eterogenea?
- un modello virtuale può riprodurre le medesime modalità sensoriali con fedeltà sufficiente ad addestrare una politica di RL trasferibile al gripper fisico?
- come si comporta una politica appresa rispetto al controllore classico di ammettenza in termini di tasso di successo, generalizzazione a oggetti non precedentemente osservati e robustezza alle perturbazioni?

Quest'ultima domanda costituisce il fulcro del piano di ricerca di lungo periodo, sebbene il confronto diretto tra i due paradigmi esuli dal lavoro qui presentato.

Stato dell'arte

In riferimento ai sensori di forza applicati in questo contesto, ci sono fondamentalmente due principali categorie di tecnologie. Le matrici resistive, come gli FSR (Figura 2.1) che usate in questo progetto, sono convenienti, sottili e facili da leggere. Infatti, per un singolo taxel basta un partitore di tensione e un canale ADC. Tuttavia, presentano alcuni problemi come l'isteresi, una bassa ripetibilità, la deriva nel tempo e variazioni tra diversi esemplari, il che significa che richiedono calibrazioni individuali. D'altra parte, sono disponibili soluzioni ottiche o multimodali come GelSight, ReSkin e BioTac, che offrono informazioni molto più dettagliate. GelSight (Figura 2.2), per esempio, riesce a ricostruire la geometria del contatto con un'ottima risoluzione spaziale grazie all'osservazione della deformazione di un elastomero, anche se questo comporta più ingombro e una maggiore richiesta computazionale. ReSkin (Figura 2.3a) utilizza una pelle elastomerica sostituibile letta in modo magnetico, mentre BioTac (Figura 2.4) è un polpastrello biomimetico pieno di liquido, capace di misurare forze normali, forze di taglio, vibrazioni e flusso termico, anche se il costo, le dimensioni e la manutenzione non lo rendono ideale per un gripper. Anche nel campo industriale, il sensing tattile a matrice è ormai considerato uno standard, come dimostrano i polpastrelli Robotiq TSF-85 (Figura 2.5), che hanno una matrice capacitiva con ventotto elementi. La scelta adottata in questa tesi, ossia una matrice FSR elaborata localmente in tempo reale, privilegia deliberatamente l'integrabilità, il costo contenuto e la bassa latenza dell'anello di controllo rispetto alla fedeltà della misura.

Riguardo al proximity sensing, la letteratura sul cosiddetto *pretouch* evidenzia come un sensore non a contatto montato sull'end effector colmi il divario tra la visione a distanza, soggetta ad occlusioni, e il tatto, disponibile solo a contatto avvenuto. I sensori ToF multizona, quali il VL53L7CX impiegato in questo lavoro (Figura 2.6), restituiscono una griglia di distanze indipendenti su un campo di vista di sessanta gradi per sessanta, con l'elaborazione della misura eseguita interamente a bordo del sensore, risultando pertanto significativamente più leggeri, sia in termini di ingombro sia di carico computazionale, rispetto a una pipeline RGB-D completa, la cui accuratezza in profondità

peraltro si degrada proprio alle distanze ravvicinate rilevanti per la fase di avvicinamento.

Sul versante del controllo, i paradigmi di ammettenza e impedenza derivano dalla formulazione classica di Hogan sul controllo dell'interazione tra robot e ambiente. Il controllo di impedenza determina la forza da applicare in funzione del moto, mentre il controllo di ammettenza opera in modo inverso, determinando il moto in funzione della forza misurata, ed è pertanto più adatto ad attuatori controllati in posizione come quello impiegato in questa tesi. In entrambi i casi la dinamica desiderata è modellata come un sistema virtuale massa-molla-smorzatore. Il limite intrinseco di tale approccio risiede nella scarsa generalizzazione dei guadagni fissi tra oggetti con proprietà meccaniche differenti. Generalmente strategie di *gain scheduling*, quali la classificazione della rigidezza dell'oggetto seguita dalla selezione dei guadagni corrispondenti, attenuano ma non risolvono il problema, motivando l'interesse verso approcci di apprendimento automatico.

Tra i vantaggi di questo approccio, si annovera la capacità della policy di apprendere direttamente dall'esperienza maturata mappando i dati sensoriali in azioni efficaci, senza richiedere uno stimatore di stato progettato manualmente per ciascuna modalità sensoriale. Tuttavia sono però presenti limiti ben noti. Richiede moltissime interazioni, più di quante si possano raccogliere su hardware reale, e per questo serve una simulazione parallela su GPU. La progettazione della ricompensa è delicata, perché termini mal bilanciati producono politiche miopi. Infine la politica resta legata a compito, ricompensa e hardware di addestramento. Per questo l'apprendimento non è presentato come un semplice miglioramento del controllo classico, ma come un approccio diverso, con compromessi propri.

3. Progetto concettuale

Gli oggetti da manipolare sono di dimensioni contenute, per cui il requisito più stringente non riguarda la capacità di carico, bensì la regolazione della forza. Il gripper deve poter applicare una forza minima e ben controllata, al fine di non danneggiare il pannello della Maintenance Task, pur disponendo di forza sufficiente ad afferrare e sollevare in sicurezza una probe (Figura 3.1). La corsa delle dita deve coprire l'intero intervallo dimensionale rilevante, dalla sezione più stretta della probe (circa venti millimetri) fino all'elemento più largo del pannello, con margine sufficiente a compensare le imprecisioni di posizionamento del braccio. L'attuazione deve garantire precisione

e ripetibilità nel controllo di posizione, essere sufficientemente retroazionabile da sostenere un comportamento cedevole, e rispettare i vincoli di ingombro, massa e potenza propri di un end effector. Il sensore FSR deve essere posizionato direttamente sulla superficie di contatto, protetto da uno strato sottile che non ne alteri il segnale, mentre il sensore ToF richiede un campo visivo privo di ostruzioni. Infine, poichè il controllo è realizzato in ammettenza, la trasmissione meccanica non deve introdurre un elemento rigido non modellato. Giochi, attriti e cedevolezza strutturale devono essere contenuti o, in alternativa, caratterizzati con precisione.

Il diagramma a blocchi in Figura 3.2 descrive l'architettura elettronica complessiva. Il rover fornisce una tensione a 24 V al driver del motore passo-passo, il quale rende disponibile un'uscita secondaria a 12 V da cui è alimentata l'intera elettronica del gripper. Il microcontrollore comunica con il sensore ToF tramite interfaccia I^2C, dispone di un'interfaccia di programmazione SWD e integra il controllore CAN, collegato tramite un transceiver al bus condiviso con il controllore del rover. La catena di lettura tattile è composta da due matrici FSR, due multiplexer per la selezione di righe e colonne, un partitore di tensione, uno stadio di buffering e una protezione dell'ingresso analogico-digitale.

A livello concettuale elettronico, ciascuna matrice FSR è strutturata come una griglia in cui ogni intersezione riga-colonna costituisce un taxel. La lettura simultanea di tutti i taxel richiederebbe un percorso analogico dedicato, e idealmente un canale ADC, per ciascuno di essi. Questa risulta essere una soluzione non praticabile su un microcontrollore con un numero limitato di canali, si adotta pertanto uno schema di scansione a matrice, nel quale due multiplexer selezionano sequenzialmente una riga e una colonna, mentre la resistenza del taxel indirizzato viene convertita in tensione da un partitore, condizionata da buffer e protezione, e infine campionata dall'ADC. A livello meccanico, la rotazione del motore è trasformata in moto lineare delle dita mediante un meccanismo pignone-cremagliera. Un pignone calettato sull'albero motore ingrana con due cremagliere integrate nelle dita, che si aprono e si chiudono simmetricamente, riducendo il sistema a un unico grado di libertà complessivo. Il sensore ToF è collocato sulla scocca, in posizione centrale tra le dita, così da disporre di un campo visivo simmetrico e privo di ostruzioni sul volume di avvicinamento. Per mantenere il sistema compatto, la scheda elettronica è concepita come modulo aggiuntivo direttamente interfacciato al driver commerciale del motore, dal quale preleva sia l'alimentazione sia il bus CAN, evitando cablaggi separati. L'intero sistema è concepito per esercitare circa due 2 Kg

di forza sulle dita, ed è progettato in modo tale da consentire, in una fase successiva, l'inserimento di un riduttore sul motore senza la necessità di modifiche sostanziali.

4. Progettazione meccanica di dettaglio

I componenti strutturali del gripper, ossia la scocca principale e le due coperture frontali che racchiudono il meccanismo delle dita, sono realizzati mediante stampa 3D con tecnologia FDM in PLA. Tale scelta risponde al compromesso tra rigidezza, peso e producibilità ricercato. Il PLA risulta economico, dimensionalmente stabile e adatto alla realizzazione di superfici organiche e non prismatiche quali quelle della scocca, consentendo al contempo un'iterazione rapida del progetto durante la fase di sviluppo. L'unica eccezione strutturale è costituita dalla piastra centrale, realizzata in fibra di carbonio. Trattandosi dell'interfaccia portante tra motore, trasmissione pignone-cremagliera e scocca, e risultando serrata tra i componenti stampati, tale piastra è soggetta a sollecitazioni più elevate e concentrate, in particolare in prossimità dell'albero motore e dei punti di fissaggio delle guide lineari. Il maggiore rapporto rigidezza-peso della fibra di carbonio ne previene la flessione, preservando l'allineamento del meccanismo.

L'assemblaggio è costituito da tre gusci stampati e dalla piastra sopra descritta (Figura 4.1). La scocca principale ospita, sul retro l'alloggiamento del motore, mentre le due coperture frontali si fissano su di essa serrando la piastra in fibra di carbonio interposta. Le coperture definiscono altresì i canali di guida del moto lineare delle dita, proteggendo il meccanismo interno da polvere e contatti accidentali. Una finestra frontale ospita il sensore ToF, garantendone il campo visivo libero anche a dita completamente aperte. La medesima piastra assolve inoltre una funzione non strutturale, integrando i canali per il passaggio dei cavi dei sensori FSR di ciascun dito e del sensore ToF, che vengono così mantenuti separati dal meccanismo in movimento e convogliati verso un unico punto di uscita in prossimità del motore, dove è collocata la scheda elettronica. Sul retro del gripper è integrato l'attacco del sistema di sgancio rapido condiviso con il braccio robotico, che consente il montaggio e lo smontaggio dell'end-effector senza l'ausilio di utensili.

La trasmissione del moto rappresenta l'elemento meccanico centrale del sistema. Come mostrato in Figura 4.4, il pignone calettato sull'albero motore ingrana simultaneamente con due cremagliere disposte su lati opposti, ciascuna montata su un pattino scorrevole lungo una guida lin-

eare. L'impiego di una guida lineare dedicata per ciascuna cremagliera, anzichè affidare il vincolo della traiettoria al solo ingranamento, evita che i carichi laterali e fuori asse gravino sui denti degli ingranaggi, mantenendo l'ingranamento pulito e ripetibile e minimizzando il gioco e la cedevolezza che altrimenti distorcerebbero la relazione tra il moto del motore e il moto del polpastrello sotto il controllo di ammettenza. Ciascun dito è fissato rigidamente al pattino della propria cremagliera. Infine, l'accesso all'albero del pignone è lasciato deliberatamente disponibile, in modo da poter inserire in futuro un riduttore tra motore e pignone senza modificare piastra, cremagliere, pattini o dita.

5. Progettazione elettronica di dettaglio

L'attuatore di chiusura delle dita è un motore passo-passo NEMA 17 con driver integrato Auxind FD1.1GA (Figura 5.1). Tale combinazione motore-driver è già adottata su tutto il braccio robotico di ProjectRED, e la sua scelta per il gripper discende direttamente da tale circostanza. Il riutilizzo di un attuatore già validato riduce il numero di componenti distinti da approvvigionare e caratterizzare, e mantiene identica l'interfaccia di attuazione, in termini di alimentazione, protocollo di comunicazione e insieme di comandi, rispetto agli altri giunti del braccio. Il driver integra controllore, stadio di potenza ed encoder magnetico a 12 bit direttamente sul motore, comunica in CANopen a velocità comprese tra 10 KHz e 1 MHz, accetta un'alimentazione compresa tra 24 e 80 V in corrente continua ed espone un'uscita secondaria a $12V$, dalla quale viene prelevata l'alimentazione per l'elettronica del gripper. Il firmware del driver implementa un controllo di coppia in anello chiuso basato sulla lettura dell'encoder, modulando automaticamente la corrente in funzione del carico e riducendo la dissipazione termica nelle fasi a vuoto, pur consentendo un incremento di coppia fino al valore massimo configurato in presenza di resistenza.

La conversione di potenza avviene in due stadi successivi mediante regolatori lineari a bassa caduta (LDO) della famiglia AMS1117, scelti per la loro semplicità e reperibilità (Figura 5.4). Il primo stadio converte i 12 V in 5 V, il secondo converte i 5 V in 3.3 V. Entrambi gli stadi operano con un margine ampiamente superiore alla caduta di tensione minima del regolatore, e integrano protezioni contro il sovraccarico e il surriscaldamento. Trattandosi di regolatori lineari, la tensione dissipata in ciascuno stadio si converte interamente in calore, compromesso accettabile data la

corrente contenuta assorbita dall'elettronica del gripper, ma che privilegia la semplicità circuitale a scapito del rendimento di conversione.

Il sensore di prossimità impiegato è lo STMicroelectronics VL53L7CX, basato sulla tecnologia FlightSense. Un emettitore laser infrarosso a 945 nm, proiettato attraverso una lente, illumina un campo visivo quadrato di 60°×60°, suddiviso in una griglia di 64 zone di misura indipendenti, ciascuna delle quali restituisce la distanza assoluta dal bersaglio più vicino, in larga misura indipendentemente dal colore e dalla riflettanza dell'oggetto. Il sensore può operare anche a risoluzione ridotta quando sia richiesta una maggiore frequenza di aggiornamento, comunica tramite interfaccia I^2C fino a 1 MHz, ed esegue internamente l'intera elaborazione della misura, incluso il trattamento degli istogrammi per la soppressione del crosstalk, cosicchè al microcontrollore è richiesta soltanto la lettura di un vettore compatto di distanze. I sensori tattili sono costituiti da due matrici FSR a film sottile, ciascuna organizzata in trentadue zone piezoresistive indipendenti su un unico substrato flessibile dello spessore di circa 0.3 mm; la resistenza di ciascuna zona decresce in modo monotono all'aumentare della pressione applicata. Come per ogni sensore resistivo di tale natura, i valori dichiarati dal costruttore sono da intendersi come indicativi, essendo soggetti a variabilità tra esemplari, isteresi meccanica e deriva sotto carico prolungato.

La catena di interfacciamento delle matrici FSR è composta da quattro elementi visibili in Figura 5.8. Il primo è costituito dalla coppia di multiplexer analogici CD74HC4067 a 16 canali, uno dedicato alle righe e uno alle colonne, pilotati direttamente dal microcontrollore mediante linee di selezione digitali. Il secondo è il partitore di tensione, nel quale il taxel selezionato è posto in serie a un trimmer da 100 $K\Omega$, così da consentire, dopo l'assemblaggio, la regolazione della porzione del campo dell'ADC su cui si distribuisce la risposta del sensore, costituendo di fatto una taratura manuale della catena. Il terzo elemento è uno stadio di buffering realizzato mediante l'amplificatore operazionale Microchip MCP6001, configurato come inseguitore a guadagno unitario, che presenta un'elevata impedenza di ingresso al partitore di tensione, evitandone il caricamento da parte della circuiteria a valle. Il quarto elemento è una protezione dell'ingresso ADC, costituita da una resistenza serie da 22 Ω e da un diodo Zener da 3.3 V verso massa, che limita la tensione in ingresso al microcontrollore entro un margine di sicurezza in presenza di transitori o condizioni fuori scala.

L'unità di elaborazione e controllo è un microcontrollore STMicroelectronics STM32F446RE, basato su un core Arm Cortex-M4 a 32 bit, operante fino a 180 MHz, e dotato di 512 KB di

memoria flash, 128 KB di SRAM, tre convertitori analogico-digitali a 12 bit, molteplici interfacce I^2C e un controllore CAN integrato. Nella fase di prototipazione il microcontrollore è impiegato nella forma della scheda di sviluppo NUCLEO-F446RE, che integra un debugger ST-LINK/V2-1, espone tutti i pin necessari e offre opzioni di alimentazione flessibili. Il controllore CAN integrato è completato da un transceiver Microchip MCP2551, conforme allo standard ISO-11898 fino a 1 Mb/s, che converte i segnali logici TXD/RXD nella coppia differenziale CANH/CANL, tollera transitori e cortocircuiti sul bus e integra protezione da spegnimento termico.

Su tale base è stata progettata una scheda elettronica custom a quattro strati, destinata a sostituire la scheda di sviluppo NUCLEO una volta completata la fase di validazione. Lo stack-up prevede un piano di massa continuo su uno strato interno e la distribuzione dell'alimentazione su un secondo strato interno, al fine di separare la circuiteria digitale da quella analogica. Il posizionamento dei componenti segue il percorso del segnale, con i connettori dei sensori tattili e i multiplexer collocati in prossimità del partitore, del buffer e della protezione, il microcontrollore in posizione centrale, e lo stadio di conversione di potenza unitamente al transceiver CAN verso il bordo di alimentazione. L'ingombro complessivo, dell'ordine di 40×40 mm, è coerente con il vincolo di integrazione compatta discusso nel capitolo relativo al progetto concettuale.

6. Architettura del firmware

Il firmware in esecuzione sul microcontrollore STM32F446RE è organizzato come un ciclo *bare-metal* gestito tramite interrupt. L'applicazione è strutturata attorno a un unico hot path, costituito dall'acquisizione delle matrici tattili, dalla stima della forza di contatto e dall'aggiornamento della legge di ammettenza, affiancato da attività secondarie, quali la gestione dei frame CAN, la misura del sensore ToF e la telemetria via UART, implementate come routine di interrupt e macchine a stati non bloccanti.

L'acquisizione tattile legge le due matrici, ciascuna organizzata in 8 righe e 4 colonne, attraverso la coppia di multiplexer condivisa e un unico canale ADC. Poichè i 64 taxel complessivi non possono essere campionati in parallelo, la scansione è implementata come una macchina a stati non bloccante, gestita tramite DMA, che consente al processore di rimanere disponibile per altre periferiche durante l'esecuzione delle conversioni. Un aspetto rilevante dell'implementazione

riguarda il transitorio introdotto a ogni cambio delle linee di selezione dei multiplexer, dal momento che il percorso analogico non raggiunge istantaneamente il nuovo valore a regime, per effetto del transitorio del buffer e della commutazione del multiplexer stesso, per cui la prima conversione successiva al cambio di canale risulterebbe ancora influenzata dal valore precedente. Si è scelto pertanto di scartare sistematicamente tale prima conversione, accettando il raddoppio del numero di conversioni per scansione a fronte della garanzia di operare sempre su campioni già assestati, senza dover caratterizzare e imporre un tempo di assestamento fisso. Ai campioni grezzi così ottenuti sono applicate due correzioni successive, una compensazione di offset, calcolata all'avvio mediante media di letture a vuoto, e un filtro a media mobile esponenziale a singolo polo, con coefficiente pari a 0.6, scelto come compromesso tra reiezione del rumore e prontezza di risposta. I valori filtrati sono infine convertiti in una stima di forza espressa in newton, mediante inversione della relazione del partitore di tensione e applicazione di un modello lineare conduttanza-forza, la cui costante di proporzionalità è destinata a essere caratterizzata sperimentalmente sui taxel specifici impiegati.

Il controllo di ammettenza richiede, per ciascuna iterazione, un unico valore scalare di forza, e non l'intera mappa di pressione restituita dalla matrice. Mediare indiscriminatamente tutti i taxel di una matrice diluirebbe una zona di contatto localizzata sotto la maggioranza delle celle scariche, producendo una stima sistematicamente sottostimata; per tale ragione si individua, in ciascuna matrice, il taxel con la lettura più elevata, che viene mediato con le celle immediatamente adiacenti prima della conversione in newton, ottenendo così una stima rappresentativa della regione di contatto e al contempo più robusta rispetto all'impiego del solo valore di picco. Su tale stima opera la legge di controllo, la quale implementa la dinamica virtuale massa-molla-smorzatore. A ogni iterazione viene calcolato l'errore tra la forza desiderata e la forza stimata, viene applicata una banda morta per sopprimere oscillazioni indesiderate attorno al punto di lavoro, e viene infine integrata nel tempo l'accelerazione virtuale mediante il metodo di Eulero, ottenendo la velocità da comandare al motore.

$$\ddot{x} = \frac{K\,(x_d - x) - B\,\dot{x} - K\,K_p\,F_{\mathrm{err}}}{M} \tag{1}$$

È rilevante osservare che i parametri virtuali di massa, smorzamento, rigidezza e guadagno sull'errore di forza non sono fissati in fase di compilazione, bensì aggiornabili in tempo reale dal controllore del rover tramite il protocollo CANopen, il che consente di ritarare il comportamento di

ammettenza senza alcuna necessità di riprogrammazione del firmware.

Il protocollo CANopen è implementato limitatamente al sottoinsieme di funzionalità effettivamente richiesto da questa applicazione. I trasferimenti SDO di tipo *expedited* vengono impiegati per i valori scalari, mentre un meccanismo segmentato è riservato ai dati di dimensione maggiore, quali le due matrici di forza e la mappa di distanze del sensore ToF. Il microcontrollore assume ruoli differenti a seconda dell'interlocutore, operando come client SDO nei confronti del driver del motore, e come server SDO, sul proprio dizionario degli oggetti, nei confronti del controllore del rover. L'interfacciamento con il sensore ToF è realizzato mediante l'*Ultra Lite Driver* fornito da STMicroelectronics, con l'acquisizione dei dati gestita tramite interrupt anzichè tramite polling, così da non introdurre attese bloccanti sul bus I^2C. La calibrazione del crosstalk del sensore, necessaria per compensare le riflessioni interne generate dal vetro di protezione, è stata eseguita una singola volta sul prototipo, e il risultato è stato memorizzato come costante nel firmware, da cui viene ricaricato a ogni accensione. Oltre al canale CAN, il firmware trasmette via UART un flusso di telemetria leggibile, ricevuto da un'applicazione desktop sviluppata in Python che visualizza in tempo reale le mappe tattili e la mappa di profondità del sensore ToF.

7. Calibrazione e dimostrazione funzionale preliminare

Il sensore VL53L7CX è autocalibrato in fabbrica, ma, essendo montato dietro la finestra protettiva della scocca, è soggetto al fenomeno di crosstalk, per cui una porzione del segnale infrarosso emesso si riflette internamente sul vetro prima di raggiungere il bersaglio; alle corte distanze tale riflesso può risultare comparabile o superiore al segnale utile, inducendo misure di distanza artificiosamente ravvicinate. La correzione è stata effettuata mediante la procedura di calibrazione del crosstalk fornita da STMicroelectronics, ponendo un bersaglio piano di riflettanza nota a distanza fissa nel campo visivo del sensore e generando una tabella di compensazione per zona, specifica per la geometria di copertura del gripper. Una volta calibrato, il sensore ha mostrato letture di distanza coerenti rispetto a un insieme di distanze di riferimento note.

La calibrazione dei sensori FSR sfrutta la condivisione, tra le due matrici, dello stesso stadio di partitore di tensione, per cui risulta sufficiente un'unica regolazione del trimmer, applicata congiuntamente a entrambe le catene di sensing. Un peso di riferimento di 2 Kg chilogrammi,

corrispondente all'obiettivo di forza di presa stabilito nel progetto, è stato applicato sull'area sensibile di una delle due matrici, e il trimmer è stato regolato, monitorando la tensione ai suoi capi mediante oscilloscopio, fino al raggiungimento di 3.3 V sotto tale carico di riferimento. La medesima configurazione è stata successivamente verificata ripetendo la prova sul secondo dito, confermando una risposta confrontabile su entrambe le catene.

Con entrambi i sensori calibrati secondo le modalità appena descritte, l'intero anello di controllo, comprendente sensing tattile, sensing ToF e controllore di ammettenza embedded, è stato testato sul prototipo NUCLEO. A seguito della taratura manuale dei parametri virtuali del controllore, il sistema in anello chiuso ha mostrato il comportamento atteso, chiudendo le dita con una risposta visibilmente cedevole al contatto con l'oggetto, e riducendo la forza applicata in risposta alla resistenza incontrata anzichè proseguire in modo rigido. Il sensore ToF ha riportato correttamente le distanze al variare della posizione degli oggetti nel campo visivo, e l'insieme dei dati sensoriali, unitamente allo stato interno del controllore, è stato trasmesso tramite CAN e visualizzato in tempo reale sull'applicazione desktop, senza interferenze percettibili tra controllo e telemetria.

8. Modello virtuale in Isaac Sim

Lo sviluppo del modello virtuale in Isaac Sim è stato condotto secondo un approccio deliberatamente incrementale. Il gripper è stato dapprima modellato e dotato con i propri sensori come oggetto articolato autonomo, al fine di verificarne cinematica, comportamento di contatto e uscite sensoriali in un ambiente isolato, prima di procedere all'integrazione con il braccio robotico completo.

Il modello cinematico e dinamico del gripper è importato da una descrizione URDF generata a partire dall'assieme CAD, preservando la gerarchia della progettazione meccanica come un unico oggetto articolato ancorato al riferimento globale. Il singolo grado di libertè meccanico della trasmissione pignone-cremagliera è riprodotto in simulazione mediante due giunti prismatici, uno per dito, dei quali soltanto uno risulta effettivamente azionato, con un'attuazione in posizione di tipo forza avente rigidezza pari a 5000 Nm e smorzamento pari a 50 Ns/m. Il secondo giunto è vincolato al primo tramite un giunto virtuale (*mimic joint*) di PhysX, con rapporto di trasmissione unitario e segno opposto, riproducendo così l'accoppiamento meccanico reale senza la necessità di simulare il contatto tra i denti degli ingranaggi, operazione computazionalmente onerosa e di

difficile taratura. Le proprietà di massa sono assegnate a partire dal progetto CAD, in coerenza con le scelte di materiale descritte nel capitolo relativo alla progettazione meccanica.

La simulazione delle matrici tattili FSR costituisce l'aspetto più originale del modello virtuale, poichè determina in misura significativa la rilevanza dei dati tattili simulati ai fini dell'addestramento di una politica applicabile all'hardware reale. Ciascuna matrice è rappresentata mediante una griglia di 32 corpi rigidi indipendenti, uno per taxel, ciascuno vincolato al link del dito corrispondente mediante un giunto fisso. Tale approccio consente l'interrogazione indipendente del rilevamento dei contatti per ciascuna delle 64 celle complessive, ottenendo una mappa di forza spaziale completa anzichè un segnale aggregato per dito. Il sensore ToF è riprodotto mediante una tecnica di *ray-casting* configurata per emulare il comportamento multizona del sensore reale.

Una volta verificata la correttezza strutturale del modello autonomo, il gripper è stato integrato con il modello completo del braccio robotico, un manipolatore seriale a sei giunti, mediante un giunto fisso applicato all'ultimo link del braccio, ottenendo un'unica struttura articolata con sette gradi di libertà complessivi, corrispondenti allo spazio delle azioni impiegato dalla politica di reinforcement learning. I due gruppi di giunti sono stati modellati con parametri di attuazione differenti, in coerenza con il fatto che braccio e gripper sono azionati da attuatori fisicamente distinti, e il giunto del gripper è caratterizzato da una rigidezza sensibilmente inferiore, coerentemente con il comportamento cedevole atteso sotto controllo di ammettenza. Al fine di prevenire il rilevamento di autocollisioni spurie tra link contigui, la rilevazione delle collisioni è stata disabilitata tra coppie di link adiacenti lungo la catena cinematica, mentre le autocollisioni effettive sono monitorate e penalizzate durante l'addestramento mediante un sensore di contatto dedicato.

Infine, la randomizzazione del dominio è stata implementata sulla posa della probe, individuata come l'incertezza operativa più rilevante per la Probing Task. A inizio di ciascun episodio di addestramento, la posizione orizzontale della probe è campionata uniformemente entro una regione rettangolare antistante il gripper, mentre l'orientamento è randomizzato attorno all'asse verticale sull'intero angolo giro, con rollio e beccheggio nulli. Poichè ciascun ambiente parallelo viene ripristinato in modo indipendente, tale procedura consente a un singolo addestramento di esporre la politica a un'ampia varietà di configurazioni della probe.

9. Addestramento della politica di reinforcement learning

Il problema della presa è formulato come un processo decisionale di Markov. A ciascun passo di controllo, la politica riceve un vettore di osservazione di 154 dimensioni, ottenuto concatenando i 64 valori del sensore ToF, i 64 valori tattili delle due matrici FSR e 26 valori di stato propriocettivo e relativo al compito, comprendenti posizioni e velocità dei 7 giunti, posizione dell'end effector, posizione della probe e posizione dell'obiettivo di posizionamento. L'azione, di dimensione pari a 7, è interpretata come un incremento di posizione applicato ai giunti, ed è successivamente saturata entro i limiti di giunto. Ciascun episodio segue la struttura di avvicinamento, presa, sollevamento e posizionamento. La probe è collocata in posa casuale davanti al gripper, e la politica deve avvicinarvisi, chiudere le dita, sollevarla e trasportarla fino a una posizione obiettivo prestabilita. L'episodio termina con successo qualora la probe raggiunga una distanza inferiore a quattro centimetri dall'obiettivo, con fallimento in caso di caduta della probe o di autocollisione con forza superiore a 50 N, ed è altrimenti troncato dopo 25 secondi simulati.

La funzione di ricompensa combina quattro componenti, allineamento, presa, sollevamento e posizionamento, con tre ulteriori termini di penalità. Il termine di allineamento è calcolato interamente a partire dalla griglia del sensore ToF. Il termine di presa è invece concesso esclusivamente quando l'end effector si trova in prossimità immediata della probe e il gripper sta effettuando la chiusura, e premia il contatto simultaneo di entrambe le dita in misura tripla rispetto al contatto singolo, mentre una penalità distinta scoraggia la chiusura indiscriminata delle dita quando la probe risulta ancora distante. I termini di sollevamento e posizionamento premiano, rispettivamente, l'altezza guadagnata rispetto al piano di appoggio e la riduzione della distanza dall'obiettivo finale, quest'ultimo attivo solamente a seguito di un sollevamento già conseguito.

La pesatura di tali componenti non è fissa per l'intera durata dell'addestramento, bensì governata da un curriculum esplicito articolato in quattro stadi, il cui avanzamento è gestito automaticamente da una funzione di callback dedicata. Quest'ultima monitora una finestra scorrevole degli ultimi cento esiti di episodio e verifica periodicamente, ogni 2048 passi, se il tasso di successo abbia raggiunto il 60 %, nel qual caso l'ambiente avanza allo stadio successivo. Tale meccanismo rende il curriculum adattivo rispetto alle prestazioni effettive della politica, anzichè vincolato a una progressione temporale prestabilita, come riportato in tabella.

Hyperparameter	**Value**
Rollout buffer size (n_{steps})	2,048 steps/environment
Mini-batch size	1,024
Epochs per update	10
Learning rate	linear decay, $3 \times 10^{-4} \rightarrow 1 \times 10^{-5}$
Discount factor γ	0.99
GAE parameter λ	0.95
Clipping range	0.2
Max gradient norm	0.5
Entropy coefficient	0.05
Parallel environments	256
Random seed	42

Table 0: PPO hyperparameters used for training.

L'algoritmo impiegato è la *Proximal Policy Optimization* (PPO), nell'implementazione Stable-Baselines3, eseguita su 256 ambienti paralleli simulati su GPU. Gli Hyperparameters principali comprendono un tasso di apprendimento con decadimento lineare, un discount factor pari 0.99 e un coefficiente di entropia relativamente elevato, scelto per sostenere l'esplorazione nelle diverse fasi del compito. Non è inoltre previsto un numero prefissato di passi totali di addestramento, essendo quest'ultimo interrotto soltanto al raggiungimento del criterio di successo relativo allo stadio finale del curriculum.

L'architettura della policy è costituita da un estrattore di caratteristiche multimodale, che elabora i dati del sensore ToF e delle matrici tattili mediante rami convoluzionali dedicati, in grado di riconoscere pattern spaziali quali l'orientamento della probe o la localizzazione del contatto, prima di fondere tali rappresentazioni con lo stato propriocettivo del sistema. La rete è stata mantenuta deliberatamente di dimensioni contenute rispetto a quanto tipicamente riportato in letteratura per compiti analoghi, scelta che non costituisce una semplificazione accidentale, bensì un vincolo di progetto legato all'obiettivo di lungo periodo di eseguire l'inferenza della policy direttamente sul microcontrollore STM32F446RE, le cui risorse limitate non consentirebbero l'esecuzione di un modello sostanzialmente più ampio entro il medesimo budget temporale reale.

Al momento della stesura della presente tesi, l'addestramento risulta ancora in corso. Dal punto

di vista qualitativo, la politica ha appreso in modo affidabile il comportamento di avvicinamento corrispondente al primo stadio del curriculum, orientandosi correttamente verso la probe sulla base delle sole letture del sensore ToF, indipendentemente dalla sua posa randomizzata. Il comportamento di presa, corrispondente al secondo stadio, risulta presente ma non ancora consistente. In un numero limitato di episodi la policy è riuscita a chiudere entrambe le dita sulla probe e a sollevarla brevemente, ma tali successi si manifestano in modo sporadico, evidenza coerente con una policy che non ha ancora accumulato esperienza sufficiente per superare stabilmente la soglia di successo richiesta dal curriculum. Le curve di addestramento registrate confermano tale interpretazione, mostrando una distanza media tra end effector e probe con andamento decrescente costante, e una ricompensa media di episodio che cresce corrispondentemente, senza che nessuna delle due grandezze abbia ancora raggiunto un plateau.

10. Conclusioni, limiti e sviluppi futuri

La tesi ha prodotto una piattaforma completa per la presa sensorizzata. Sono stati progettati e realizzati un end effector a due dita con doppia matrice FSR e ToF multi-zona e un'architettura di comunicazione CANopen che condivide un unico bus tra driver motore, microcontrollore del gripper e controllore del rover. Sono stati inoltre sviluppati un controllore di ammettenza embedded, il cui funzionamento in tempo reale è stato dimostrato sul prototipo, e una procedura di calibrazione dei due sensori. Sul versante dell'apprendimento, sono stati costruiti un modello virtuale del gripper, fedele ai sensori reali ed stato succesivamente integrato con il braccio completo. Una pipeline di addestramento con curriculum e rete compatta è stata realizzata per le oparazione di pick and place prevista dalla Probing Task. Alla data di stesura di questa tesi, gli elementi meccanici, elettronici e firmware del gripper risultano funzionali e operanti congiuntamente. Entrambi i sensori sono stati calibrati e, una volta fatto, ne è stato dimostrato il corretto funzionamento del controllo di ammittenza. Nello specifico qusto è stato tarato specificamente per la presa della probe, andando a chiudere l'anello di controllo in tempo reale con comportamento cedevole e trasmettendo la relativa telemetria via CAN. Sul versante dell'apprendimento, l'addestramento della politica non è ancora concluso. La politica ha appreso in modo affidabile la fase di avvicinamento, mentre la presa resta sporadica e il curriculum non è ancora avanzato oltre gli stadi iniziali. Non sono pertanto riportati,

in questa sede, un tasso di successo definitivo, una valutazione su condizioni non precedentemente osservate, nè un confronto diretto tra controllo classico e controllo appreso.

Sono oltretutto presenti alcune limitazioni, infatti il controllo di ammettenza è stato tarato e mostrato solo sulla presa della probe, senza un collaudo formale né misure quantitative, e il suo comportamento su altri oggetti e nelle condizioni del Maintenance non è caratterizzato. La politica non è convergente e i risultati riflettono un addestramento in corso. La randomizzazione del dominio è limitata alla posa della probe, mentre massa, attrito e altre proprietà restano fisse. Nessun rumore è iniettato nei sensori simulati. Tatto e ToF usati in addestramento sono quindi idealizzati, privi di rumore, deriva e variabilità tra esemplari dei sensori reali, e la capacità della politica di tollerare queste imperfezioni non è stabilita. La risoluzione 8×8 del ToF, adeguata alla rilevazione grossolana pre-presa, pone inoltre un limite alla finezza con cui si stimano posizione e orientamento della probe. Infine la lenta convergenza è in parte attribuibile alla rete volutamente piccola, scelta per l'inferenza sul microcontrollore.

Con riferimento agli sviluppi futuri, il passo più immediato consiste nel completamento dell'addestramento in corso, al fine di verificare l'avanzamento del curriculum attraverso gli stadi di sollevamento e posizionamento completo, unitamente all'estensione della randomizzazione del dominio a massa e attrito degli oggetti e all'introduzione di modelli di rumore realistici nei sensori simulati. Sul piano del sensing, risulta opportuna la valutazione di un sensore ToF a risoluzione superiore, quale il TMF8829 di ams OSRAM, capace di risoluzioni configurabili fino a 1536 punti di misura, che consentirebbe una stima più accurata di posizione e orientamento della probe. Sul piano dell'apprendimento, il vincolo dimensionale sulla rete della policy è specifico del microcontrollore STM32F446RE e non costituisce un limite intrinseco all'inferenza embedded. La famiglia STM32N6, dotata di acceleratore hardware dedicato all'esecuzione di reti neurali, rimuoverebbe tale vincolo, consentendo l'impiego di reti di maggiore capacità. Una volta completati tali sviluppi, la naturale prosecuzione del lavoro consisterà nella calibrazione del modello virtuale rispetto a misure reali, nel benchmark formale del controllore di ammettenza su un insieme definito di oggetti e prove ripetute, e nel conseguente confronto diretto, su hardware fisico e con identiche modalità sensoriali, tra il controllo classico e la politica appresa.

Contents

Abstract ii

Italian Summary iv

1 Introduction 1

1.1 Thesis context . . . 1

1.2 Motivation . . . 3

1.3 Problem statement . . . 4

1.4 Contributions . . . 5

1.5 Thesis outline . . . 6

2 Related work 7

2.1 Tactile and Force-Sensorized Grippers . . . 7

2.2 Proximity/ToF Sensing for Pre-Grasp Planning . . . 11

2.3 Admittance / Impedance Control for Grasping . . . 13

2.4 Reinforcement Learning: Advantages and Limitations . . . 15

2.5 Digital Twins and Sim-to-Real Transfer in Manipulation . . . 16

2.6 Gap Identified . . . 18

3 Conceptual Design 20

3.1 Design Requirements . . . 20

3.2 Block Diagram . . . 22

3.3 Functional Decomposition . . . 24

3.4 Concept Generation - Electronic . . . 27

3.5 Concept Generation - Mechanics . . . 29

4 Detailed mechanical design 31

4.1 Manufacturing Process and Material Selection . . . 31

4.2 Overall Structural Assembly . . . 32

4.3 Cable Routing . . . 34
4.4 Rear Interface and Robotic-Arm Mount . . . 34
4.5 Motion Transmission: Pinion, Racks, and Linear Guides . . . 35

5 Detailed electronic design **37**
5.1 Motor . . . 37
5.2 Power Conversion Stage - LDO . . . 40
5.3 Sensors . . . 41
5.3.1 Time of Flight (ToF) Sensors . . . 41
5.3.2 Tactile Sensors - Force Sensitive Resistor Matrix . . . 42
5.4 FSR Matrix Interfacing Circuitry . . . 45
5.4.1 Row and Column Multiplexing . . . 45
5.4.2 Voltage Divider . . . 46
5.4.3 Signal Buffering . . . 46
5.4.4 ADC Protection . . . 47
5.5 Microcontroller . . . 47
5.6 CAN Transceiver - MCP2551 . . . 49
5.7 PCB Layout . . . 50
5.7.1 Board-Level Integration and Auxiliary Circuitry . . . 51
5.7.2 Stack-Up and Grounding Strategy . . . 53
5.7.3 Physical Footprint and Component Placement . . . 55

6 Firmware architecture **57**
6.1 Firmware Architecture Overview . . . 57
6.2 Tactile Sensing Acquisition . . . 59
6.3 Contact Force Estimation for Control . . . 61
6.4 Admittance Control Implementation . . . 62
6.5 CANopen Communication Layer . . . 63
6.6 Motor Interface . . . 64
6.7 Proximity Sensing (ToF) Integration . . . 64
6.8 Telemetry and Desktop Visualization . . . 65

7 Preliminary Functional Demonstration **67**

7.1 ToF Sensor Calibration 67

7.2 FSR Sensor Calibration 68

7.3 System-Level Functional Demonstration 70

8 Virtual model development in Isaac Sim **72**

8.1 Modeling the gripper 72

8.2 Simulating the FSR Tactile Arrays 73

8.3 Simulating the ToF Sensor 75

8.4 Integration with the Robotic Arm 75

8.5 Domain Randomization 78

9 Reinforcement Learning Policy Training **79**

9.1 Task Formulation 79

9.1.1 Observation Space 79

9.1.2 Action Space 80

9.1.3 Episode Structure 80

9.2 Reward Design 80

9.3 Algorithm and Training Setup 82

9.4 Policy Architecture 84

9.5 Training Progress and Preliminary Results 86

10 Conclusions and Future Work **90**

10.1 Summary of Contributions 90

10.2 Current Status 90

10.3 Limitations 91

10.4 Future Work 92

References **94**

List of Figures

1.1 The ProjectRED rover prototype, designed and built by the student team of the DISMI Department at UNIMORE, competing in the European Rover Challenge. The rover is equipped with a robotic arm and gripper, which are the focus of this thesis. 2

2.1 (a) A commercial FSR sensor (Tekscan FlexiForce A301). (b) Construction details of a typical FSR sensor, showing the resistive layer, conductive traces, and protective layers. 8
2.2 GelSight Tactile Sensor. 8
2.3 ReSkin and uSkin Tactile Sensors 9
2.4 BioTac Tactile Sensor. 10
2.5 Robotiq Tactile Sensor. 10
2.6 VL53L7CX Multi-Zone ToF Sensor. 12

3.1 Target objects for the two ERC tasks. The Maintenance task panel (a) requires actuating lever and rotary switches, inserting an IEC320 C14 plug into a socket, and placing a metal plate on an electromagnet. The Probing task probe (b) is a cylindrical aluminum body 200 mm long, with a tapered tip and a gripping section on the order of 20-40 mm in diameter. 21
3.2 System block diagram 24
3.3 Gripper's principal function 25
3.4 Functional decomposition 27

4.1 Exploded-color view of the end effector shell assembly: the cyan main body and the pink and green front covers are fastened together with the carbon-fiber plate (yellow) sandwiched between them. 32
4.2 Top view of the end effector showing the unobstructed 60°×60° field of view of the palm-mounted ToF sensor, extended here to a 1000 mm range for illustration. . . . 33

4.3 (a) Rear-isometric view of the end effector assembly, showing the mounting flange and the motor-side interface. (b) Quick-release mechanism in operation. The gripper can be attached to or detached from the robotic arm without tools, facilitating rapid iteration and testing during development. 35
4.4 Internal rack-and-pinion transmission mounted on the carbon-fiber plate (yellow): the motor drives a central pinion that engages two racks, each mounted on its own linear-guide carriage. 36

5.1 FD1.1G applied to NEMA 17 37
5.2 Electrical Characteristics 38
5.3 Motor Pinout 39
5.4 LDO Schematics 40
5.5 ToF sensor module and field of view 42
5.6 FSR Matrix sensors 43
5.7 Typical force-resistance characteristic of a single FSR taxel, showing the monotonic decrease in resistance with increasing applied load. 44
5.8 FSR matrix interfacing circuitry schematic 45
5.9 NUCLEO-F446RE development board 48
5.10 MCP2551 CAN Transceiver 49
5.11 Support circuitry for the STM32F446RET6 microcontroller on the custom PCB, including SWD programming/debugging header, BOOT0 switch, NRST push-button, and status LED. 52
5.12 CAN transceiver and associated passive network on the custom PCB, including RC filter and jumper-selectable termination resistor. 53
5.13 PCB stack-up 54
5.14 Front and back sides of the custom PCB, showing the compact component placement and the four M2 mounting holes at the corners. The front side hosts the microcontroller, multiplexers, and FSR connectors, while the back side accommodates the power-conversion stage and CAN transceiver. 56

6.1 Firmware architecture overview 58

6.2 Tactile sensing acquisition state machine 60
6.3 Tactile sensing acquisition pipeline 61
6.4 Desktop telemetry visualizer, showing the two FSR matrices (left) and the ToF depth map (right) alongside a live mean-force readout and a raw console. The FSR heatmaps are colour-mapped along a blue–green–yellow–red ramp, with the quadrant containing the current peak reading outlined; the ToF map uses a separate near-to-far colour ramp with invalid zones shown in dark grey. 66

7.1 The crosstalk phenomenon in the ToF sensor, showing the reflected signal that can interfere with the cover glass. 68
7.2 Calibration setup 68
7.3 FSR sensor calibration: a 2 kg reference weight is applied to the sensing area of one finger's FSR matrix, while the voltage divider trimmer is adjusted to bring the voltage drop across it to 3.3 V. This single trimmer setting governs the divider used by both matrices, and was verified by repeating the test on the second finger. 69
7.4 The complete control loop – tactile sensing, ToF sensing, and the embedded admittance controller – was exercised on the NUCLEO prototype. The fingers closed with a visibly compliant response when contact was made with an object, and the ToF sensor reported distances correctly as objects were moved into and out of its field of view. All sensor data and the controller's internal state were streamed over CAN and displayed in real time on a desktop monitoring application. 71

8.1 The FSR tactile arrays are represented in the scene as a grid of 32 individual taxel prims per finger, each instantiated as its own rigid body and rigidly attached to the parent link through a fixed joint. The taxels are arranged in eight rows and four columns, matching the layout of the real sensor. 74
8.2 The palm-mounted ToF sensor is represented in the simulation by a dedicated Lidar prim, attached at the sensor's real mounting location on the gripper's palm. The sensor's ray-casting pattern is configured to emulate the VL53L7CX's multi-zone ranging behavior, producing an 8×8 grid of independent range measurements, directly analogous to the real sensor's output. 76

8.3 The robotic arm assembly, with the gripper mounted on the end effector. 77

9.1 The policy architecture is a convolutional, modality-aware design that processes the ToF and tactile channels through separate convolutional branches before fusing them with the proprioceptive state. The resulting shared representation is then passed to separate actor and critic heads for action selection and value estimation. . 86

9.2 The mean distance between the end effector and the probe shows a consistent downward trend across training, reflecting the increasingly accurate approach behavior while the mean episode reward increases correspondingly as training progresses. . 88

9.3 The total loss recorded during training shows a noisy but downward trend over the course of the run. 89

List of Tables

9.1 Reward weighting and success criteria across the four curriculum stages. 82
9.2 PPO hyperparameters used for training. 84

1. Introduction

1.1 Thesis context

This project originates from a concrete operational need of ProjectRED, the student team of the DISMI Department (Dipartimento di Scienze e Metodi dell'Ingegneria) at the University of Modena and Reggio Emilia (UNIMORE), whose main activity is designing and building a rover prototype for extraterrestrial exploration (Figure 1.1) to compete in international competitions, foremost among them the European Rover Challenge. Founded in 2019 with a "Learning by Doing" approach, the team has competed in several editions of the European Rover Challenge with a progression of rovers, and its results have placed it consistently among the top international university teams in the competition.

The European Rover Challenge (ERC) is an annual, planetary-robotics-themed competition in which university teams design and build rovers to perform tasks modeled on real space-agency mission scenarios, competing on an artificial Mars terrain (the "Mars Yard"). The competition is structured around several independent tasks – Science, Navigation, Maintenance, Probing, and Presentation – each targeting a specific set of technologies relevant to future planetary exploration missions.

Two of these tasks are directly relevant to this thesis, since both require the rover to perform dexterous, contact-rich manipulation with an onboard robotic arm and end effector. The Maintenance Task requires the rover to operate a control panel designed for human use – setting lever and rotary switches, inserting a plug into a socket, and placing a metal plate on an electromagnet – entirely through the manipulation capabilities of the rover's arm and gripper, without any physical intervention by the team. The Probing Task requires the rover to locate and collect three probes scattered across the Mars Yard, grasp them with the end effector, and store them onboard, with the final score depending on both successful collection and completion time. Both tasks are scored, among other criteria, on the level of automation achieved in detection, approach, and handling, rather than purely on manual teleoperation.

The sensorized gripper developed in this thesis is the end effector ProjectRED intends to deploy

for these two tasks. Tactile feedback and admittance control directly address the contact-rich, force-sensitive nature of the Maintenance Task, where panel elements can be damaged by excessive force or torque and must be manipulated one at a time with controlled compliance. The virtual-model and reinforcement-learning components of this work, in particular, are developed with the Probing Task specifically in mind: probing objects must be located and grasped reliably and quickly under a strict time budget, which motivates investigating a learned grasping policy capable of generalizing across probe positions and approach conditions, as a complement to the classical, hand-tuned admittance controller. This thesis therefore contributes both a piece of competition-ready hardware and the first stage of a longer-term research pipeline that ProjectRED can continue to develop across future editions of the ERC.



Figure 1.1: The ProjectRED rover prototype, designed and built by the student team of the DISMI Department at UNIMORE, competing in the European Rover Challenge. The rover is equipped with a robotic arm and gripper, which are the focus of this thesis.

1.2 Motivation

Despite decades of research, enabling robots to reliably grasp everyday objects remains challenging. Objects vary widely in stiffness, geometry, surface friction, and mass distribution; as a result, no single set of gripper parameters or control gains performs well across all cases. A gripper tuned for rigid, high-friction objects may perform poorly on soft, slippery, or irregularly shaped items. Robust grasping therefore requires controllers that can estimate relevant object properties online and adapt in real time, rather than relying on fixed, prior assumptions [50].

Tactile feedback is central to this capability and cannot be fully replaced by vision or joint-torque sensing. In contact-rich manipulation, high-resolution information about fingertip forces and pressure distribution is essential. Once contact occurs, visual feedback is often occluded by the gripper, and joint-torque or motor-current measurements provide only indirect, low-spatial-resolution cues that are sensitive to friction and backlash. In contrast, tactile sensing directly measures contact location and area, applied forces, and incipient slip at the interface between the object and the gripper [3, 10, 20, 49].

However, tactile sensing is inherently limited prior to contact. Proximity or Time-of-Flight (ToF) sensing complements tactile feedback by providing pre-contact geometric information – such as object presence, approximate shape, and approach direction – which supports planning and finger closure. In many sensorized gripper systems, pre-grasp perception and in-grasp tactile control are implemented as separate subsystems, often managed by different hardware and processing stages. Integrating both sensing modalities within a single low-level embedded controller could enable a tighter, lower-latency perception-action loop spanning the full grasping sequence, from approach to contact and force regulation [52].

Simulation-based reinforcement learning (RL) offers an attractive alternative to manual controller tuning. Conventional admittance controllers typically require object-specific parameterization and often generalize poorly to novel objects or unanticipated contact conditions. RL, by contrast, can in principle learn grasping strategies that are robust across diverse object properties without continual retuning. Real-world performance, however, depends strongly on the fidelity of the simulated sensing pipeline. The simulation-to-reality gap is typically larger for tactile sensing than for joint encoders or other proprioceptive signals due to the difficulty of modeling contact-to-signal

transduction, variability introduced by protective layers, and sensor-level noise. Consequently, effective sim-to-real transfer for grasping requires simulation environments that replicate the sensor modalities and characteristics of physical systems, rather than relying on idealized or purely synthetic tactile measurements.

1.3 Problem statement

This project revolves around three main research questions. Together, they help define the system's scope and what we'll use to measure its success.

The first question looks into whether a budget-friendly FSR-matrix combined with a ToF sensor suite, when processed locally on a microcontroller, can effectively support a real-time admittance control loop for gripping objects with varying stiffness. This inquiry has been a driving force behind our hardware design from the start. Instead of sending sensor data to a host computer, we aimed to keep everything – from sensing to actuation – on one embedded platform. The idea was to close the control loop locally while using CANopen just for telemetry and higher-level commands. To tackle this question, we needed to integrate the tactile and proximity sensors at both the mechanical and electronic levels, create an effective real-time sensor-reading and control scheme, and ensure stable and responsive operations across different object compliances.

The second question investigates whether we can replicate these sensor modalities in a virtual model with sufficient accuracy to train a reinforcement learning policy that can also be applied to the physical gripper. This influenced our decision to base the virtual model on the same sensor modalities as the actual hardware instead of using idealized or purely kinematic simulations. It also encouraged us to recreate the FSR tactile array and ToF sensor within the simulation environment, rather than just treating the simulation as a generic source of training data. Fully answering this question involves comparing the behavior of simulated and real sensors and evaluating how well a policy trained in simulation performs on the physical gripper – a comparison this thesis begins to set up but hasn't finalized yet.

The third question, which relies on the answers to the first two, examines how a learned grasping policy stacks up against the classical admittance controller in terms of success rates, adaptability to new objects, and resilience to disturbances. This question justifies the decision to build both the

classical controller and the learning pipeline on the same hardware using identical sensors. This way, we can make a fair comparison between the two control approaches, rather than contrasting results from various studies with potentially different setups. It's a key aspect of the long-term research vision this thesis aims to establish, even if the direct comparison is still pending completion.

In summary, these three questions capture the essence of the project: the first checks the viability of the physical platform, the second explores its faithful reproduction in simulation, and the third delves into the possibilities for a rigorous, side-by-side comparison between classical and learned control once the first two are addressed.

1.4 Contributions

The contributions of this work can be summarized as follows:

1. **Mechanical, electronic, and firmware co-design of a two-finger sensorized end effector** for the European Rover Challenge's Maintenance and Probing tasks, integrating dual FSR tactile matrices and a multi-zone ToF proximity sensor into a compact, single-actuator gripper, with sensor and driver electronics designed to interface directly onto a commercial closed-loop stepper motor driver.

2. **A CANopen-based communication architecture** that shares a single field bus between the motor driver, the gripper's own sensing and control microcontroller, and the rover's higher-level controller, supporting simultaneous closed-loop motor control and multi-sensor telemetry streaming.

3. **An embedded admittance control implementation** using tactile array feedback for compliant grasping, demonstrated to run in real time on the NUCLEO prototype platform and to produce visibly compliant behavior under contact, together with a calibration procedure for both the FSR tactile matrices and the ToF sensor's crosstalk compensation.

4. **A sensor-faithful virtual model of the gripper, built in Isaac Sim and integrated with a full robotic-arm model**, reproducing the same tactile and ToF sensor modalities used on the physical hardware at the taxel and per-zone level respectively, together with an Isaac

Lab reinforcement learning pipeline – including a purpose-built, curriculum-driven reward formulation and a compact, embedded-deployment-oriented policy architecture – for training a model free pick-and-place policy targeting the ERC Probing Task.

1.5 Thesis outline

The following thesis is structured as follows: Chapter 2 reviews related work across the areas this thesis draws on – tactile and force-sensorized grippers, ToF sensing for pre-grasp planning, admittance control, and sim-to-real transfer in manipulation – closing with the specific gap this thesis addresses. Chapter 3 then lays out the conceptual design of the gripper, translating the requirements identified in this introduction into a functional and architectural specification, before Chapters 4 and 5 develop it into a physical system through the detailed mechanical and electronic design. Chapter 6 describes the firmware that brings this hardware to life as a real-time admittance controller, and Chapter 7 reports the sensor calibration work and the resulting preliminary functional demonstration of the complete prototype.

The second half of the thesis turns to the learning-based side of the platform. Chapter 8 describes the sensor-faithful virtual model built in Isaac Sim, including its integration with the full robotic-arm custom model, and Chapter 9 covers the reinforcement learning policy trained on it, from task and reward design through to the training results obtained at the time of writing. Chapter 10 closes the thesis with a summary of its contributions, an account of its current limitations, and the roadmap toward the full sim-to-real deployment on the physical gripper.

2. Related work

In this chapter, we will explore the existing literature and research related to tactile and force-sensorized grippers. The focus will be on understanding the advancements in sensor technology, the integration of tactile feedback in robotic systems, and the implications for robotic manipulation tasks. We will also discuss various approaches and methodologies that have been proposed in the field, highlighting their strengths and limitations.

2.1 Tactile and Force-Sensorized Grippers

Tactile sensing technologies for robotic grippers can be generally categorized into two main types: resistive or piezoresistive arrays – which include Force Sensing Resistors (FSRs) like those used in this project – and higher-end options like capacitive sensors and vision-based solutions, such as GelSight, ReSkin, and BioTac. Each type comes with its own set of trade-offs regarding cost, spatial resolution, bandwidth, durability, and how easily they can be integrated.

FSR sensors (as shown in Figure 2.1) measure force by detecting changes in electrical resistance that occur under mechanical stress. They're commonly used in robotic grippers and biomechanics because they're affordable and provide a solid way to sense contact force [35]. However, the main downside is the quality of the signal: while FSRs are low-cost, thin, and versatile, they're prone to issues like hysteresis, inconsistent performance, and sensitivity to the surfaces they touch. This often means that each sensor needs careful calibration to work well [2]. Despite these flaws, FSR arrays are still pretty popular for embedded tactile sensing due to their simplicity. A single FSR taxel can be read with nothing more than a voltage-divider circuit and a microcontroller ADC channel, and multiple taxels can be multiplexed into compact arrays mounted directly on a gripper's fingertip, as demonstrated in several recent low-DoF grippers using small commercial FSR matrices for distributed contact sensing [39, 27].

At the other end of the spectrum, camera-based optical tactile sensors like GelSight (Figure 2.2) provide high-resolution reconstructions of contact geometry. GelSight works by assessing deformation on a soft elastomer surface allowing it to infer contact force and movement [51]. But



(a) Force Sensing Resistor (FSR) sensor. (b) FSR Construction Details.

Figure 2.1: (a) A commercial FSR sensor (Tekscan FlexiForce A301). (b) Construction details of a typical FSR sensor, showing the resistive layer, conductive traces, and protective layers.

with that high resolution comes a bigger size, increased weight, and a heavier computational load, since every sensor has to incorporate a camera, lighting, and an image-processing system, making them more complex to fit into small, low-power devices [51, 48].

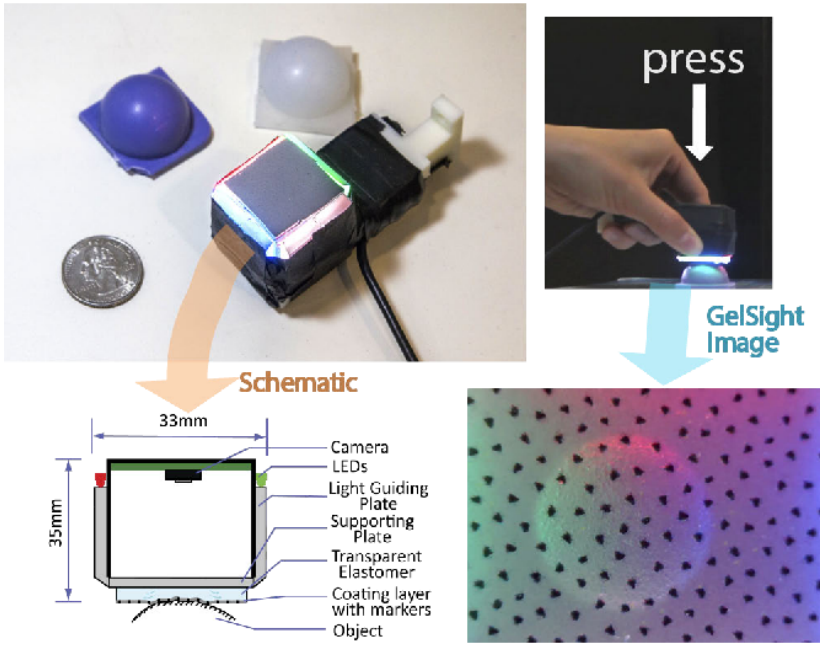
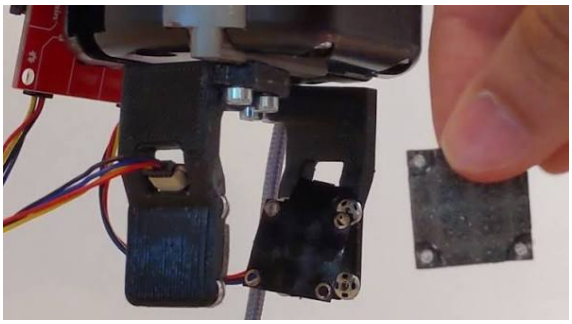


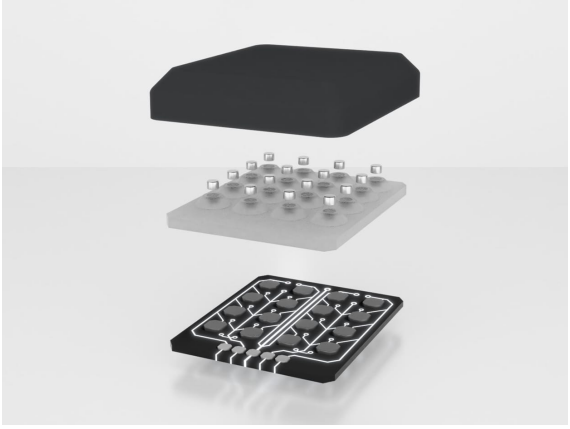
Figure 2.2: GelSight Tactile Sensor.

ReSkin (Figure 2.3a) and uSkin (Figure 2.3b), on the other hand, take a middle-ground approach using magnetic sensing instead of optics. These products feature a thin, replaceable elastomer sheet embedded with magnetic particles, and a magnetometer measures how the surface deforms. They

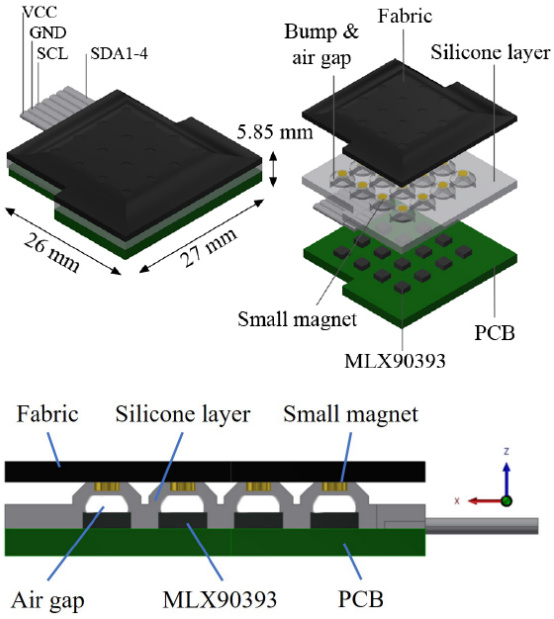
stand out due to being low-cost, durable, and easily replaceable tactile skins [26, 24].



(a) ReSkin Tactile Sensor.



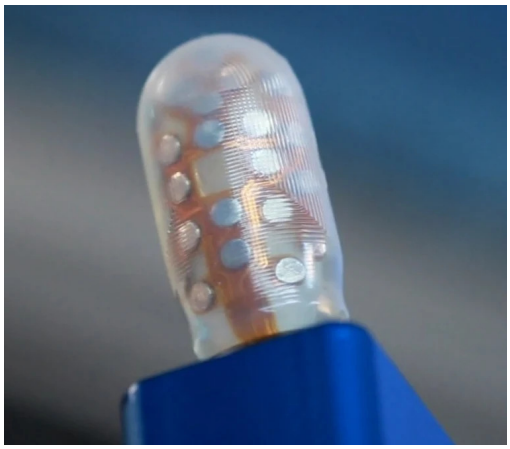
(b) uSkin Tactile Sensor.



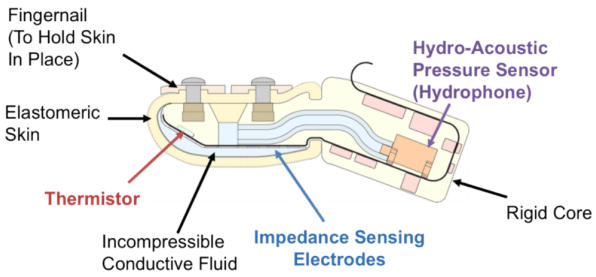
(c) Principal of operation of ReSkin and uSkin.

Figure 2.3: ReSkin and uSkin Tactile Sensors

BioTac (Figure 2.4) presents a more sophisticated option. It's a biomimetic fingertip sensor that combines fluid-filled elastomeric skin with a rigid core. Here, the deformation changes the impedance of a set of electrodes, allowing it to detect normal and shear forces, vibrations, and temperature changes all from one fingertip [37, 45]. While it offers a wealth of sensing capabilities, BioTac also comes with a higher price, a bulkier size, and maintenance needs (like fluid sealing) that make it less ideal for compact and low-cost embedded grippers.



(a) BioTac Tactile Sensor.



(b) BioTac Tactile Sensor.

Figure 2.4: BioTac Tactile Sensor.

Recently, tactile sensing has made its way into mainstream industrial gripper technology. For example, Robotiq's TSF-85 fingertips, which can be added to their 2F-85 adaptive gripper (Figure 2.5), use a 28-taxel capacitive array to capture pressure along with measuring vibration and proprioception at the fingertip. This aims to deliver reliable grasping and richer data collection for physical-AI applications [32, 31]. This commercial progress is noteworthy: it shows that fingertip-level, array-based tactile sensing is now considered standard for contact-aware manipulation, and it showcases a design philosophy where the fingertip is a self-contained unit that connects through a standard industrial interface – this is pretty different from the more tightly integrated, single-microcontroller approach we're exploring in this thesis.



Figure 2.5: Robotiq Tactile Sensor.

Academic research on sensorized two-finger grippers shows a similar range. For example,

TIAGo's parallel-jaw gripper has been fitted with load-cell fingertip sensors to create a grasping controller that combines position and force feedback, effectively reducing object movement during grasping [21]. This is conceptually similar to the admittance control discussed in this thesis, but relies on discrete load cells rather than a full tactile array. On the high-resolution end, some newer two-finger grippers incorporate multiple GelSight sensors on both fingers to help reconstruct intricate contact geometries for tasks like scooping and separating objects, emphasizing spatial resolution over simplicity, and usually needing a host computer for image processing instead of relying on-device [53, 48].

In this context, the gripper development discussed in this thesis, leans towards the affordable, resistive-array method: using a matrix FSR tactile sensor that's processed locally and in real-time on a microcontroller, instead of opting for a high-resolution optical or fluid-based sensor that needs external processing. This choice balances sensing quality with ease of embedding, cost, and reaction time, and aligns with the overall design goal of keeping the entire sensing-to-actuation pipeline – which consists of the tactile array, ToF sensor, and the admittance controller – on a single, low-power embedded platform, setting this gripper apart from vision-based tactile sensors and setups that offload tactile processing to a host computer.

2.2 Proximity/ToF Sensing for Pre-Grasp Planning

Detecting the space between a gripper and the object it aims to grasp before making contact – often referred to as "pretouch" or proximity sensing – fills a gap that neither long-range vision nor tactile sensing can adequately address individually. While vision and depth sensors are typically used to identify objects before grasping, they can struggle with issues like occlusion, inaccurate readings, and problems resulting from shiny or clear surfaces. On the other hand, tactile sensing only kicks in after contact has been made, which means the object could already be moved or altered by that point [29]. Pretouch sensors aim to bridge this gap: they're non-contact sensors that work at a shorter range than standard optical devices but longer than tactile sensors. They're often integrated right into the end effector, helping to sidestep issues related to occlusion that can affect cameras mounted externally when the robot gets close to the object. The VL53L7CX, a multi-zone Time-of-Flight sensor used in this project (see Figure 2.6), falls into the category of short-range proximity

sensors. It features an 8x8 grid of independent ranging zones across a 60°x60° field of view, giving an absolute distance for each zone regardless of the object's color or reflectivity, at update rates reaching up to 60 Hz and distances of up to 4 meters [46, 44]. This design essentially makes it behave more like a low-resolution, solid-state 3D scanner rather than just a single-point rangefinder, setting it apart from simpler binary or single-zone proximity sensors, but it's also lighter, cheaper, and less power-hungry than a full RGB-D camera.



Figure 2.6: VL53L7CX Multi-Zone ToF Sensor.

The importance of proximity/ToF sensing in pre-grasp planning has been highlighted in various studies. For example, pretouch data can help fill in gaps in point clouds created by depth cameras – like adding data from proximity sensors to complete parts of an object's outline that are occluded or transparent before trying to plan a grasp. It's also useful in closed-loop pre-grasp control, where an initial grasp region is estimated with a depth camera and then fine-tuned by using a proximity sensor to trace the edge of the object. This allows for grasps that can't be reliably achieved using just vision or touch, such as picking up thin, flexible tissue paper [9]. Proximity-and-contact sensors built into the fingertip have also been shown to enhance pre-grasp alignment, detect when contact happens, and match locally sensed points against an external RGB-D point cloud. This again emphasizes that proximity sensing works best as a complement to longer-range vision, rather than as a substitute [28].

This collection of research supports the idea that proximity/ToF sensing is a better fit for embedded systems compared to a full RGB-D setup for the pre-grasp function outlined in this thesis. RGB-D sensors produce dense, pixel-by-pixel depth maps that can be pretty noisy and need a fair bit of processing – like filtering, building point clouds, and estimating grasp regions – to be effective. Plus, the quality of depth data usually drops off at close ranges, which is precisely where pre-grasp actions are happening [30]. A multi-zone ToF sensor, however, simply delivers a small, fixed-size array of distances for each zone directly via I^2C and performs the averaging calculations internally. This makes it manageable to read and interpret on a smaller microcontroller without needing extra processing resources. This approach aligns well with the overall philosophy of embedded processing for the gripper developed in this thesis: instead of adding a depth camera that would complicate things, pre-grasp knowledge – like object presence, rough distance, and approach guidance – comes from a sensor whose data can be processed locally on the same microcontroller, together with the FSR tactile array, all within a real-time control loop.

2.3 Admittance / Impedance Control for Grasping

Admittance and impedance control are the two classical frameworks for regulating the dynamic interaction between a robot and its environment, and together they form the theoretical basis for the compliant grasping controller developed in this thesis. Both originate from Hogan's formulation of impedance control, which proposes controlling the dynamic relationship between motion and force at the interaction port, rather than controlling position or force in isolation [13]. The two frameworks are functional inverses of one another: impedance control computes the force to apply given a measured or commanded motion, while admittance control computes the motion to command given a measured interaction force [16, 15]. Admittance control is generally the more practical choice for platforms, such as the gripper developed in this thesis, that are built around a position-controlled actuator rather than a torque-controlled one: because the controller only needs to convert a measured force into a desired position or velocity reference for an existing motion control loop, it can be implemented without requiring high-fidelity torque control at the joint level [25]. In both cases, the desired dynamic relationship between force and motion is typically defined by a virtual mass-spring-damper system, whose parameters set how compliant or rigid the interaction

feels [36, 4].

Admittance control has a long history of use in compliant motion and force-guided tasks such as assembly and peg-in-hole insertion [36], and it has also been applied directly to gripper-level grasping. In space-robotics applications, variable-damping admittance control has been used to let a manipulator correct positioning deviations and grasp objects reliably based on the contact force measured during the grasp itself, rather than relying solely on a pre-planned trajectory [1]. Closer to the scale and actuation style relevant to this thesis, admittance control has recently been applied directly at the fingertip of an underactuated gripper to regulate pinch force during contact, achieving sub-half-Newton force-tracking accuracy and enabling simple reactive grasping behaviors such as blind object search and object-loss detection purely from force feedback [12]. This class of results supports the core premise of this thesis: that a real-time admittance loop, closed using distributed tactile feedback rather than a single force sensor, can provide compliant, force-aware grasping without requiring a fully torque-controlled robotic hand.

The main limitation of this classical approach, and the one that motivates the reinforcement-learning direction pursued later in this thesis, is that a fixed set of admittance or impedance parameters does not generalize well across objects with different mechanical properties. Because the parameters of the virtual mass-spring-damper model are typically set from engineering experience rather than derived automatically, fixed-gain impedance control tends to perform poorly when the robot encounters conditions or objects it was not tuned for, which can lead to large control errors or unstable contact [7]. In practice, this has pushed the field toward gain-scheduling or object-dependent strategies – for instance, using tactile sensor readings to first classify an object's stiffness and then select the corresponding impedance parameters [7] – which mitigates but does not eliminate the underlying problem: performance still depends on the object falling within a pre-identified stiffness class, and each new class of object may require its own re-tuned gain set [3]. This lack of generalization across heterogeneous, previously unseen object properties is precisely the gap that a learned grasping policy is well positioned to address, and is the motivation for the reinforcement-learning component investigated in the second part of this thesis.

2.4 Reinforcement Learning: Advantages and Limitations

The reason for going after a learned grasping policy, which we touched on at the end of Section 2.3, is rooted in a key advantage of reinforcement learning. Unlike needing a manual control system and a tweaked set of gains for each type of object, an RL policy can, in theory, learn a control strategy simply through interacting with its environment. This means it can figure out how to connect raw or lightly processed sensory data to effective actions without needing a carefully crafted state estimator for every new sensing type. That's why RL is appealing for a system that combines two unique and complementary sensor types on the same fundamental controller, as discussed in Sections 2.1 and 2.2. Instead of manually creating separate contact and pre-grasp models for the FSR and ToF signals, the policy is allowed to learn how to bring those together through experience.

But this strength does come with real challenges, some of which are really important to the choices made later in this thesis. One major issue is sample inefficiency. Even model-free algorithms that manage to integrate experience well across various initial conditions usually need a lot of environment interactions to really come together – way more than what's practical to gather on real hardware [19, 14]. This is exactly why the GPU-parallelized simulation we will discuss next isn't just a nice-to-have, but actually a must-have for training. Another challenge relates to the complexity of designing rewards. To make a long-term manipulation task learnable, we often need multi-term, shaped reward functions, but getting the weights right can be tricky. If one term is too heavily emphasized, it can cause the policy to be shortsighted, preventing it from finding actions that require stepping away from a short-term goal to accomplish the main task [14]. This isn't just a theory for this thesis, either; it's the reason why the reward function in Section 9.2 includes a specific penalty against closing the gripper too soon or too indiscriminately. This was added to counteract the risk of falling into a trap where the system would just keep closing the gripper because of a simple proximity and contact reward combination.

Safety and formal guarantees present another limitation. The admittance controller discussed in Section 2.3 is based on an explicit virtual mass-spring-damper model, where the parameters clearly determine how compliant or rigid the behavior will be, giving the designer a clear understanding of the forces the gripper can exert. But a trained neural network policy doesn't offer those same guarantees. There's nothing in its structure that stops it from commanding unsafe forces or actions

outside the norm, and making sure safety rules are followed is widely seen as a significant hurdle for using reinforcement learning in real hardware [8]. This concern is vital for the ERC Maintenance Task introduced in Chapter 1, where some panel components can't tolerate excessive force or torque, which is one reason the traditional admittance controller remains as a baseline instead of being tossed aside once a learned policy is developed.

Lastly, a trained policy is typically tied to the specific task, reward function, and hardware that it was trained on. Its ability to adapt beyond the conditions it learned in is totally dependent on how representative those training situations were. This limitation is clear in the narrow scope of the domain randomization applied so far in this thesis (Section 8.5), where only the probe's pose is varied, leaving object properties and sensor noise unchanged. When you combine this with how opaque a neural network policy can be compared to an explicit control law, it shows that reinforcement learning shouldn't be seen as a straightforward upgrade over classical control. Instead, it represents a different approach with its own set of trade-offs. The actual performance of this learned policy, compared to the admittance controller we've built and demonstrated here, can only be determined through direct tests on the same hardware, which is what this thesis aims to achieve.

2.5 Digital Twins and Sim-to-Real Transfer in Manipulation

Training a reinforcement learning grasping policy at scale requires a simulation environment capable of generating far more interaction data than could ever be collected on physical hardware. NVIDIA Isaac Gym addressed this by moving both physics simulation and policy training onto the GPU, exchanging data directly between physics buffers and PyTorch tensors without passing through the CPU, which yields two-to-three orders of magnitude speedup over conventional CPU-simulator/GPU-training pipelines and allows thousands of environments to be simulated in parallel on a single GPU [23]. Isaac Lab is presented as the direct successor to Isaac Gym, extending the same GPU-native paradigm to large-scale, multi-modal robot learning: it combines GPU-parallel physics with photorealistic rendering, actuator models, multi-frequency sensor simulation, and built-in domain randomization tools within a single, modular framework [18]. This is the environment used in this thesis to build the virtual model of the gripper and to train the grasping

policy.

Even with high-throughput simulation available, a policy trained purely in simulation does not automatically perform well on real hardware, because of discrepancies between simulated and real dynamics collectively known as the reality gap [5]. Two complementary strategies are used to narrow this gap. Domain randomization varies simulation parameters – object mass, friction, sensor noise, and similar quantities – across training episodes, so that the real world is statistically just another sample from the distribution the policy was trained on, rather than an out-of-distribution condition [43]. System identification instead takes the opposite approach: rather than randomizing broadly around a nominal model, it fits the simulator's parameters directly to measurements taken from the real system, so that the specific simulated instance matches the specific real platform as closely as possible [47]. In practice, the two strategies are often combined: domain randomization provides robustness to residual, unmodeled discrepancies, while system identification narrows the range that needs to be randomized in the first place. This thesis relies primarily on domain randomization, since a full system identification campaign for the gripper's mechanical and sensor parameters is left as future work, as discussed in Chapter 10.

Simulating rigid-body kinematics and dynamics is now a comparatively mature capability, but simulating tactile sensors remains a much less developed area, and the choice of tactile simulation strategy materially affects how well a policy trained on simulated tactile data will transfer to a real FSR array. Broadly, three families of approaches exist in the literature. The first relies on optical rendering, generating a synthetic sensor image directly from the geometry of the contact rather than computing physical contact quantities – this is the approach taken by TACTO, an open-source simulator that renders tactile images for vision-based sensors such as GelSight by combining a rigid-body physics engine with an image renderer [41, 38]. The second relies on penalty-based contact models, in which each taxel is represented as a contact point on the sensor surface and the per-taxel force is computed from the local penetration depth and relative velocity using a spring-damper penalty law – for instance, a recent matrix-pressure tactile sensor simulation models each taxel as a dense contact point, queries the signed distance field of the contacted object at each physics step, and applies a Kelvin-Voigt spring-and-damper model to obtain per-taxel force from penetration depth and normal velocity [11]; a similar taxel-grid, contact-point approach has also been used to simulate piezoresistive tactile arrays directly in a GPU-parallel physics engine [40].

The third family relies on learned sensor models, which fit the mapping from simulated contact geometry to sensor output directly from real sensor data rather than deriving it analytically – Taxim, for example, is described as an example-based simulation model that learns the optical response of a GelSight sensor from real contact data, improving realism over purely geometric rendering approaches [33].

Because the tactile sensor used in this thesis is a resistive FSR matrix, its simulation is naturally closer to the second family.This is consistent with recent work showing that matrix-based pressure sensors, unlike optical tactile sensors, produce structured, lower-dimensional contact patterns that are comparatively more tractable to simulate and align with real sensor data [11], and it is the strategy adopted for the tactile array in the virtual model developed in Chapter 8.

Finally, the reward function used to train the grasping policy is itself an active design choice in the literature. Reward formulations for learned grasping typically combine several terms – for example, a linear combination of the change in the number of contact points between the hand and the object, the change in distance between the hand and the object, an action-magnitude penalty, a fingertip-spread term, and a final drop-test success signal has been used to shape learning toward stable, reachable grasps [22]. Purely sparse, binary success rewards have also been used successfully at scale, sometimes combined with a small per-timestep penalty specifically to encourage the policy to grasp more quickly rather than needlessly prolonging the episode [19]. This range of reward-shaping choices – success, force regulation, stability, and time-to-grasp – directly informs the reward design adopted for the grasping policy trained in Chapter 9 of this thesis.

2.6 Gap Identified

The literature reviewed in this chapter shows that tactile sensing, proximity/ToF-assisted grasping, admittance control, and sim-to-real reinforcement learning have each individually reached a mature level of development. Low-cost resistive tactile arrays and admittance control are well established for compliant, contact-rich grasping [35, 12], multi-zone ToF sensors provide a lightweight, embedded-friendly complement to tactile sensing for pre-grasp planning [9], and GPU-parallel simulation frameworks now make it practical to train reinforcement learning grasping policies at scale [23, 18].

What is largely missing is the integration of these strands into a single pipeline. Existing

sensorized two-finger grippers with classical control typically stop at hardware and admittance or force control, without extending into a learning pipeline built on the same platform [21, 12]. Conversely, sim-to-real tactile learning work tends to focus on the tactile simulation problem in isolation, developing and validating simulated sensor models such as GelSight-class renderers or contact-point tactile arrays without pairing them with a classical embedded controller running on the same physical device [41, 11, 33]. As a result, the tactile and proximity sensors used to close a classical control loop on real hardware are, in most published systems, not the same sensors reproduced in simulation to train a learned policy.

This thesis attempts to address this gap. It develops a single physical gripper, sensorized with the same FSR tactile array and ToF sensor used throughout, first to close a classical embedded admittance control loop and then, from the same hardware, to build a sensor-faithful virtual model used to train a reinforcement learning grasping policy. This is the platform-level contribution motivated by the gap identified above, and it is what ultimately makes a fair, apples-to-apples comparison between classical and learned control possible – the centerpiece of the long-term research plan this thesis lays the groundwork for, as outlined as last question in section 1.3.

3. Conceptual Design

This chapter addresses the conceptual design of the end-effector, describing the high-level ideas and solutions that will need to be developed further. The aim is therefore to provide specific guidelines for the detailed design phase.

3.1 Design Requirements

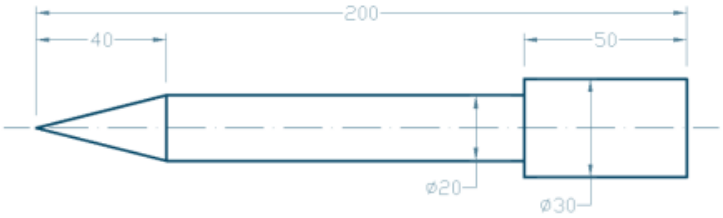
The mechanical design of the gripper is driven directly by the two ERC tasks it is intended to perform, the Maintenance Task and the Probing Task, together with the sensing and control requirements introduced in Chapter 1. Taken together, these tasks translate into a coherent set of requirements spanning payload, stroke, actuation, sensor placement, and compliance, which are discussed in this section.

The ERC probes are cylindrical aluminum bodies roughly 200 mm long, with a tapered tip and a gripping section on the order of 20-40 mm in diameter, while the Maintenance panel requires actuating lever and rotary switches, inserting an IEC320 C14 plug into a socket, and placing a metal plate on an electromagnet (Figure 3.1). The design does not therefore need to support a large payload; the more demanding requirement is force regulation rather than force capacity, since several panel elements are explicitly sensitive to forces and torques exceeding their operational limits and must not be damaged during manipulation. The gripper must consequently be able to apply a small, well-controlled grasping force at the low end of its range, while still retaining enough force authority at the high end to grip and lift a probe securely. This same variety of target objects also sets the requirement on finger stroke, which must span the full range of sizes encountered across both tasks – from the narrowest gripping section of a probe, on the order of 20 mm, up to the widest panel element to be manipulated – with enough margin to approach and center on a target without requiring pixel-perfect positioning from the rover's arm. The stroke requirement is therefore set by the largest expected grasp width in other tasks.

Meeting these force and stroke requirements in turn constrains the actuation mechanism, whose motor and transmission must satisfy three requirements simultaneously. It must be precisely



(a) Maintenance task panel



(b) Probe task probe

Figure 3.1: Target objects for the two ERC tasks. The Maintenance task panel (a) requires actuating lever and rotary switches, inserting an IEC320 C14 plug into a socket, and placing a metal plate on an electromagnet. The Probing task probe (b) is a cylindrical aluminum body 200 mm long, with a tapered tip and a gripping section on the order of 20-40 mm in diameter.

and repeatably position-controllable, so that the same admittance controller can command small, accurate finger displacements; it must be backdrivable enough to support compliant behavior under the admittance law, instead of fighting external forces rigidly; and it must fit within the size, weight, and power budget of a small end effector mounted on the rover's arm.

Alongside these actuation requirements, the mechanical design must also accommodate both sensing modalities without compromising their signal quality. The FSR tactile matrix must sit directly on the finger's contact surface, since any interposed material would damp or distort the force signal reaching the sensor; at the same time, the sensor requires some form of thin protective layer to survive repeated contact with rigid objects such as the aluminum probes and the panel's metal elements, which introduces a design trade-off between signal fidelity and durability. The ToF sensor, mounted on the palm, requires an unobstructed field of view of the space between the fingers and the approach volume in front of the gripper; the surrounding mechanical structure, finger linkages, and cabling must therefore be routed and shaped so that they do not intrude into the sensor's cone of view or actuation mechanisms.

Finally, because the gripper is controlled via admittance, the mechanical transmission must not itself behave as an unmodeled stiff element between the motor and the fingertip. Backlash, friction, and structural compliance in the finger linkage all distort the relationship between commanded motor motion and actual fingertip motion, and excessive amounts of any of the three would undermine the admittance controller's ability to render a predictable, tunable virtual mass-spring-damper behavior at the contact point. These properties cannot be eliminated entirely, but they must be small enough – or characterized precisely enough – to be compensated for in the control law described in Chapter 6.4.

3.2 Block Diagram

To better understand the overall architecture of the gripper, a block diagram is provided in Figure 3.2. This diagram illustrates the main components and their interactions. Each block represents a functional unit, and the arrows indicate the flow of information and control signals between them.

Power stage: This block represents the power supply and distribution system that provides the

necessary electrical energy to all components of the gripper. The rover's main power source is connected to this stage, which regulates and distributes power to the motor, sensors, and control electronics. Specifically, the rover supplies 24V to the gripper's stepper motor, which in turn provides a 12V output from which the rest of the electronics draws power. Since this voltage is too high to directly power the microcontroller, a series of voltage regulators is used to step it down to 5V and 3.3V. In total, two LDO (Low-Dropout) regulators are present: one converts 12V to 5V to power the analog part of the circuit, limiting current consumption, and the other converts 5V to 3.3V to power the microcontroller and the digital part of the circuit.

Control and communication stage: This block groups the components responsible for processing sensor data, executing the control loop, and communicating with the rest of the rover. At its core is the microcontroller, which runs the embedded admittance control loop, reads and processes the tactile and ToF sensor data, and manages communication with the rover through an integrated CAN controller. The microcontroller communicates with the ToF sensor over I2C to obtain multi-zone ranging data used for pre-grasp behavior, and exposes an SWD programming interface used during development and firmware updates. On the communication side, the CAN controller, embedded in the microcontroller, generates the CAN protocol frames, which are then converted into differential CAN signals by an external CAN transceiver. This transceiver connects the gripper to the rover's controller over the shared CAN bus, allowing the gripper to receive high-level commands and stream telemetry data without requiring a dedicated wired interface for each signal.

Tactile sensing chain: This block represents the analog signal path used to acquire data from the FSR tactile matrices. Since two FSR matrices are used, each with its own rows and columns, two multiplexers are employed to sequentially select which taxel is being read, avoiding the need for one dedicated ADC channel per taxel. The selected taxel's resistance is converted into a voltage signal by a voltage divider, whose output is then conditioned by a buffer stage to preserve signal integrity and prevent loading effects before reaching the analog-to-digital conversion stage. Before the signal reaches the microcontroller's ADC input, it passes through an ADC protection block, which clamps the voltage within safe limits and protects the microcontroller from transients or out-of-range readings. The microcontroller then reads the conditioned signal through its internal ADC and, in coordination with the multiplexer's row/column selection, reconstructs the full pressure map of both FSR matrices.

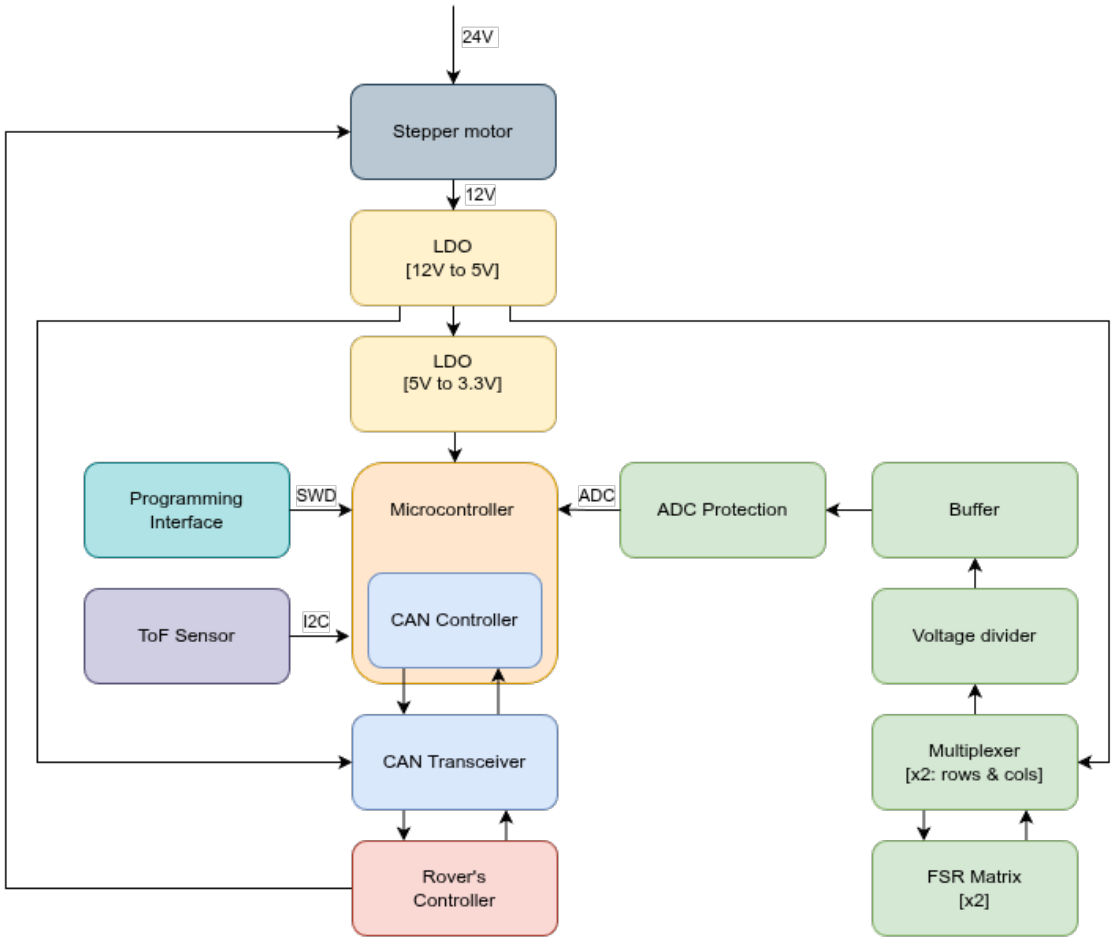


Figure 3.2: System block diagram

3.3 Functional Decomposition

Before detailing the physical implementation of the gripper, it is useful to describe the system in purely functional terms, independently of the specific components used to realize it. This functional view complements the block diagram presented in Section 3.2: while that diagram describes *what physical components* are present and how they are wired together, the functional decomposition describes *what the system must do*, breaking the overall behavior down into a hierarchy of functions and sub-functions. This separation is a standard step in systems engineering, and it is useful here because several sub-functions identified below are shared across otherwise different physical blocks, while others map onto more than one component acting together.

The main function of the system is presented in Figure 3.3: *grasp and manipulate objects of*

interest. This function is the top-level goal of the system, and it is the one that will be used to evaluate the success of the design.

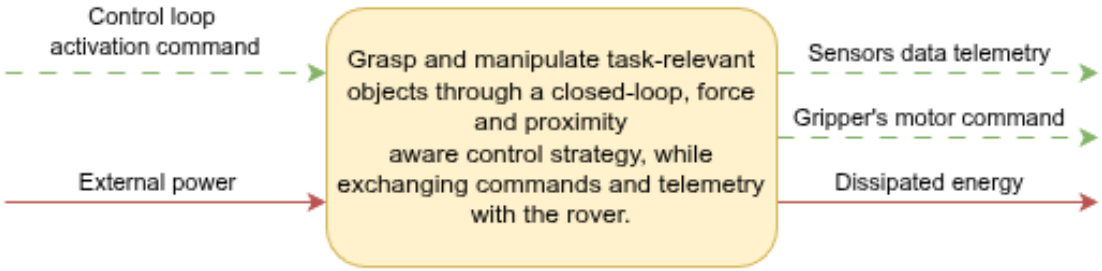


Figure 3.3: Gripper's principal function

This top-level function cannot be implemented directly; it must be decomposed into a set of cooperating sub-functions, grouped here into five first-level branches: *sense*, *process and control*, *actuate*, *communicate*, and *supply power*. Each branch is further decomposed in Figure 3.4 and discussed below.

Sense: This branch covers all functions responsible for acquiring information about the environment and the state of the gripper itself.

- *Sense pre-contact geometry*: detect the presence, approximate distance, and coarse shape of an object before physical contact occurs, using the multi-zone ToF sensor.
- *Sense contact force distribution*: measure the normal force distribution over each fingertip's contact surface during and after contact, using the FSR tactile matrices.
- *Sense finger position*: infer the current finger opening/closing state, needed both for the admittance control law and for coordinating with the rover's higher-level task logic.

Process and control: This branch covers the computational core of the system, responsible for turning raw sensor data into a coherent control action.

- *Acquire and condition sensor signals*: multiplex, convert, and filter the raw tactile signals, and read the ranging data from the ToF sensor over its digital interface.
- *Compute the control law*: execute the embedded admittance control algorithm, converting the estimated contact force into a commanded finger position or velocity.

- *Manage communication protocol*: encode/decode CAN frames, arbitrate between real-time control traffic and telemetry/logging traffic without violating the control loop's timing budget.

Actuate: This branch covers all functions needed to physically move the fingers according to the computed control command.

- *Convert the control command into a motor command*: translate the desired finger position/velocity into the appropriate driving signal for the stepper motor.
- *Transmit mechanical motion*: convert the motor's rotary output into the fingers' linear closing/opening motion through the mechanical transmission, while remaining backdrivable enough to support compliant behavior.

Communicate: This branch covers the exchange of information between the gripper and the rest of the rover.

- *Receive high-level commands* from the rover's controller (e.g., open/close requests, target force or position setpoints).
- *Transmit telemetry and logging data* (tactile maps, ToF readings, control-loop status) back to the rover, without disrupting the real-time control function.

Supply power: This branch covers the functions needed to make the rover's raw power usable by every other branch.

- *Step down and regulate voltage* from the rover-supplied and motor-derived voltage levels to the levels required by the analog and digital electronics.
- *Distribute regulated power* to the microcontroller, sensors, and communication interfaces while limiting current draw and protecting against transients.

This functional breakdown will be used in the following sections as the reference structure against which the physical architecture described in Section 3.2 is justified: each physical block introduced there can be traced back to one or more of the sub-functions identified here, ensuring that no required function is left unimplemented and that no physical component exists without a corresponding functional justification.

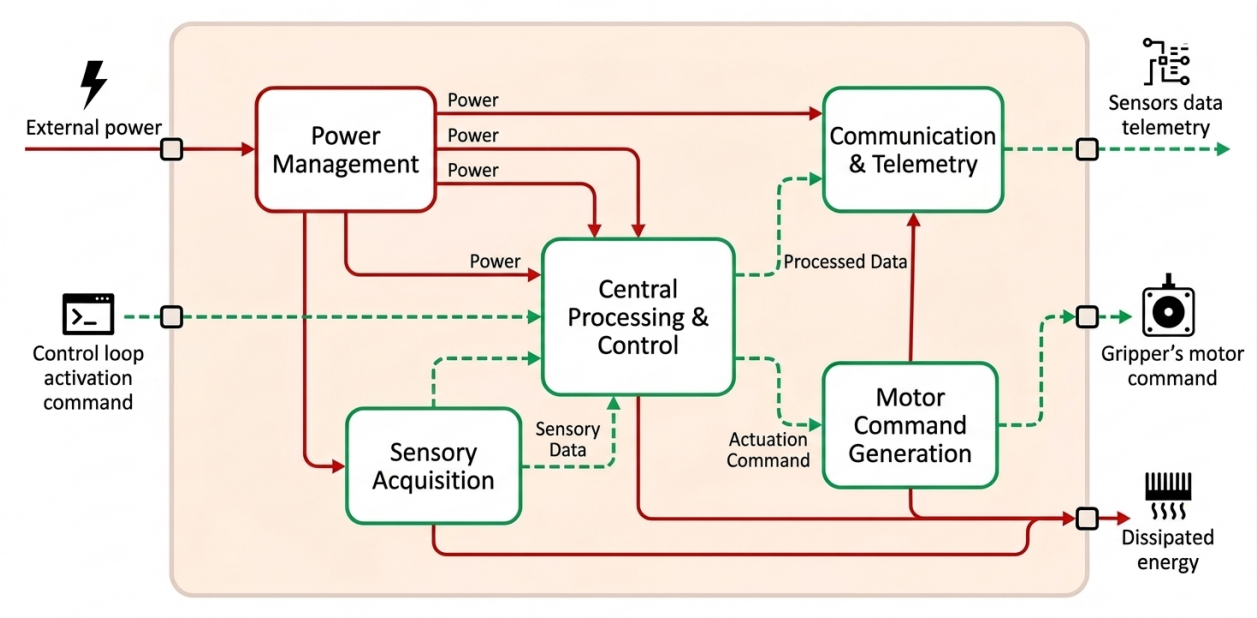


Figure 3.4: Functional decomposition

3.4 Concept Generation - Electronic

Each finger of the gripper is equipped with a matrix-type FSR tactile sensor, meaning that the sensing surface is not a single force sensor but a grid of individual pressure-sensitive cells, or taxels, arranged in rows and columns. Reading every taxel simultaneously and continuously would in principle give the fastest possible update rate, but it would also require one dedicated analog signal path, and ideally one ADC channel, per taxel. For a matrix with tens of cells per finger, and two fingers to instrument, this quickly becomes impractical on a microcontroller with a limited number of ADC channels and a limited pin count, and it is precisely the kind of design choice that motivates the low-cost, embedded-friendly sensing approach discussed in Section 2.1. The conceptual solution adopted here is therefore not to read all taxels in parallel, but to read them one at a time, in a fast, repeating sequence, a scheme generally referred to as row-column scanning or matrix multiplexing.

Conceptually, the taxel grid can be thought of as a set of rows and a set of columns, with a resistive tactile element sitting at each row-column intersection. To read a single taxel, one row is selected and driven with a known reference condition, while one column is selected and connected to the sensing electronics; the resistance of the taxel at that specific row-column intersection then

determines the electrical signal seen by the analog front-end. By selecting rows and columns one pair at a time, the same analog signal path can be reused to sequentially read every taxel in the matrix, rather than requiring a separate physical channel for each one. This is the role played by the multiplexer stage introduced in the block diagram of Section 3.2: rather than a single multiplexer, two are used in practice, one dedicated to selecting the active row and one to selecting the active column, so that the pair of selection signals together address exactly one taxel of the matricies at any given instant.

Once a specific taxel is addressed in this way, its resistance must still be converted into a signal the microcontroller's ADC can interpret, since the ADC measures voltage, not resistance directly. This conversion is done conceptually through a voltage divider: the selected taxel's variable resistance is placed in series with a second, fixed resistance, and a known reference voltage is applied across the series combination. As the force applied to the taxel changes, its resistance changes, and so does the fraction of the reference voltage that appears at the midpoint of the divider – this midpoint voltage is the signal that is ultimately digitized. Because this voltage can be sensitive to loading effects from the downstream circuitry, and because it needs to be protected from any conditions outside the microcontroller's safe input range, a buffering stage and a protection stage are conceptually interposed between the voltage divider and the ADC input, as already indicated in the block diagram.

The overall tactile reading process for each finger therefore consists of the microcontroller sweeping through all row-column combinations of its matrix in sequence, taking one ADC sample per combination, and reconstructing the full pressure map of the finger from these individual readings. Repeating this scan continuously and fast enough yields a sequence of pressure maps at a usable update rate, which is later used both by the tactile signal processing described in Chapter 6.2 and by the sensor-faithful simulation of the FSR array described in Chapter 8.2. The specific scanning frequency, multiplexer part numbers, and timing budget needed to keep this process compatible with the real-time constraints of the admittance control loop are addressed together with the rest of the electronic implementation in Chapter 5.

3.5 Concept Generation - Mechanics

At the conceptual level, the mechanical architecture of the gripper follows directly from the requirements identified in Section 3.1: a compact, single-actuator finger-closing mechanism, compatible with the sensor placement constraints for the tactile matrices and the ToF sensor, and capable of being scaled up in force output without a substantial redesign. This section discusses the concepts adopted to satisfy these requirements; the detailed mechanical design, including component sizing and CAD implementation, is addressed separately in the Chapter 4 dedicated to the mechanical design of the gripper.

The rotary motion produced by the electric motor is converted into the linear opening and closing motion of the fingers through a rack-and-pinion mechanism. A pinion gear is keyed onto the motor's output shaft, and each finger integrates a linear rack that meshes with this pinion on opposite sides, so that as the pinion rotates, the two racks – and therefore the two fingers – are driven symmetrically in opposite linear directions. A single rotary actuator is consequently sufficient to open and close both fingers simultaneously, which reduces the gripper to a single degree of freedom. This is a deliberate simplification: it leaves a single motor to drive and a single position/force variable for the admittance controller described in Chapter 5 to regulate, rather than two independently actuated fingers that would need to be coordinated with one another.

Consistent with the sensor placement constraints discussed in Section 3.1, the ToF sensor is located on the body of the end effector, centered between the two fingers. This central position gives the sensor a symmetric, unobstructed field of view of the approach volume directly in front of the gripper, along the same axis the fingers close on, which is the region relevant for pre-grasp object detection and coarse localization; placing the sensor on the palm rather than on either finger also decouples its field of view from the finger's motion, so that the ToF readings remain consistent regardless of the current finger opening.

A further architectural choice at the concept stage concerns how the sensing and control electronics are physically and electrically integrated with the actuation stage. Rather than treating the custom PCB as an independent subsystem with its own power input and its own communication interface toward the rover, the board is conceived as an add-on that connects directly to the commercial closed-loop stepper motor driver already required to actuate the motor, for both power and

communication. The driver's 12 V output, already used internally by the motor stage, is reused to supply the sensing and control electronics rather than routing a separate power line to the PCB, and the same CAN bus used by the driver to communicate with the rover's controller is shared with the microcontroller's CAN interface rather than duplicated. This is what allows the gripper to remain a compact, self-contained end effector with a single electrical and communication interface toward the rest of the rover, instead of requiring separate cabling for the driver and for the sensing electronics.

The target grasping force for the gripper is set at approximately 2 kg-force at the fingertips, a figure large enough that it is unlikely to be reached directly by the selected motor without some form of mechanical reduction. Rather than committing at the concept stage to a specific gear ratio, or to a motor sized to deliver the full target force without any reduction, the gripper is conceived so that a gearbox can later be introduced between the motor and the pinion without requiring a substantial redesign of the rest of the mechanism. The rack-and-pinion stage, the finger geometry, and the overall envelope of the end effector are therefore kept independent of the specific reduction ratio ultimately required, with the reduction stage treated as a drop-in addition at the motor interface rather than as a constraint that reshapes the finger or transmission design. This keeps the concept flexible with respect to the final force requirement, which is resolved quantitatively, together with the motor and gear ratio selection, in the detailed mechanical design.

4. Detailed mechanical design

In this chapter, the detailed mechanical design of the system is presented. The design process includes the selection of materials, the design of mechanical components, and the integration of these components into a cohesive system. Also, the cable routing and the mechanical assembly of the system are discussed. The design is based on the concepts generated in Chapter 3.5 and aims to meet the functional requirements outlined in Chapter 3.1.

4.1 Manufacturing Process and Material Selection

The gripper's shell components – the main body and the two front covers that enclose the finger mechanism – are manufactured using Fused Deposition Modeling (FDM) 3D printing in PLA (Polylactic Acid). This choice follows directly from the trade-off between rigidity, weight, and manufacturability discussed at the concept stage in Section 3.5. PLA is inexpensive, dimensionally stable, and easy to print reliably even for the organically shaped, non-prismatic surfaces that characterize the gripper's shell, while FDM printing allows the whole assembly to be iterated quickly during development, in line with the design-and-implementation scope of this thesis discussed in Section 1.

The one structural exception to this is the central mounting plate, shown in yellow in Figures 4.1 and 4.4, which is machined from carbon fiber rather than 3D-printed. Because this plate is the load-bearing interface between the motor, the pinion-and-rack transmission, and the outer shell – and, as discussed in Section 4.2, it is also sandwiched directly between two of the printed shell halves – it is subjected to higher and more concentrated mechanical stress than the surrounding shell geometry, particularly around the motor shaft bore and the linear-guide mounting points. Carbon fiber was chosen for this component specifically for its higher stiffness-to-weight ratio compared to PLA, ensuring that the plate does not flex appreciably under load and thereby preserving the alignment of the pinion and linear guides described in Section 4.5, which is important for the compliance requirements identified in Section 3.1.

4.2 Overall Structural Assembly

The gripper's external structure is built from three printed shell components and the carbon-fiber plate introduced above, assembled as shown in Figure 4.1. The main body (shown in cyan) forms the rear and lateral structure of the end effector, incorporating the mounting flange and the motor bore. The carbon-fiber plate (yellow) is sandwiched directly against this main body, and the two remaining printed shells – one covering the front-upper section of the mechanism (pink) and one covering the front-lower section (green) – are then fastened on top, clamping the plate firmly between themselves and the main body. This layered assembly, rather than a single continuous shell as might otherwise be assumed, is what allows the plate to serve simultaneously as a structural stiffener for the transmission and as a mounting reference that the two front covers are aligned and fastened against.

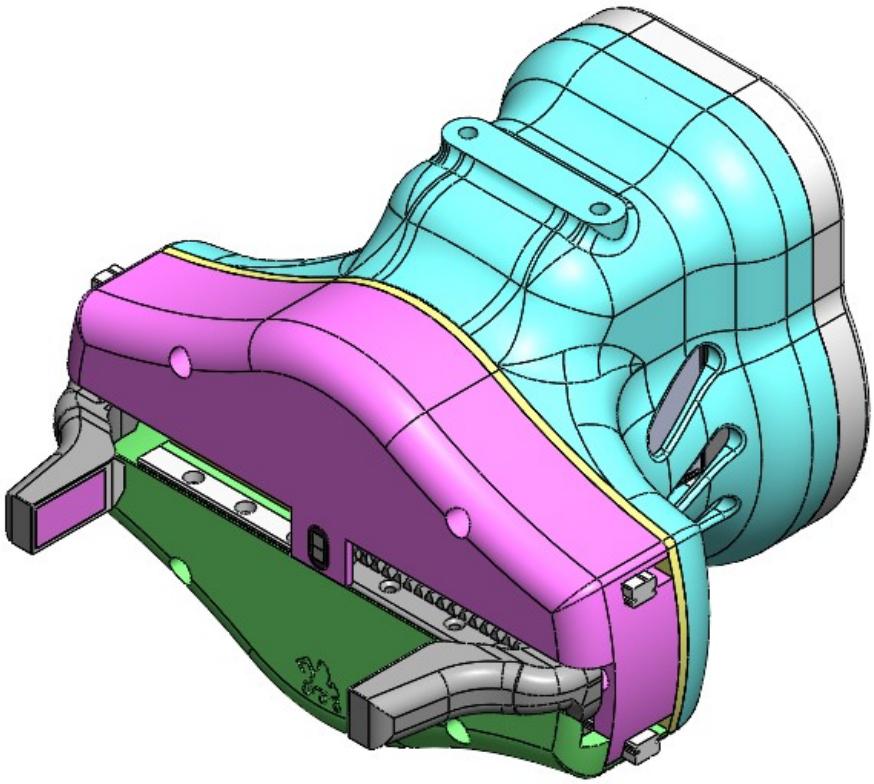


Figure 4.1: Exploded-color view of the end effector shell assembly: the cyan main body and the pink and green front covers are fastened together with the carbon-fiber plate (yellow) sandwiched between them.

On the front-facing side of the assembly, the slotted channels formed by the pink and green covers guide the linear motion of each finger as it opens and closes, constraining it to move along the direction defined by the rack-and-pinion transmission described in Section 4.5, while shielding the internal mechanism – the pinion, the racks, and the linear guides – from dust and incidental contact during operation. The flat, purple-highlighted window visible on the front cover is the mounting location for the palm-centered ToF sensor introduced in Section 2.2: positioned between the two finger channels, this window gives the sensor the clear, unobstructed field of view illustrated in Figure 4.2, without any part of the shell or fingers intruding into its 60°×60° cone even when the fingers are fully open.

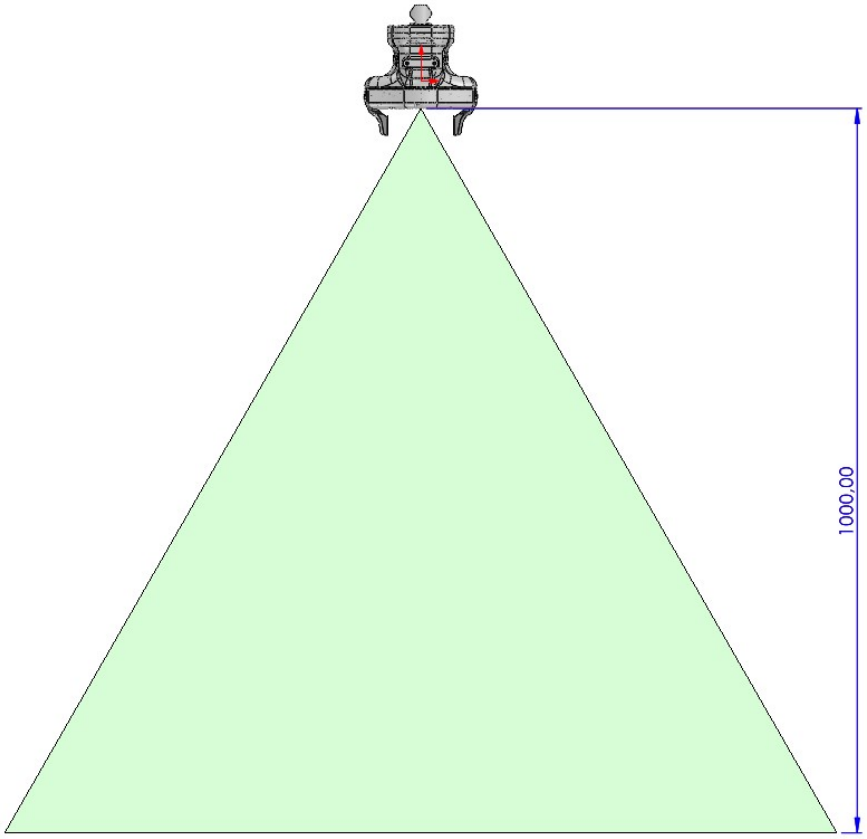


Figure 4.2: Top view of the end effector showing the unobstructed 60°×60° field of view of the palm-mounted ToF sensor, extended here to a 1000 mm range for illustration.

4.3 Cable Routing

Because the tactile sensing chain described in Chapter 5.4 requires signal wires to run from each finger's FSR matrix, and from the palm-mounted ToF sensor, back to the electronics mounted behind the motor, the carbon-fiber plate also serves a second, non-structural function: it integrates dedicated channels along which these sensor cables are routed. Rather than running the FSR and ToF wiring loosely through the internal cavity of the shell, where it could interfere with the moving racks and carriages described in Section 4.5, the cables are guided through slots machined directly into the plate, keeping them mechanically separated from the transmission and consistently routed from each sensor toward an exit point near the motor. From there, all sensor cables converge on the PCB mounted on the rear of the assembly, directly behind the motor, which is where the tactile and ToF reading circuits described in Chapter 5 are physically located.

4.4 Rear Interface and Robotic-Arm Mount

The rear of the main body integrates the mechanical interface to the rover's robotic arm, in the form of a quick-release mount matching the corresponding fixture on the arm's end joint (Figure 4.3). This interface allows the gripper to be attached to, and detached from, the arm without tools, which is a practical necessity given the design-and-implementation scope of this thesis. The gripper is expected to be swapped, re-mounted, and iterated on repeatedly during development and testing, and a tool-less quick-release connection avoids having to disturb the arm's other wiring or fasteners each time. A central hole through the quick-release attachment provides the sole cable pass-through between the arm and the end effector, and is sized to carry both the power and CAN communication lines for the motor driver described in Section 5.1, keeping the gripper's only external electrical connection to the rest of the rover concentrated at this single interface point.

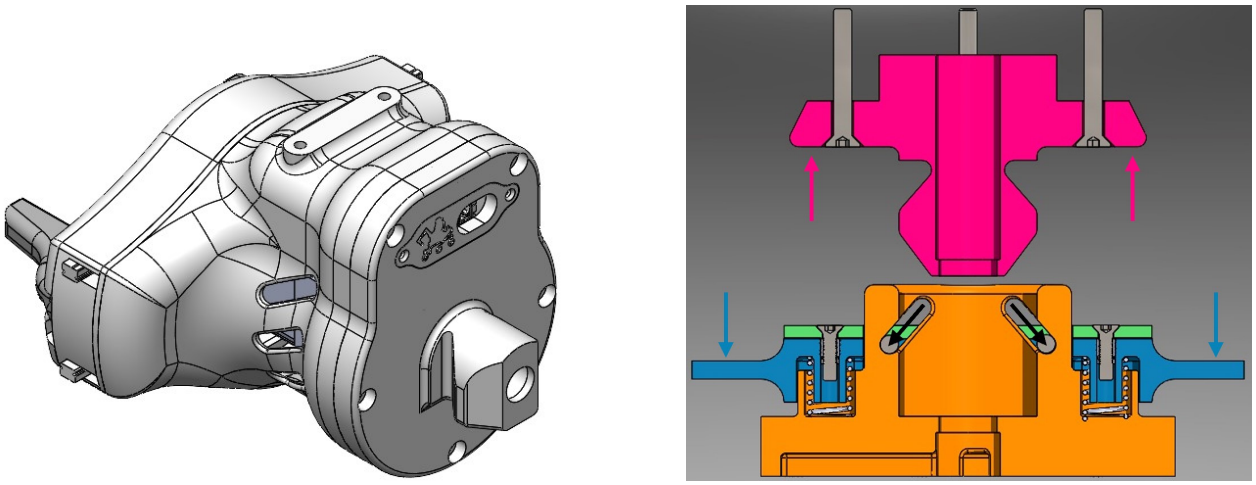


Figure 4.3: (a) Rear-isometric view of the end effector assembly, showing the mounting flange and the motor-side interface. (b) Quick-release mechanism in operation. The gripper can be attached to or detached from the robotic arm without tools, facilitating rapid iteration and testing during development.

4.5 Motion Transmission: Pinion, Racks, and Linear Guides

The internal mechanism that converts the motor's rotary output into the fingers' linear motion is shown in Figure 4.4, mounted directly on the carbon-fiber plate introduced in Section 4.1. A pinion gear is keyed onto the stepper motor's output shaft and sits at the center of the mechanism, meshing simultaneously with two linear racks positioned on opposite sides of it. Each rack is not free-floating but is instead mounted on its own linear guide carriage, which rides along a fixed linear rail fastened directly to the shell. This use of a dedicated linear guide per rack, rather than relying on the rack-pinion mesh alone to constrain the finger's trajectory, directly addresses the compliance requirements identified in Section 3.1: the linear guide takes up the lateral and off-axis loads that would otherwise be reacted by the gear teeth themselves, keeping the mesh engagement clean and repeatable and minimizing the backlash and structural compliance.

As the pinion rotates, the two racks are driven in opposite linear directions along their respective guides, so that the two fingers open and close symmetrically about the assembly's central axis from a single rotary actuator, exactly as described conceptually in Section 3.5. Each finger is rigidly fastened to the top of its rack's carriage, extending upward and outward from the mechanism to

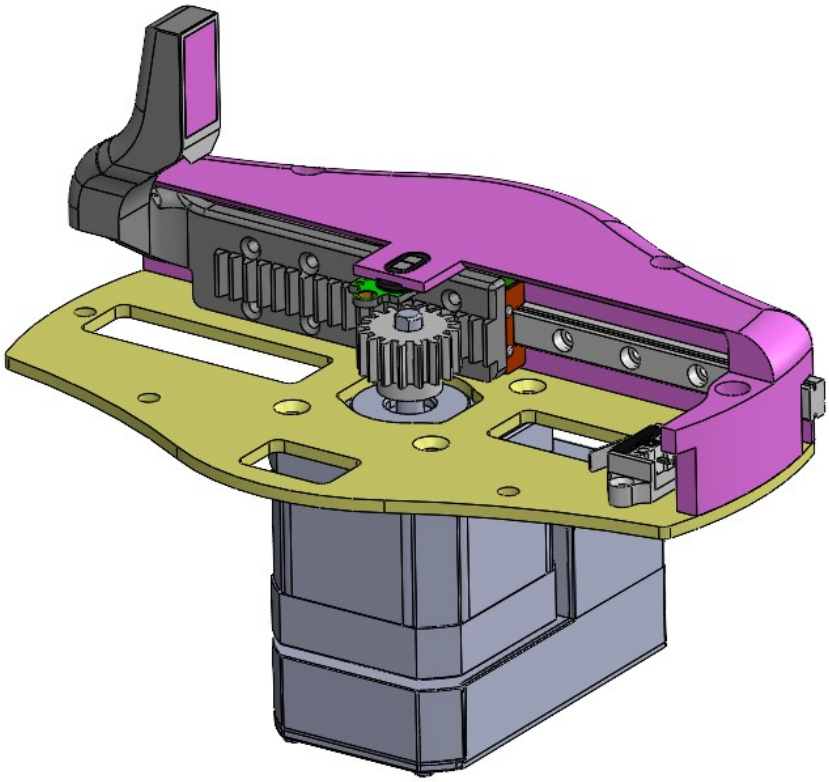


Figure 4.4: Internal rack-and-pinion transmission mounted on the carbon-fiber plate (yellow): the motor drives a central pinion that engages two racks, each mounted on its own linear-guide carriage.

form the actual contact surface that engages the object, as visible in Figure 4.4 for one of the two symmetric finger-rack-carriage sub-assemblies. This arrangement reduces the entire gripper to the single degree of freedom already discussed in Section 3.5, while the pinion-shaft interface is left deliberately accessible so that a reduction gearbox can be inserted between the motor and the pinion at a later stage, per the force-scalability reasoning also introduced in that section, without requiring any change to the plate, racks, carriages, or fingers themselves.

5. Detailed electronic design

This chapter describes the electronic design of the system, covering component selection, circuit topology, and hardware integration. The discussion traces the development process from early-stage prototyping to the layout of a custom PCB, developed for seamless mechanical integration. The objective is to illustrate the critical role of the electronic system in achieving target performance and functionality.

5.1 Motor

The gripper's finger-closing actuator is driven by an Auxind FD1.1GA integrated stepper motor driver, mounted directly on the end-shield of a NEMA 17 stepper motor (Figure 5.1). This is the same motor-driver combination already adopted throughout ProjectRED's robotic arm, and its selection for the gripper follows directly from the fact that reusing an actuator already validated and integrated elsewhere on the rover reduces the number of distinct components the team has to source, characterize, and maintain firmware and configuration for. It also keeps the gripper's actuation interface – power, communication protocol, and command set – identical to that of every other joint on the arm. This consistency is particularly valuable given the compact, add-on integration strategy discussed in Section 3.4, since the gripper's electronics are designed to plug directly onto the driver.



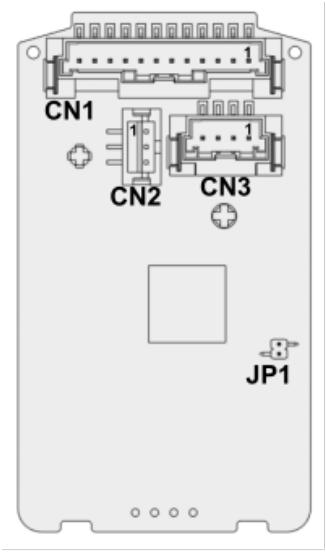
Figure 5.1: FD1.1G applied to NEMA 17

Beyond this standardization argument, the FD1.1GA is also well suited to the gripper on its own technical merits (reported in Figure 5.2).

	Range			Unit
	Min	Typ	Max	
Supply voltage	24		80	V_{DC}
	Note: minimum power supply output capacitor 470 µF *Driver maximum allowed voltage is limited by the motor rated voltage*			
Supply current			3	A
	Note: depends upon power supply voltage, configured motor current, speed and load			
Motor current	500		4 500	mA_P/phase
	Note: Maximum and minimum current are configurable (limited to internal predefined value)			
Angular resolution	400	12 800	204 800	µstep / revolution
Stepper motors	Bi-phase; 4, 6 and 8 wires; from 0.5 to 15 mH inductance			
Size	70 x 40 x 18 mm + the motor size			
Ambient temperature	-5		45	°C
Storage temperature	-25		85	°C
IP	40			

Figure 5.2: Electrical Characteristics

The driver integrates the motor controller, power stage, and a 12-bit magnetic encoder directly on the motor itself, which removes the need for a separate driver board and the associated wiring inside the already space-constrained end effector. It is controlled by an ARM-based microcontroller and communicates over CANopen at rates from 10 kHz up to 1 MHz, which is the same field bus used throughout the rest of the rover's control architecture and, as discussed in Section 3.5, the bus that the gripper's own microcontroller shares with the driver. The driver accepts a supply voltage in the 24–80 V_{DC} range and exposes a secondary 12 V_{OUT} rail (pin CN3 in Figure 5.3), which is precisely the voltage tapped to power the gripper's sensing and control electronics, as introduced in the power-stage discussion of Section 3.2. Its onboard firmware implements closed-loop torque control based on the magnetic encoder reading, automatically modulating motor current with the load reducing in this way unnecessary heat dissipation during idle or low-load phases of the grasp cycle while still allowing the driver to increase torque up to its configured maximum when resistance is encountered. At $70 \times 40 \times 18$ mm plus the motor's own footprint, the driver's compact form factor is consistent with the tight mechanical envelope available on the end effector, and its rated motor current (up to 4 - 5 mA/phase) leaves ample margin for the addition of the reduction stage discussed in Section 3.5 to reach the target fingertip force.



CN1		CN2		CN3		JP1
Power supplies Digital inputs CAN / RS-485		RS-232		Digital input		Programming jumper
Header						
Molex 560020-1220		JST B3B-PH-SM4-TB		Molex 560020-0420		Harwin M50-3630242
Housing						**Socket**
Molex 502351-1200		JST PHR-3		Molex 502351-0400		Harwin M50-1900005
Contacts						
Molex 560085-0101		JST SPH-002T-P0.5S		Molex 560085-0101		
Pins						
1	CAN H RS-485 A	1	RS-232 TX	1	GND	
2	CAN L RS-485 B	2	GND	2	12 V_{OUT}	
3	Shield	3	RS-232 RX	3	12 V_{OUT}	
4	CAN H RS-485 A			4	IN4	
5	CAN L RS-485 B					
6	Shield					
7	IN1					
8	IN2					
9	IN3					
10	24 V_{EXT}					
11	V_{POW}					
12	GND					

Figure 5.3: Motor Pinout

5.2 Power Conversion Stage - LDO

As introduced in Section 3.2, the 12 V rail derived from the stepper motor driver is too high to power the microcontroller and the rest of the digital and analog electronics directly, and is therefore stepped down in two successive stages using low-dropout (LDO) linear regulators. Both stages are implemented with the AMS1117 family of fixed three-terminal regulators: an AMS1117-5.0 converts the 12 V rail down to 5 V, and an AMS1117-3.3 further converts this 5 V rail down to 3.3 V. The AMS1117 was chosen for both stages primarily for its simplicity and ubiquity. It is a fixed-output, three-pin device that requires no external feedback network, only input and output decoupling capacitors, and it is widely available, inexpensive, and well characterized, which made it a natural default choice for a first hardware iteration whose priority was to get a working prototype onto the NUCLEO board and, later, onto the custom PCB (Figure 5.4).

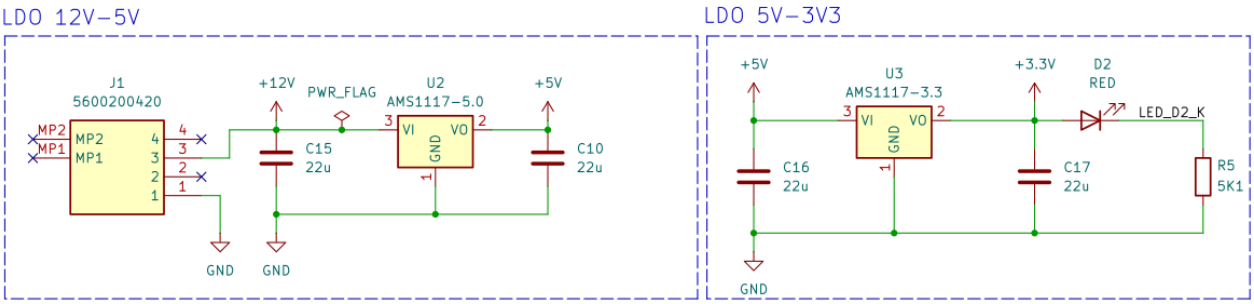


Figure 5.4: LDO Schematics

Both regulators share the same electrical characteristics, since they belong to the same IC family and differ only in their fixed output voltage. The AMS1117 delivers up to 1 A of output current, with a typical dropout voltage of 1.2 V and a guaranteed maximum of 1.3 V at full load, meaning the regulator can maintain regulation as long as the input voltage stays at least this far above the nominal output. This is comfortably satisfied by both stages in this design: the 12 V-to-5 V stage operates with roughly 7 V of headroom, and the 5 V-to-3.3 V stage with roughly 1.7 V, both well above the dropout limit even accounting for typical supply tolerances. Line and load regulation are specified at a maximum of 0.4% and on-chip trimming keeps the output voltage accuracy within about 1–1.5% of nominal, which is more than sufficient for supplying digital logic and the analog signal-conditioning circuitry described in Section 3.4. The device also integrates current limiting

and thermal shutdown protection, providing a basic level of self-protection against output short circuits or excessive load without requiring additional protection components on the regulator itself.

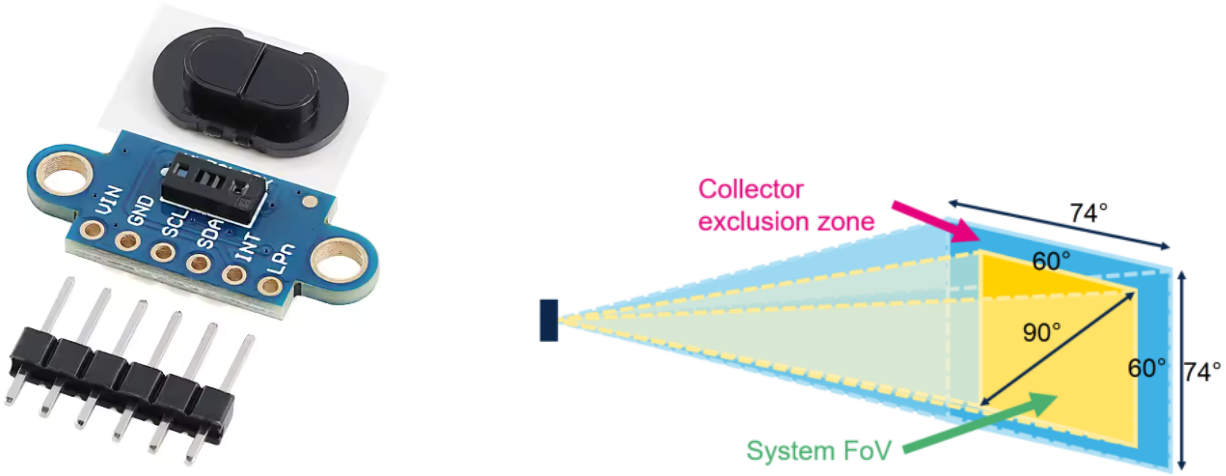
The two-stage arrangement follows the same reasoning already outlined in Section 3.2. Splitting the conversion keeps the input-to-output differential – and therefore the power dissipated as heat – more balanced across the two devices, and it allows the intermediate 5 V rail to be used directly wherever it is needed by the analog front-end of the tactile sensing chain. Since the AMS1117 is a linear regulator, all of the voltage dropped across each stage is dissipated as heat which is an acceptable trade-off given the modest current draw of the gripper's electronics, but is noted here as a design simplification that trades conversion efficiency for circuit simplicity.

5.3 Sensors

5.3.1 Time of Flight (ToF) Sensors

Pre-grasp proximity sensing is provided by the STMicroelectronics VL53L7CX (in Figure 5.5), a multizone Time-of-Flight sensor based on ST's FlightSense technology [46]. The sensor projects an infrared, 940 nm VCSEL laser source through a metasurface lens that shapes it into a 60°×60° square field of view (90° diagonal), and divides this field of view into an 8×8 grid of 64 independent ranging zones, each returning an absolute distance measurement to the nearest target within it, largely independent of the target's color or reflectance (Figure 5.5) [46, 44]. This gives the sensor the character of a coarse, low-resolution 3D range image rather than a single-point distance reading, which is precisely what motivates its use for pre-grasp geometric awareness, as discussed in Section 2.2: it can report not just whether an object is present in front of the gripper, but roughly where within the approach volume it sits, and at what distance. The sensor can also operate at a reduced 4×4 resolution (16 zones) when a faster or less data-intensive update is preferred, with ranging frequency configurable up to 60 Hz in 8×8 mode and higher in 4×4 mode [44], and its rated maximum range is on the order of 4 m, far beyond what is needed for pre-grasp sensing but useful during the rover's broader approach maneuvers.

The VL53L7CX communicates over a standard I^2C interface, capable of running up to 1 MHz, with a default device address of 0x52 that can be reprogrammed in firmware if multiple sensors need



(a) ToF sensor module

(b) ToF sensor field of view

Figure 5.5: ToF sensor module and field of view

to share the same bus [44]. This digital, register-based interface is a key reason the sensor fits the embedded-processing philosophy of the gripper introduced in Section 2.2. Unlike a depth camera, it requires no dedicated image-processing pipeline or external compute, since all ranging computation – including per-zone histogram processing to suppress crosstalk and multi-target detection within a single zone – is performed on the sensor's own internal hardware, and the microcontroller only needs to poll a compact, fixed-size array of distance values over I^2C. This keeps the ToF sensor's data acquisition compatible with the same real-time, resource-constrained microcontroller loop used to read the FSR tactile matrices described in Section 5.3.2, consistent with the overall design goal of keeping the entire sensing-to-actuation pipeline on a single embedded platform.

Mechanically, the sensor is mounted on the palm of the end effector, centered between the two fingers, as introduced in Section 3.5; this placement gives it an unobstructed, symmetric view of the approach volume directly ahead of the gripper, along the same axis the fingers close on, which is the region relevant to detecting an object's presence and coarse position before contact occurs.

5.3.2 Tactile Sensors - Force Sensitive Resistor Matrix

Contact force sensing on each finger is provided by a low-cost, flexible thin-film pressure sensor array, structured as a 8×4 grid of 32 independent piezoresistive sensing zones on a single flexible substrate (Figure 5.6).

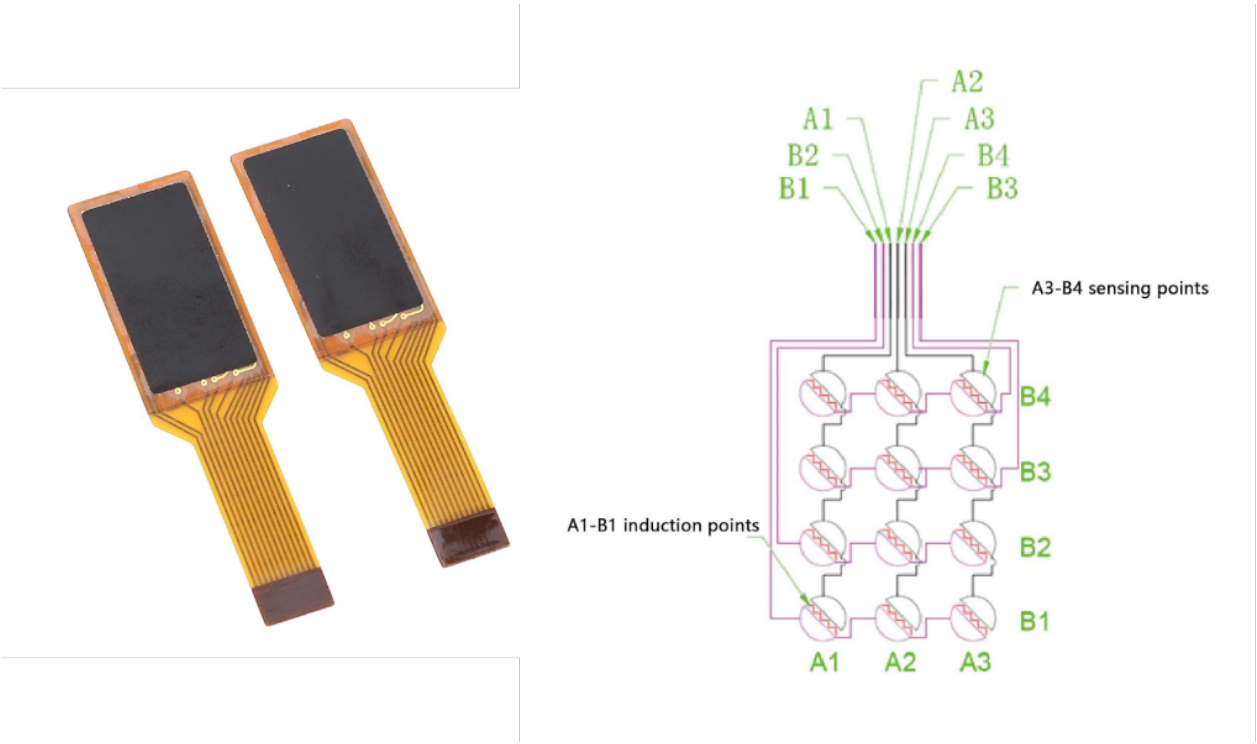


Figure 5.6: FSR Matrix sensors

Each zone behaves as an individual FSR taxel: as shown in Figure 5.7, its electrical resistance decreases monotonically as the applied pressure increases, from an open-circuit (very high resistance) state with no load down to a few kilo-ohms under a load on the order of 2 kg, with an activation threshold as low as about 1 g. The 32 sensing zones are arranged over an active area with individual taxels approximately 2.5×2.5 mm in size, and the whole sensor is manufactured as a thin (approximately 0.3 mm), flexible polyester-film laminate, which allows it to be mounted directly onto the gripper's fingers. Electrical connection to the sensor is made through a single flat flexible-printed-circuit (FPC) connector with a 0.5 mm pin pitch, which brings out the row and column lines.

It should be noted that, as with any resistive thin-film sensor of this kind, the manufacturer-specified force range, accuracy, and repeatability figures are indicative rather than tightly guaranteed, and are subject to unit-to-unit variation, mechanical hysteresis, and drift under sustained load, consistent with the general limitations of FSR technology discussed in Section 2.1.

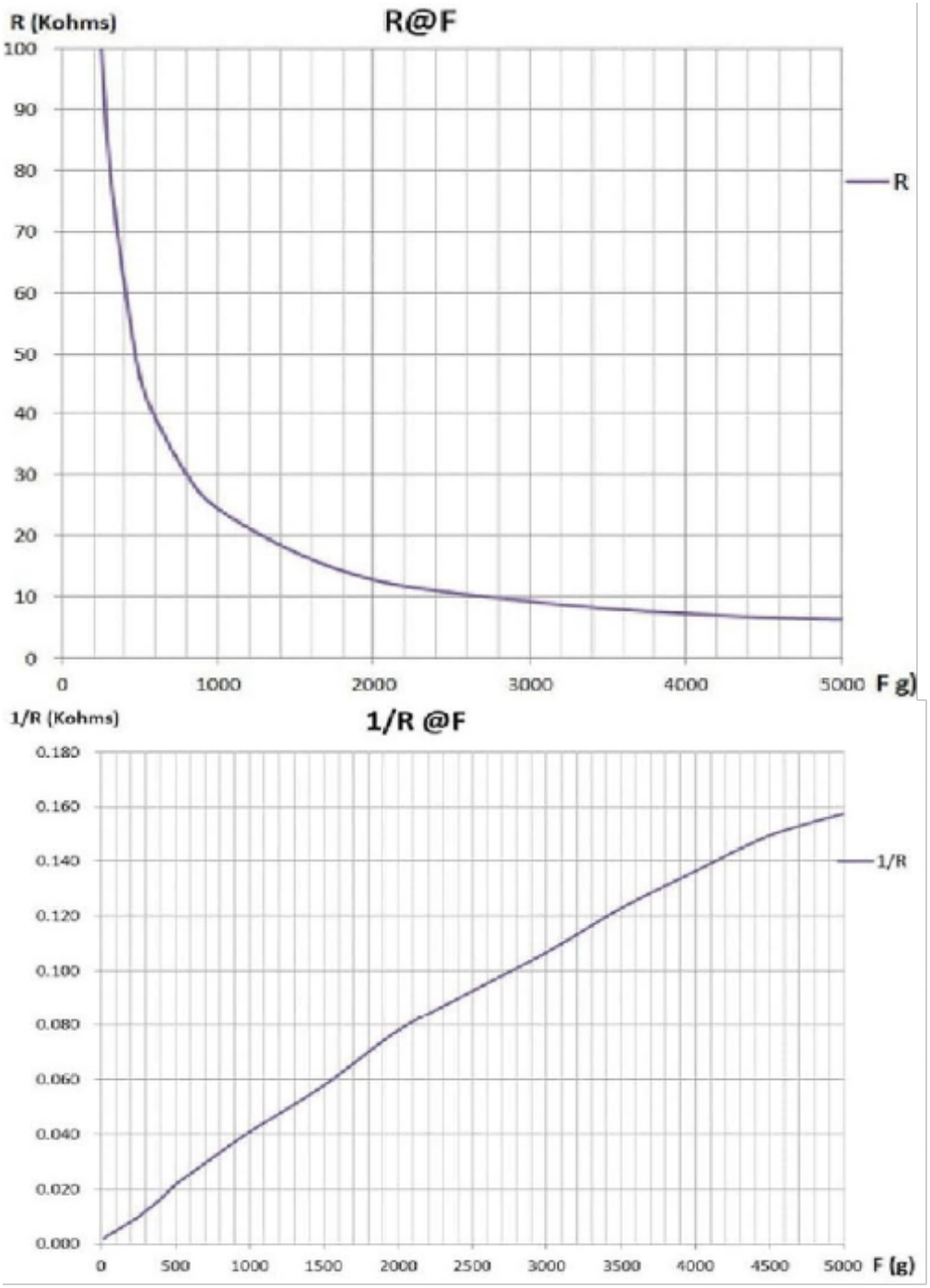


Figure 5.7: Typical force-resistance characteristic of a single FSR taxel, showing the monotonic decrease in resistance with increasing applied load.

5.4 FSR Matrix Interfacing Circuitry

Having introduced the tactile sensor itself in Section 5.3.2 and the general reading concept in Section 3.4, this section describes the actual analog signal chain implemented to read the FSR matrices, following the same order in which a signal travels from a single taxel to the microcontroller's ADC. For sake of clarity, a schematic is reported in Figure 5.8.

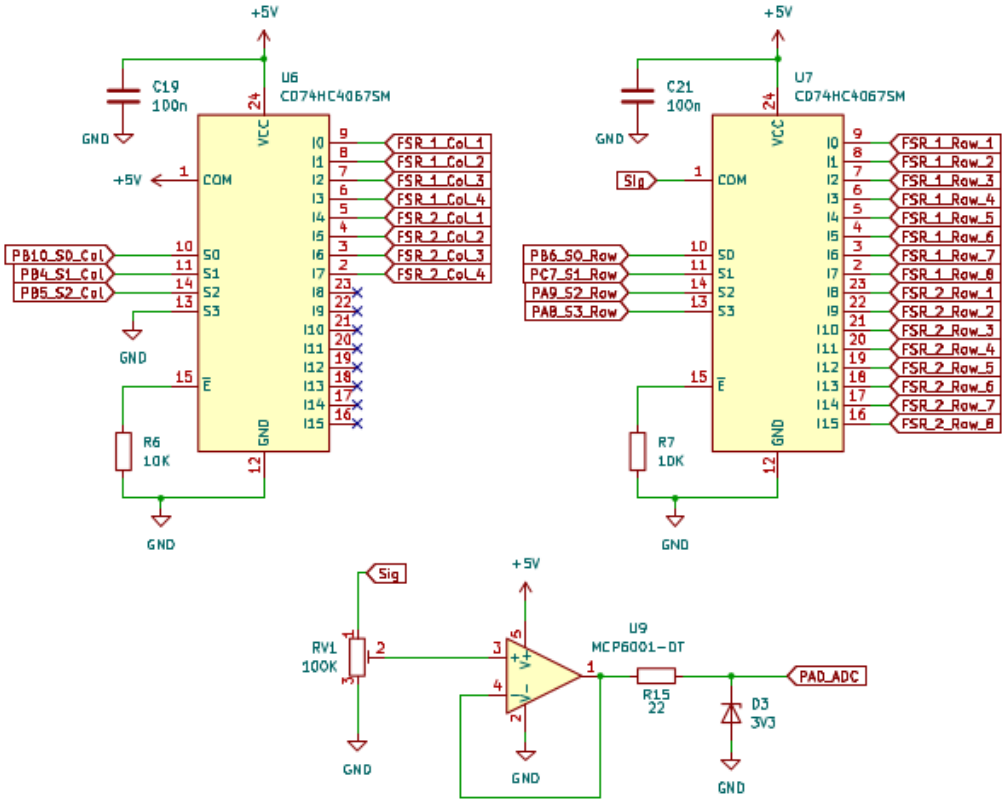


Figure 5.8: FSR matrix interfacing circuitry schematic

5.4.1 Row and Column Multiplexing

Each 8×4 FSR matrix exposes its rows and columns as separate electrical lines rather than as 32 individually wired taxels, which is exactly what makes the multiplexed reading scheme described conceptually in Section 3.4 both necessary and convenient. In this design, row and column selection is implemented using two Texas Instruments CD74HC4067 16-channel analog multiplexers/demultiplexers, one dedicated to the rows and one to the columns of each finger's matrix.

The CD74HC4067 is a digitally controlled, bidirectional analog switch: four binary select lines (S0–S3) address one of sixteen input/output channels at a time, connecting it to a single common pin, and an active-high enable pin can disable all channels simultaneously. The devices operate over a wide supply range (2–6 V) with a low, largely resistance-independent ON-resistance, and the microcontroller drives their select lines directly as digital outputs. By stepping through all valid row/column combinations in sequence and sampling the resulting analog signal after each change, the microcontroller reconstructs the full pressure map of both matrices from a single shared analog front-end, exactly as described conceptually in Section 3.4.

5.4.2 Voltage Divider

Once a specific taxel is addressed through the row and column multiplexers, its variable resistance must still be converted into a voltage the ADC can measure. This is done with a simple voltage divider, in which the selected taxel is placed in series with a second, fixed-value resistive element, and a known reference voltage is applied across the pair; the voltage measured at their midpoint depends on the ratio between the taxel's resistance and the fixed resistance, and therefore on the force currently applied to that taxel. Rather than using a fixed resistor for the second element of the divider, a 100 kΩ trimmer potentiometer is used instead. This choice is deliberate: because the FSR taxels used here, like resistive tactile sensors in general, show unit-to-unit variation, nonlinearity, and drift, as discussed in Section 5.3.2, a fixed divider resistance would fix the sensor's useful voltage range at design time, with no way to adjust it afterward. The trimmer instead allows the divider ratio – and therefore the portion of the ADC's input range over which the sensor's force response is spread – to be tuned in place, after assembly, effectively acting as a manual calibration stage for each finger's tactile sensing chain.

5.4.3 Signal Buffering

The voltage produced by the divider is a comparatively high-impedance signal, since it is generated across resistances on the order of tens to hundreds of kilo-ohms; if this signal were routed directly into further circuitry, or into a long trace toward the ADC protection stage, any loading from downstream components could pull the divider's operating point away from its intended value and

distort the force reading. To prevent this, the divider's output is buffered through a Microchip MCP6001 operational amplifier, configured as a unity-gain voltage follower. The MCP6001 is a low-power, rail-to-rail input/output op-amp that operates from a single supply as low as 1.8 V and up to 6 V, drawing only about 100 μA of quiescent current, with a gain-bandwidth product of 1 MHz – more than sufficient given that the signal being buffered changes only as fast as the multiplexers step through row/column combinations. In this configuration, the buffer presents a very high input impedance to the voltage divider, so it does not meaningfully load it, while presenting a low output impedance to the rest of the circuit, isolating the divider's sensitive operating point from whatever comes after it and preserving the accuracy of the force-to-voltage conversion.

5.4.4 ADC Protection

Between the buffer output and the microcontroller's ADC input, a simple protection network is inserted to guard the microcontroller against voltage excursions outside its safe input range, whether from transients, wiring faults, or an unexpectedly high divider output during calibration. This network consists of a 22 Ω series resistor followed by a 3.3 V Zener diode connected from the signal line to ground. Under normal operation, the buffered signal passes through the series resistor essentially unaffected, since no current flows through the reverse-biased Zener diode as long as the signal stays below its 3.3 V breakdown voltage. If the signal were to rise above this threshold, the Zener diode would begin to conduct, clamping the voltage at the ADC input to approximately 3.3 V and diverting the excess current through the 22 Ω resistor, which limits the current drawn from the buffer stage and protects both the op-amp output and the microcontroller's ADC pin from damage. This protection stage is deliberately simple and passive, adding negligible distortion to the signal under normal conditions while providing a hard safety margin against out-of-range inputs before the signal reaches the microcontroller described in Section 5.5.

5.5 Microcontroller

The control and communication stage described in Section 3.2 is built around an STMicroelectronics STM32F446RE microcontroller, chosen to satisfy the embedded, real-time processing requirements set out throughout this thesis.

The STM32F446RE is built around an Arm Cortex-M4 32-bit RISC core with a hardware floating-point unit, running at up to 180 MHz and delivering up to 225 DMIPS, made possible in part by ST's ART Accelerator, which allows zero-wait-state code execution directly from flash memory. This performance margin leaves substantial headroom for the admittance control algorithm and sensor-processing routines to run comfortably within the timing budget imposed by the real-time control loop, discussed further in Chapter 6. The device offers up to 512 kB of flash memory and 128 kB of SRAM, more than sufficient for the firmware, tactile-processing buffers, and communication stack required by this application, and it integrates a rich set of peripherals directly relevant to the gripper's architecture: three 12-bit ADCs, used to digitize the buffered tactile signal described in Section 5.4; multiple I^2C interfaces, used for the ToF sensor link introduced in Section 5.3.1; and, a built-in CAN controller, which generates the CAN protocol frames handled by the external transceiver described in Section 3.2 without requiring a separate microcontroller peripheral or coprocessor for the field bus.

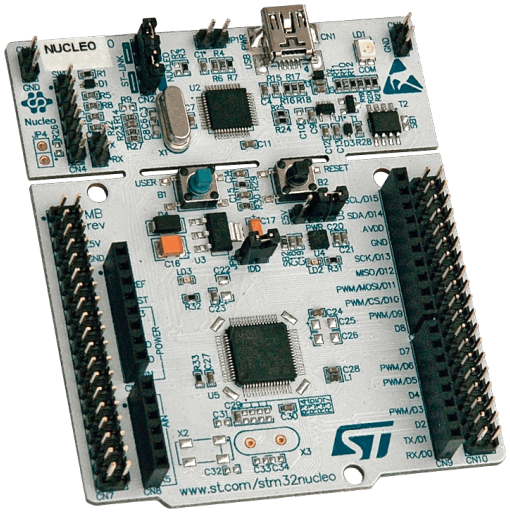


Figure 5.9: NUCLEO-F446RE development board

For the prototyping phase of this project, the STM32F446RE is used in the form of the NUCLEO-F446RE development board. The NUCLEO-F446RE is a STM32 Nucleo-64 board carrying the STM32F446RET6 variant of the microcontroller in an LQFP64 package, and it was chosen for prototyping specifically because it removes several sources of friction that would otherwise slow down early development. It integrates an on-board ST-LINK/V2-1 debugger and programmer, exposed through the SWD interface already shown in the block diagram of Section 3.2, so that

firmware can be flashed and debugged directly from a PC without any external programmer hardware; it exposes the microcontroller's I/Os through both Arduino Uno V3 headers and ST Morpho connectors, giving full access to every pin needed to wire the FSR reading circuit, the ToF sensor, and the CAN transceiver during breadboard prototyping; and it offers flexible power-supply options (USB or an external 3.3–12 V source), which made it straightforward to bench-test the electronics independently of the motor driver before integrating the two, even though in the final architecture the board is powered from the 3.3 V rail derived from the motor driver's 12 V output, as described in Section 5.2.

5.6 CAN Transceiver - MCP2551

As introduced in the block diagram of Section 3.2, the microcontroller's internal CAN controller generates the CAN protocol frames used for both real-time control and telemetry, but it does not itself drive the physical bus. The CAN controller's logic-level TX and RX signals still need to be converted into the differential voltage levels that travel over the shared CAN bus, and protected from the electrical conditions present on it. This conversion is performed by a Microchip MCP2551 high-speed CAN transceiver (Figure 5.10), placed between the microcontroller's CAN controller and the CAN bus shared with the rover's controller and the motor driver, as discussed in Section 3.5.

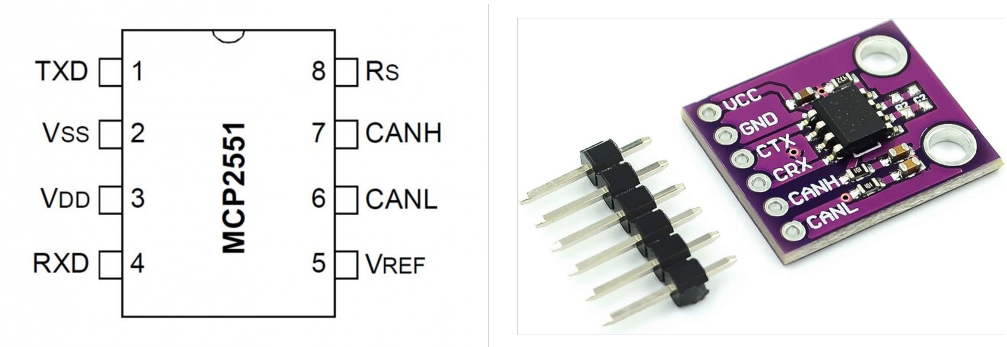


Figure 5.10: MCP2551 CAN Transceiver

The MCP2551 is a fault-tolerant, high-speed CAN transceiver that implements the physical-layer requirements of the ISO-11898 standard and supports bus speeds of up to 1 Mb/s, comfortably covering the CANopen data rates relevant to this project. On the microcontroller side, it accepts the

single-ended TXD/RXD logic signals produced by the CAN controller; on the bus side, it drives and receives the differential CANH/CANL signal pair, converting the microcontroller's logic-level bits into the dominant/recessive voltage states that constitute the CAN protocol on the wire. Because it exposes a purely digital interface toward the microcontroller and a purely differential interface toward the bus, the MCP2551 is essentially the only component in the communication chain that needs to interface directly with the physical characteristics of the shared bus, keeping the rest of the communication stack – the CAN controller and higher-level protocol logic running on the microcontroller – independent of the specific electrical implementation of the bus itself.

Beyond simple signal conversion, the MCP2551 provides a level of protection and robustness that is particularly relevant given that the gripper's CAN interface is shared, as discussed in Section 3.5, with the motor driver rather than being wired independently: it is rated for 12 V and 24 V systems, tolerates high-voltage transients and short-circuit conditions on the bus lines without damage, and includes automatic thermal shutdown protection. It also implements a ground-fault detection feature on its TXD input, disabling the CANH/CANL output drivers if TXD is held low for longer than the specification allows, which prevents a faulty or misbehaving node from corrupting bus traffic indefinitely. An unpowered or reset MCP2551 node does not disturb the bus either, an important property on a shared CAN network with multiple nodes – the gripper's microcontroller, the motor driver, and the rover's controller – where any single node's power-up or reset transient must not interfere with ongoing communication between the others. Finally, its differential signaling and the bus's inherent noise immunity make it well suited to operating in close physical proximity to the switching noise generated by the stepper motor driver, a consideration directly relevant to the electromagnetic compatibility of the compact, integrated PCB layout discussed later in this chapter.

5.7 PCB Layout

This section describes how these blocks were brought together on the custom printed circuit board, replacing the NUCLEO-F446RE prototyping platform introduced in Section 5.5, and details the auxiliary circuitry, stack-up, and physical layout choices needed to integrate them reliably within the gripper's mechanical envelope. Full PCB schematics are available in the appendix.

5.7.1 Board-Level Integration and Auxiliary Circuitry

Moving from the NUCLEO board to a custom board required re-implementing, around the same STM32F446RET6 device (now used in bare LQFP-64 form rather than as a pre-mounted module), several support circuits that the development board had previously provided for free (Figure 5.11). Firmware programming and debugging, no longer available through an on-board ST-LINK as on the NUCLEO board, is instead exposed through a dedicated four-pin SWD header carrying SWDIO, SWCLK, 3.3 V, and GND, allowing an external ST-LINK probe to be connected during development and, if needed, during field servicing of the gripper. Two further additions support bring-up and diagnostics: a two-position switch allows BOOT0 to be forced high while the microcontroller is reset, so that the device can be placed in DFU mode without removing it from the assembly, while a separate push-button drives NRST for manual resets, debounced with a local decoupling capacitor. A single status LED, driven by a spare GPIO, was also included as a firmware-controllable indicator, giving a visual readout of the control loop's state without requiring a debugger to be attached.

On the communication side, the CAN transceiver introduced in Section 5.6 is completed at board level with the passive network needed to interface with the physical bus (Figure 5.12).

An RC filter (a series resistor followed by a capacitor to ground on each of CANH and CANL) is placed close to the transceiver's bus-side pins to attenuate high-frequency conducted noise before it reaches the connector, complementing the transceiver's own transient protection. The 120 Ω differential termination resistor required at the ends of a CAN bus is not populated by default; instead, it is connected through a jumper-selectable header, so that the same board can be configured as either a terminating node or a mid-bus node purely by placing or omitting a jumper at assembly time, without requiring a separate PCB variant for each case – a convenient property given that the gripper's position on the rover's CAN network, shared with the motor driver and the rover's controller as discussed in Section 3.5, need not be fixed at design time.

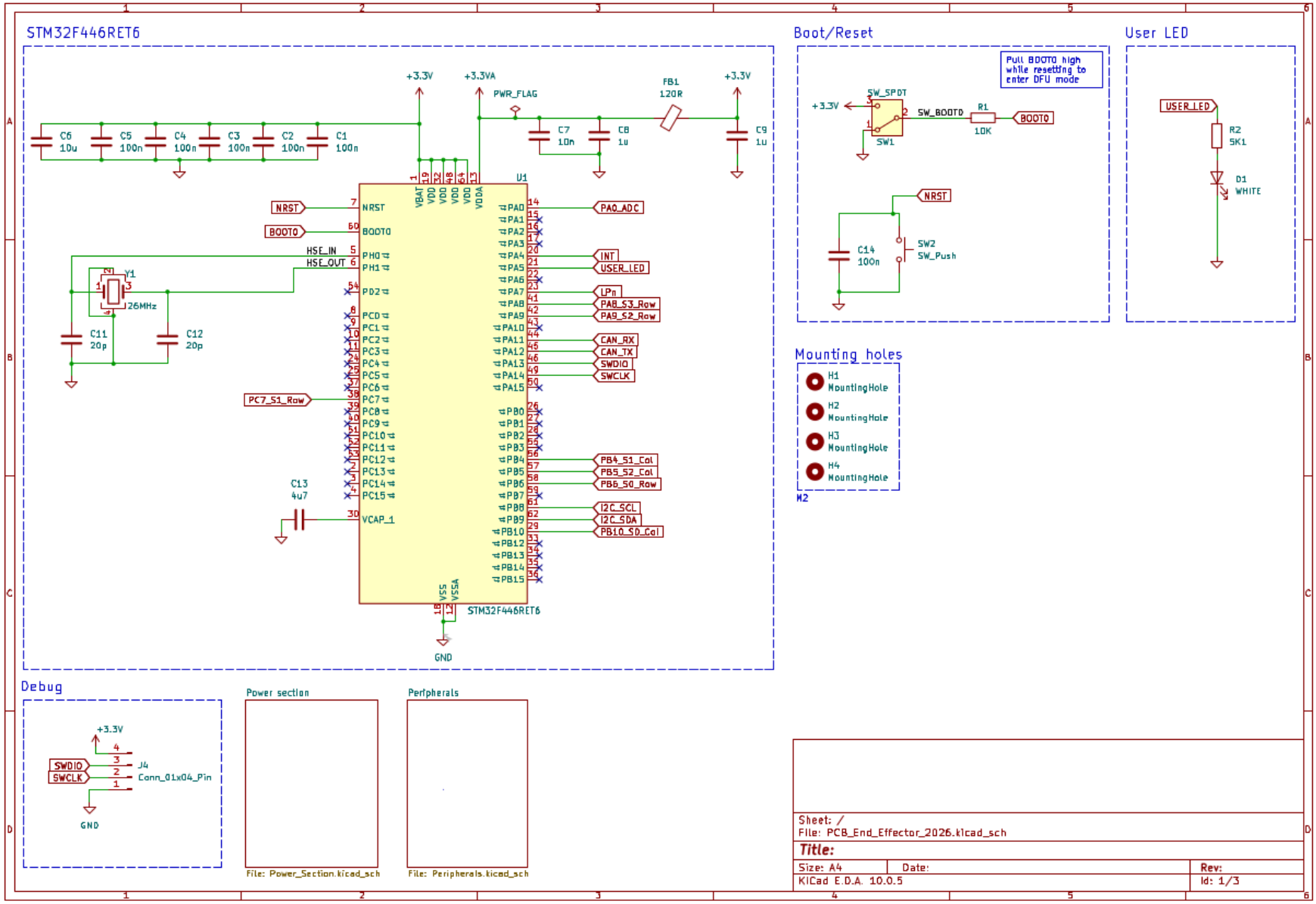


Figure 5.11: Support circuitry for the STM32F446RET6 microcontroller on the custom PCB, including SWD programming/debugging header, BOOT0 switch, NRST push-button, and status LED.

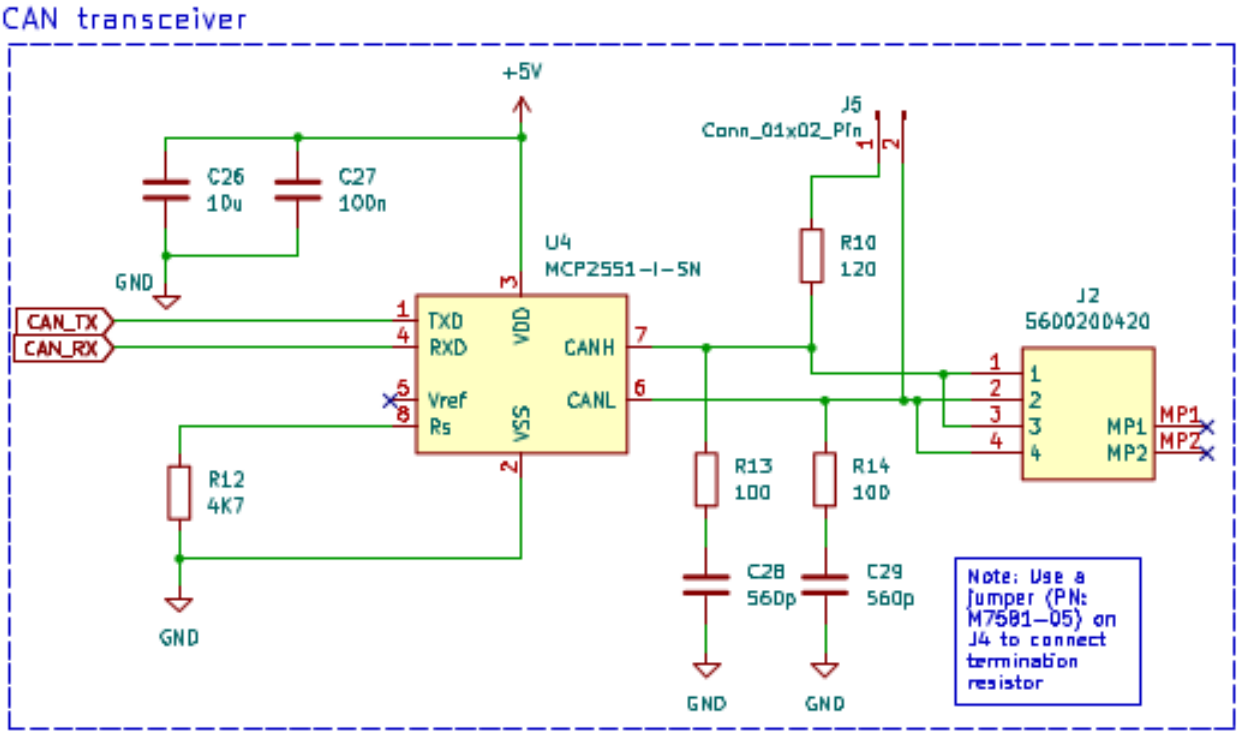
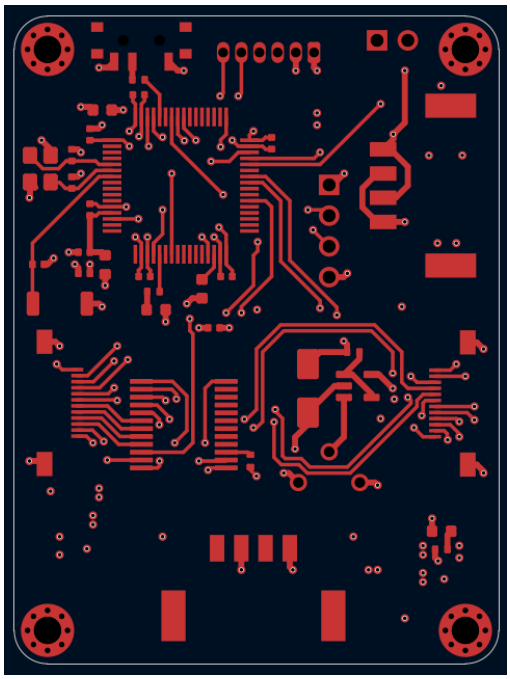


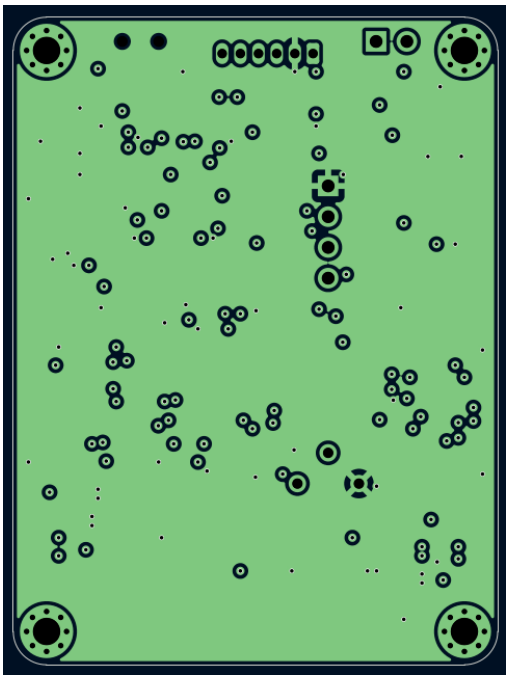
Figure 5.12: CAN transceiver and associated passive network on the custom PCB, including RC filter and jumper-selectable termination resistor.

5.7.2 Stack-Up and Grounding Strategy

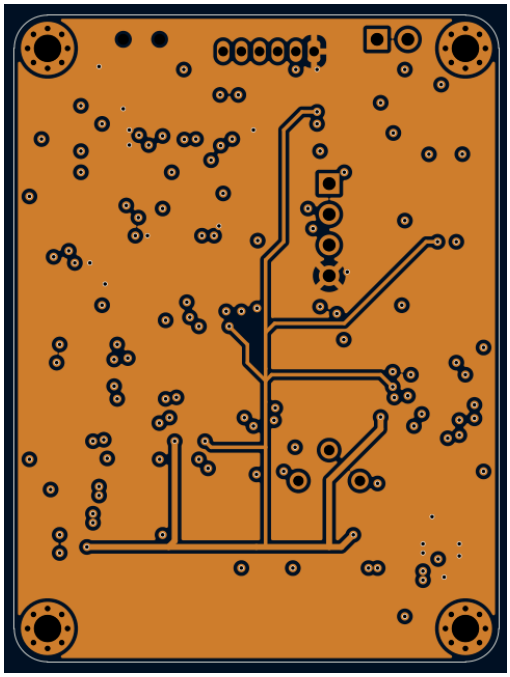
Given the density of digital, analog, and power circuitry required within the gripper's compact mechanical envelope, the board was implemented as a 4-layer FR-4 PCB rather than a 2-layer one. The four layers (Figure 5.13) are allocated as follows: the top layer (F.Cu) carries the majority of the components – the microcontroller and associated passive components, one of the two CD74HC4067 multiplexer and the power, communication and sensor's connectors; the first inner layer (In1.Cu) is a continuous ground plane; the second inner layer (In2.Cu) is used for power distribution, carrying the 5 V and 3.3 V rails as filled copper pours to shorten the current return path and reduce the number of vias otherwise needed to bring power to each device; and the bottom layer (B.Cu) completes the routing not accommodated on the top layer, in particular around the peripheral connectors.



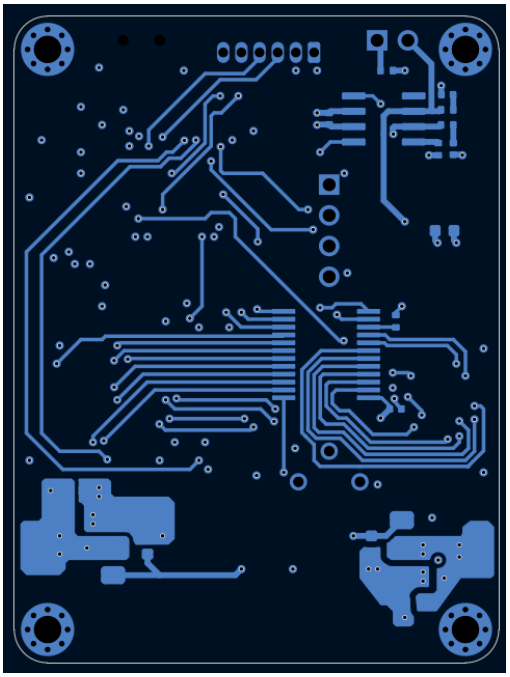
(a) Top layer (F.Cu)



(b) First inner layer (In1.Cu)



(c) Second inner layer (In2.Cu)



(d) Bottom layer (B.Cu)

Figure 5.13: PCB stack-up

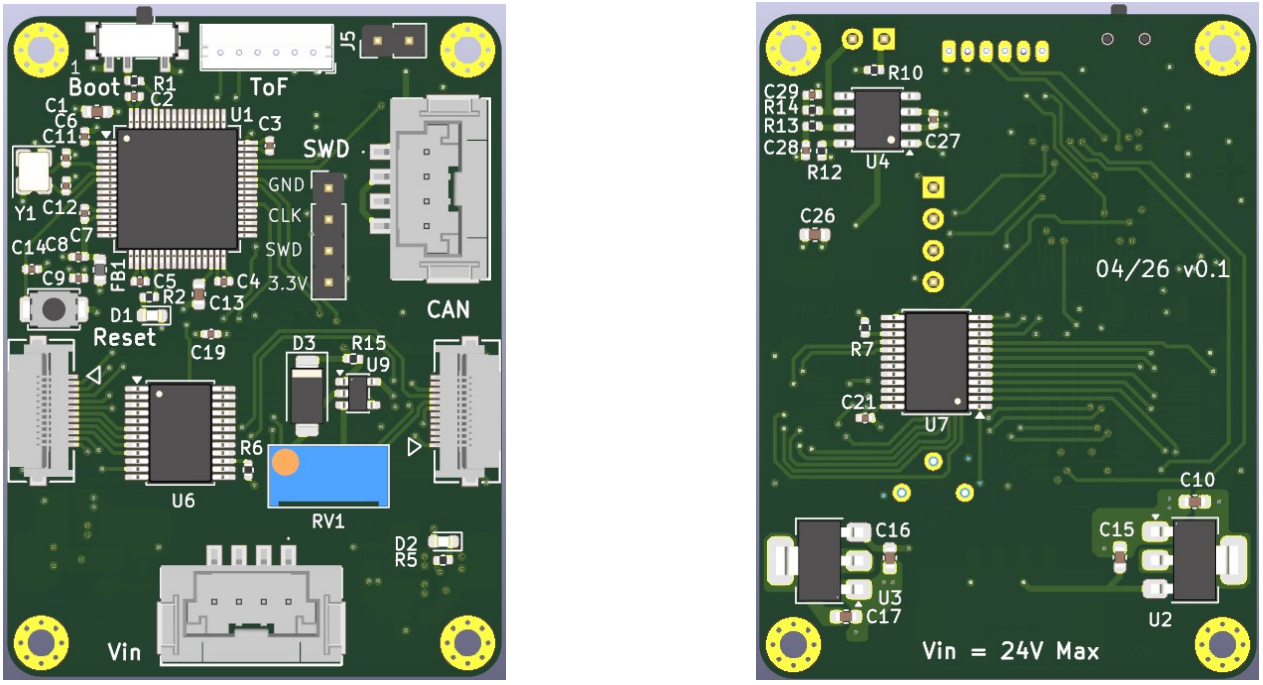
The continuous ground plane on In1.Cu, sandwiched directly beneath the component and routing layer, was the single most consequential stack-up decision from a signal-integrity standpoint. It provides a low-impedance return path for the digital switching currents generated by the multiplexer select lines and by the microcontroller's I/O toggling, and – more importantly for the sensing chain described in Section 5.4 – it forms a shielding layer between this digital activity and the analog signal path carrying the buffered FSR voltage, whose amplitude and bandwidth are small enough to be sensitive to common-mode noise injected from the digital domain. This layer arrangement extends, at the layout level, the same digital/analog separation already pursued at the schematic level through the filtered VDDA rail described above and the buffering and protection stages of Sections 5.4.3-5.4.4.

5.7.3 Physical Footprint and Component Placement

The board outline was constrained by the requirement, introduced in Section 3.4, that the sensing and control electronics be integrated as a compact add-on to the FD1.1GA driver rather than as an independently sized enclosure, and it was accordingly kept to a footprint on the order of 40×54 mm, with four M2 mounting holes at the corners for direct fastening to the end-effector structure. Meeting this footprint while accommodating the LQFP-64 microcontroller, the two SSOP-24 multiplexers, the CAN transceiver, both LDO regulators, and the associated passive networks required populating nearly all passive components – resistors, capacitors, and the ferrite bead – in 0402 packages, trading manual assembly and inspection convenience for board area.

Component placement follows the signal flow established in the preceding sections (Figure 5.14): the two FSR matrix connectors and the row/column multiplexers are grouped together on one side of the board, immediately adjacent to the voltage-divider, buffering, and protection stages of Section 5.4, so that the sensitive analog signal between the multiplexer output and the microcontroller's ADC input travels the shortest possible trace length; the microcontroller sits at the center, within short routing distance of both the analog front-end and the digital peripherals (I^2C connector for the ToF sensor, SWD header, and CAN transceiver); and the power-conversion stage, together with the CAN transceiver and its bus-side filtering network, is placed toward the board's power-input edge, keeping the higher-current, more EMI-relevant circuitry away from the low-level analog signals. This floorplan, together with the grounding strategy described above, is intended to preserve, at

the physical layout level, the same signal-quality considerations that motivated the schematic-level choices detailed throughout the rest of this chapter.



(a) Front side

(b) Back side

Figure 5.14: Front and back sides of the custom PCB, showing the compact component placement and the four M2 mounting holes at the corners. The front side hosts the microcontroller, multiplexers, and FSR connectors, while the back side accommodates the power-conversion stage and CAN transceiver.

6. Firmware architecture

This chapter describes the embedded firmware running on the STM32F446RE microcontroller introduced in Section 5.5, which implements the tactile and proximity sensing pipelines, the admittance control law, and the CANopen-based communication with the rest of the rover. Where the block diagram of Section 3.2 and the functional decomposition of Section 3.3 described what the system must do and which physical blocks realize it, this chapter describes how the corresponding behavior is actually implemented in software, module by module. The discussion follows the data flow of a single control iteration: from the raw ADC samples produced by the FSR reading chain of Section 5.4, through calibration, filtering, and force estimation, into the admittance control law, and out through the CANopen link to the motor driver and the rover's controller, with the proximity sensing and telemetry paths described alongside as they interleave with this main loop. A companion desktop application, developed to visualize this same sensor data during development and testing, is also described.

6.1 Firmware Architecture Overview

The firmware is organized as a bare-metal loop with interrupt-driven event sources. The application has a single hot path – acquire the tactile matrices, estimate contact force, and update the admittance law – together with a handful of loosely-coupled background activities (CAN frame handling, ToF ranging, UART telemetry) that are naturally expressed as interrupt service routines and non-blocking state machines.

Concretely, as schematized in Figure 6.1, `main()` initializes the peripherals generated by CubeMX (ADC, DMA, timer, CAN, I^2C, UART, GPIO), initializes the application-level modules (the tactile sensing library, the admittance controller, the CANopen layer, the UART streaming utility, and the ToF sensor), and then enters an infinite loop that does nothing but poll the state of the tactile-acquisition DMA state machine described in Section 6.2. Every time this state machine transitions from `DMA_READING` back to `DMA_IDLE` – meaning a full scan of both FSR matrices has just completed – the main loop invokes `Control_Task()`, which computes a new contact-force

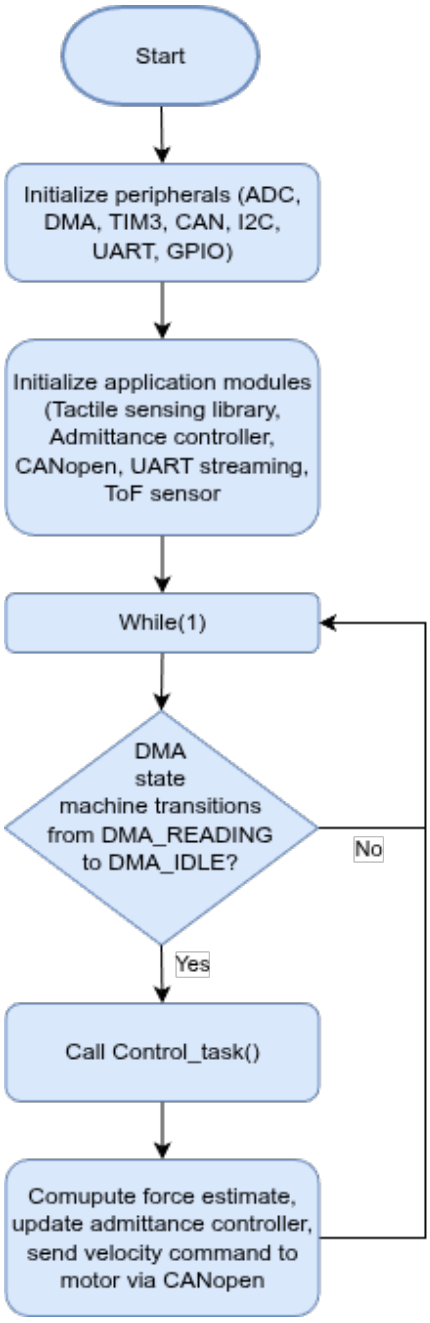


Figure 6.1: Firmware architecture overview

estimate, updates the admittance controller, and issues the resulting velocity command to the motor. This ties the control loop's execution directly to the completion of a new sensor scan rather than to a fixed timer, a point discussed further in Section 6.4. A separate periodic timer (TIM3) is instead responsible only for triggering the start of each new acquisition cycle, decoupling "when a new scan begins" from "when the resulting control action is computed."

6.2 Tactile Sensing Acquisition

The tactile sensing library reads both 8×4 FSR matrices through the shared multiplexer pair described in Section 5.4, exposing the row and column select lines as two small configuration structures (`MuxRowConfig`, `MuxColConfig`) that are bound once, at initialization, to the actual GPIO ports and pins used on the board – the same PB10/PB4/PB5 column lines and PB6/PC7/PA9/PA8 row lines identified in the schematic at Figure 5.8. Selecting a specific taxel is simply a matter of writing the binary representation of its row and column index onto these four and three select lines respectively, exactly as described conceptually in Section 3.4 and implemented in hardware in Section 5.4.

Because both matrices share a single ADC channel (PA0) through the same pair of multiplexers, the 64 taxels — 32 per sensor — cannot be sampled in parallel and must instead be scanned sequentially. This is implemented as a non-blocking, DMA-driven state machine, so that the CPU remains free to service other peripherals while a conversion is in progress. `ForceSensor_DMA_StartRead()` selects the first cell, arms a single ADC conversion in DMA mode, and returns immediately; `ForceSensor_DMA_ConvCpltCallback()`, invoked by the HAL when the conversion completes, stores the result and advances the state machine to the next cell via `ForceSensor_DMA_ProcessNextCell()`, which reconfigures the multiplexer select lines, inserts a short settling delay, and re-arms the next conversion — repeating this sequence until all 32 cells of both matrices have been visited, at which point the state machine returns to `DMA_IDLE`.

A detail worth highlighting is that every single-cell measurement is actually preceded by a discarded "dummy" conversion. After the multiplexer select lines change, the first ADC conversion is thrown away and only the second one is stored. This directly addresses a settling-time concern already anticipated at the hardware level in Section 5.7: switching the multiplexers' select lines does not instantaneously bring the analog signal path to its new steady-state value, since the buffer stage, trace parasitics, and the multiplexer's own switching transient all need a short interval to settle; sampling immediately after a channel change would otherwise read a value still partly influenced by the previously selected taxel. Discarding the first conversion after every mux change, at the cost of doubling the number of ADC conversions per scan, is a simple and robust way to guarantee that only settled samples are used, without having to characterize and hard-code a precise settling time for the delay loop alone.

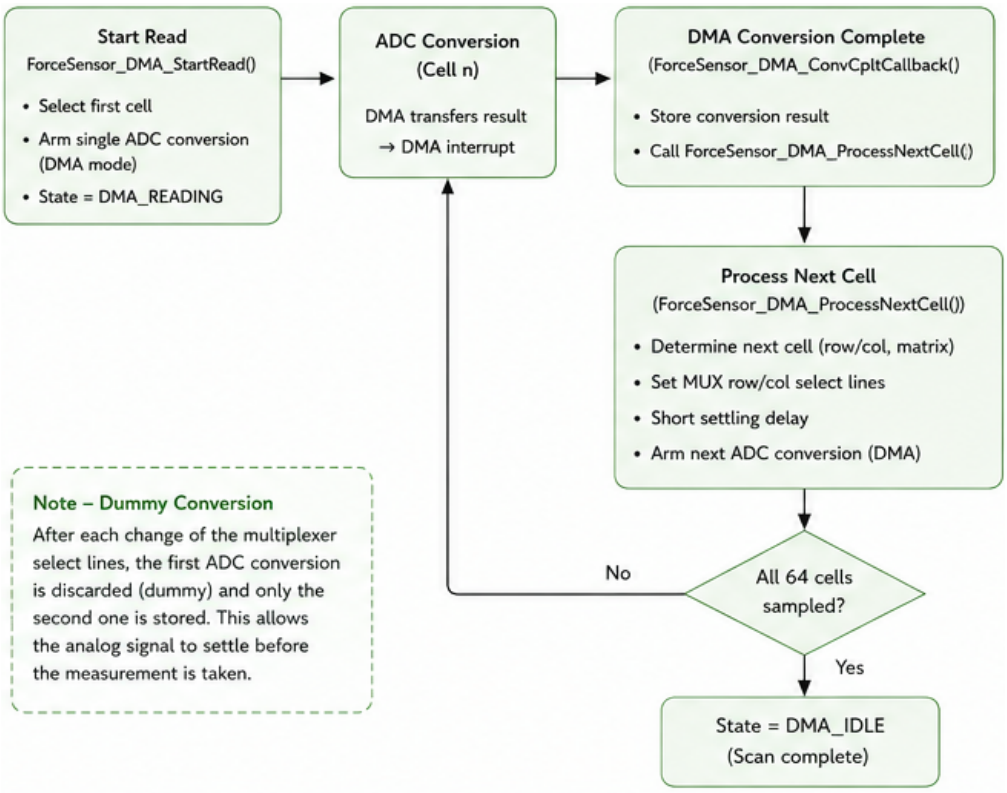


Figure 6.2: Tactile sensing acquisition state machine

As shown in Figure 6.3, two further stages are applied to each raw ADC sample before it is stored in the working buffer, both implemented directly inside the DMA completion callback:

- **Offset correction**: a per-cell baseline, captured once at startup by `ForceSensor_Offset()` by averaging several unloaded readings of every one of the 64 taxels, is subtracted from every subsequent raw sample (clamped to zero to avoid underflow). This compensates for the fixed, cell-to-cell variation introduced by the multiplexers' non-zero and non-uniform ON-resistance, by mismatches in the FSR taxels' unloaded resistance, and by the single shared trimmer's operating point discussed in Section 5.4, none of which are corrected for in the hardware itself.

- **Exponential moving-average filtering**: the offset-corrected sample is combined with the previous filtered value for that same cell through a single-pole IIR filter (Output = $\alpha \cdot$ New + $(1-\alpha) \cdot$ Old), with the coefficient α exposed through `ForceSensor_SetFilterAlpha()` and set, in this application, to 0.6 as a compromise between noise rejection and responsiveness.

Because each taxel is only revisited once per full 64-cell scan, this filter also acts as a simple low-pass smoothing of the scan-to-scan variation of each individual cell, reducing the impact of the occasional noisy sample on the downstream force estimate without materially slowing the response to a genuine contact event.

The filtered, offset-corrected ADC counts are subsequently converted into an estimated force in Newtons by `ForceSensor_ADCtoNewton()`, which inverts the voltage-divider relationship described in Section 5.4 to recover the FSR's instantaneous resistance from the measured voltage, and then applies a linear conductance-to-force approximation ($F = m \cdot G$, with $G = 1/R_{FSR}$) to obtain the final force value. The proportionality constant used here is explicitly marked in the source as a placeholder to be characterized experimentally for the specific FSR taxels used, consistent with the general caveat already raised in Section 2.1 and Section 5.3.2 that resistive tactile sensors of this kind require per-unit calibration rather than relying on a single, universal force-resistance curve; the hardware-level trimmer of Section 5.4 and this firmware-level linear model are best understood as two complementary calibration knobs — one coarse and manual, the other explicit and re-flashable — addressing the same underlying sensor variability.

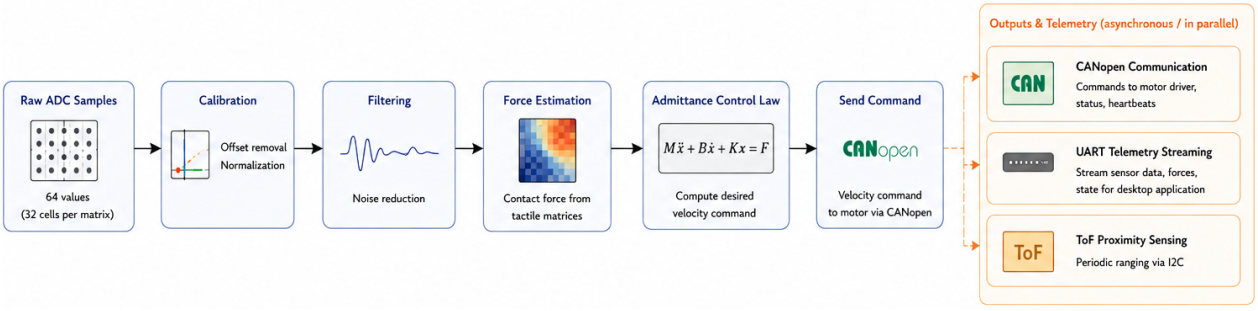


Figure 6.3: Tactile sensing acquisition pipeline

6.3 Contact Force Estimation for Control

Moving on in the chain shown in Figure 6.3, the admittance controller described in Section 6.4 requires a single scalar force value per control iteration, not a full 8×4 pressure map. Rather than averaging all 32 taxels of each matrix directly – which would dilute a small, concentrated contact area under a majority of unloaded cells and produce a systematically underestimated force

– `ForceSensor_GetMeanForce()` first locates the taxel with the highest raw ADC reading in each matrix, then averages the (up to) nine cells in the 3×3 neighbourhood centered on that peak before converting the result to Newtons. This gives a force estimate that is representative of the actual contact region while still being more robust to a single noisy or momentarily saturated cell than using the peak value alone. The two per-sensor estimates (one for each finger) are then averaged into the single force value passed to the admittance controller.

6.4 Admittance Control Implementation

The control law implemented in `AdmittanceControlLib.c` directly realizes the virtual mass-spring-damper dynamics introduced in Section 2.3. At every call, `Admittance_Update()` computes a force error between the desired grasping force setpoint and the force estimate of Section 6.3, applies a configurable deadband to suppress chatter around the setpoint, and integrates the resulting virtual acceleration forward in time using explicit Euler integration to obtain a commanded velocity:

$$\ddot{x} = \frac{K\,(x_d - x) - B\dot{x} - K\,K_p\,F_{error}}{M} \tag{6.1}$$

This velocity is clamped to a configured maximum before being handed to the motor interface described in Section 6.6, and the corresponding position is separately clamped to a maximum travel range purely as an anti-windup safeguard, since the actual finger position is tracked by the motor's own encoder.

Two aspects of the implementation are worth calling out explicitly. First, the virtual mass, damping, stiffness, and force-error gain (M, B, K, K_p) are not read from the compile-time defaults stored in the controller's own parameter struct at every update, but from a separate `set_up_params` structure populated and kept up to date by the CANopen layer described in Section 6.5; this allows the admittance behavior to be re-tuned from the rover's controller at runtime, without reflashing firmware, which is particularly useful given that appropriate gains are expected to vary with the object being grasped, as discussed in Section 2.3.

As introduced in Section 6.1, `Control_Task()` – and therefore this control update – is invoked once per completed tactile scan only while the CANopen control word has previously enabled the loop (`motor_data_canopen.start`, set and cleared remotely through the `CONTROL_WORD` object

described in Section 6.6). This means the achievable control rate is upper-bounded by the time required to scan both FSR matrices – 64 taxels, each requiring two ADC conversions due to the dummy-read scheme of Section 6.2, plus the associated multiplexer settling delays. This coupling between sensing throughput and control rate is a direct, practical instance of the first research question posed in Section 1.3, on whether this sensor suite, processed locally, can support a control loop adequate for stable grasping; characterizing the resulting loop frequency experimentally, and verifying it against the bandwidth required for stable admittance behavior across the object stiffness range of interest, is addressed in Chapter 7.

6.5 CANopen Communication Layer

The firmware implements a compact, purpose-built subset of the protocol in `CanOpenLib.c`, covering exactly the services this application needs: expedited SDO upload/download for single scalar values up to four bytes, and a segmented-transfer mechanism for the two data types too large to fit in one CAN frame – the two 32-byte force matrices and the 64-byte ToF distance map.

The library plays two distinct communication roles simultaneously, reflected in the two COB-ID ranges handled by `HAL_CAN_RxFifo0MsgPendingCallback()`. Toward the motor driver (node `SLAVE_ID`), the gripper's microcontroller acts as an SDO client, issuing `Set_sdo()` and `Get_sdo()` requests – for instance to command a target velocity or request the current encoder value – and parsing the driver's asynchronous `0x580`-range responses to update the shared `MotorData_CanopenUser` structure consumed by `Control_Task()`. Toward the rover's controller (node `STM_ID`), the same microcontroller instead acts as an SDO server, responding to upload and download requests on its own object dictionary. Scalar objects (force, position feedback, and the admittance gains) are served directly through the expedited path, while the two force matrices and the ToF map are served through the segmented path, in which `ok_sdo_segmented()` announces the transfer size and `upload_segmented_data()` then streams the payload seven bytes at a time, toggling a single-bit sequence flag between frames exactly as the segmented SDO protocol requires, tracked through the module-level `seg_len`/`pos_len` bookkeeping variables.

6.6 Motor Interface

`motor.c` provides a small, application-specific abstraction over the CANopen SDO calls needed to drive the FD1.1GA motor driver introduced in Section 5.1, translating between the physical units used by the control law and the raw units expected by the driver's object dictionary. `Profile_Velocity()` takes a desired finger-closing velocity in m/s, clamps it to a configured maximum, converts it into encoder steps per second using the driver's configured resolution, and writes it to the driver's target-velocity object via a single SDO download – issued asynchronously, with no blocking wait for acknowledgement, consistent with the non-blocking philosophy followed throughout this firmware. `Get_Position()` and `Get_Velocity()` similarly issue asynchronous SDO upload requests for the encoder and velocity objects; rather than returning a value directly, they rely on the shared CAN receive callback to write the driver's eventual response into the `MotorData_CanopenUser` structure, which `Control_Task()` reads on its next iteration.

6.7 Proximity Sensing (ToF) Integration

The VL53L7CX driver wrapper in `tof_sensor.c` is built directly on top of STMicroelectronics' Ultra Lite Driver (ULD) API for the sensor, rather than a custom register-level I^2C implementation: every ranging-related operation – reset, initialization, ranging-mode and resolution configuration, target ordering, integration time, and data retrieval – is expressed as a call into the `vl53l7cx_*` function set provided by the ULD, with `tof_sensor.c` acting as a thin application-level wrapper around it, mirroring at the software level the same low-integration-effort argument already made for the sensor itself in Section 5.3.1. `ToF_Sensor_Init()` resets the sensor, enables its low-power mode, verifies communication, and configures it for autonomous 8×8 ranging with the closest-target ordering and the frequency and integration time set out in Section 5.3.1, before starting continuous ranging. Rather than polling the sensor for new data, the firmware relies on the INT line described: `HAL_GPIO_EXTI_Callback()` simply raises a flag when the pin falls, and the actual data retrieval – reading the current resolution and the full ranging-result structure over I^2C – is deferred to `ToF_Sensor_BuildFrame()`, called once per main-loop pass from within the same telemetry-building step as the tactile data (Section 6.8); the function returns immediately, without touching

the I^2C bus, whenever no new frame is pending, keeping this sensing path from ever blocking the rest of the firmware.

Cross-talk calibration – needed to compensate for stray reflections from the sensor's own cover glass or housing back into its receiver, which would otherwise bias every ranging measurement – was performed on the physical prototype using the ULD's built-in calibration routine, `ToF_Sensor_CalibrateXtalk()`, with a known-reflectance target placed at a fixed distance in front of the sensor. Rather than re-running this procedure at every power-up, the resulting calibration buffer was retrieved once via `ToF_Sensor_GetXtalkData()` and hard-coded into firmware as a constant array, which is then reloaded directly into the sensor via `ToF_Sensor_SetXtalkData()` at the start of every subsequent `ToF_Sensor_Init()` call. This removes the calibration delay and the need for a controlled target at a known distance from the ordinary boot sequence, at the cost of not adapting to any change in the sensor's optical path – for instance dust or scratching of the cover window – occurring after the calibration buffer was captured. Re-running the calibration procedure remains available as a maintenance step should ranging accuracy degrade over the gripper's service life.

6.8 Telemetry and Desktop Visualization

Alongside the CANopen link of Section 6.5, the firmware streams a human-readable telemetry frame over UART, combining the per-taxel force values of both matrices, the mean force, and the latest ToF frame into a single line of text. A small double-buffering utility, `uart_stream.c`, transmits one buffer via DMA while the next frame is built into the other; `Control_Task()` only sends a new frame when the previous transmission has completed, so the link is self-throttling and never stalls the control loop. Together, CAN and UART are the only two channels through which the electronics exchange data with the outside world: the former carries control and telemetry to the rover, the latter serves as a diagnostic link to a development PC.

To consume this stream, a standalone Python desktop application was developed alongside the firmware, reading the serial port on a background thread and parsing the same frame formats used for CAN telemetry (Sections 6.2–6.7). It renders both FSR matrices as 8×4 heatmaps and the ToF data as an 8×8 depth map with live min/max/mean statistics, alongside a mean-force readout

and a raw console. This tool was the primary instrument used during the development to validate the tactile scanning's row/column mapping, confirm correct contact localization, and check the VL53L7CX's ranging behavior, without requiring external lab equipment.

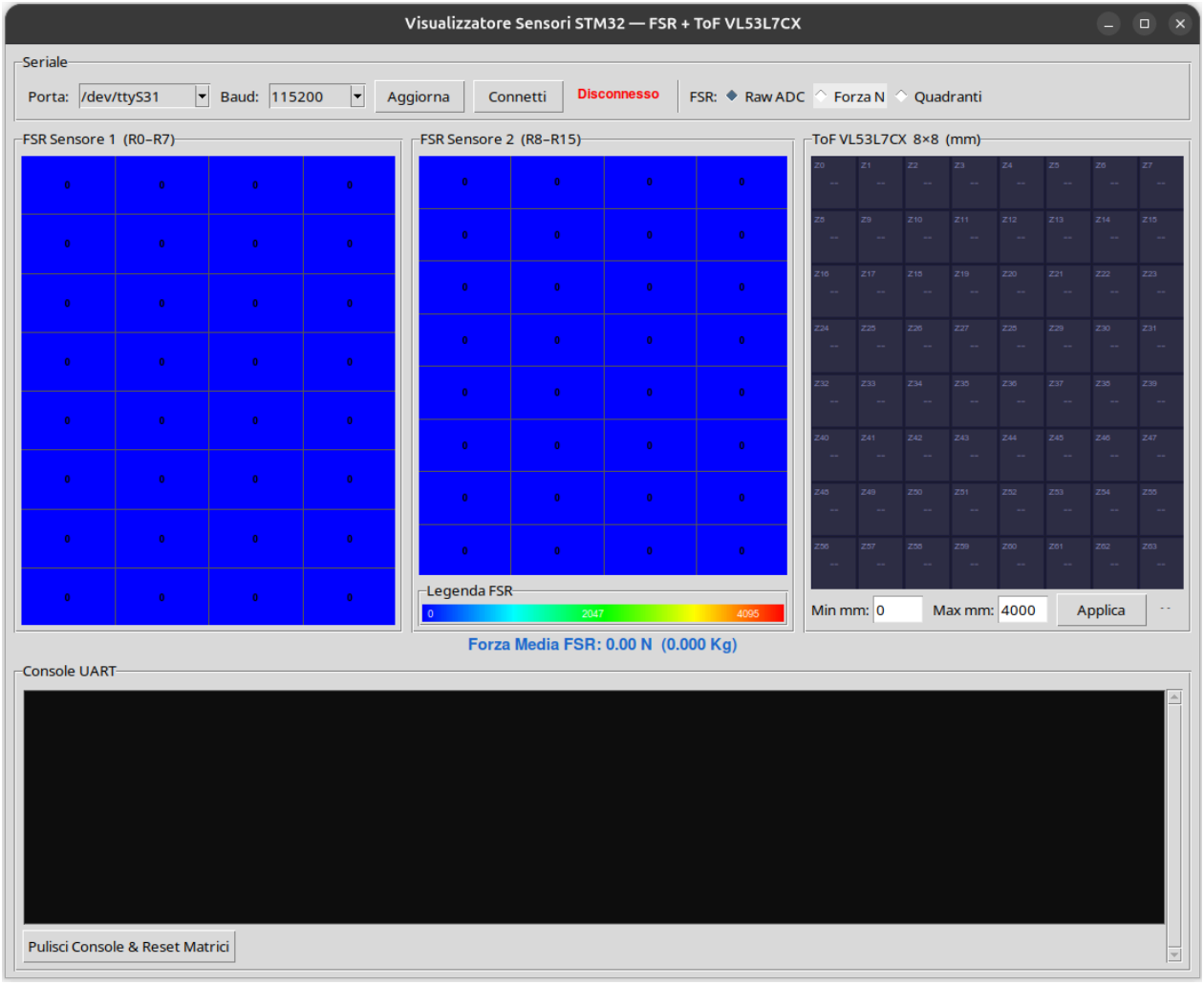


Figure 6.4: Desktop telemetry visualizer, showing the two FSR matrices (left) and the ToF depth map (right) alongside a live mean-force readout and a raw console. The FSR heatmaps are colour-mapped along a blue–green–yellow–red ramp, with the quadrant containing the current peak reading outlined; the ToF map uses a separate near-to-far colour ramp with invalid zones shown in dark grey.

7. Preliminary Functional Demonstration

Before reporting on the behavior of the assembled control loop, this chapter first describes the calibration procedures carried out on the two sensor modalities – the ToF sensor and the FSR tactile matrices – since a meaningful functional demonstration of the gripper depends on both sensors returning trustworthy readings in the first place.

7.1 ToF Sensor Calibration

The palm-mounted VL53L7CX is factory self-calibrated and does not strictly require further calibration to operate. However, because the sensor is mounted behind the protective front window of the gripper's shell, described in Section 4.2, a portion of the emitted infrared signal reflects internally off this cover before reaching the target, a phenomenon referred to as crosstalk. As shown in Figure 7.1, at short range in particular, this reflected signal can be comparable to, or larger than, the genuine signal returned from an actual object, causing the sensor to report false, artificially close targets rather than the real distance to the object in front of it.

To correct for this, a crosstalk calibration was performed using ST's dedicated calibration routine guidelines. As shown in Figure 7.2, a flat target of known reflectance was placed at a fixed known distance within the sensor's field of view, and the calibration function was run to build a per-zone compensation table, characterizing the amount of internally reflected signal specific to this gripper's cover geometry. This compensation data is stored and loaded back into the sensor at every power-up, so that subsequent ranging measurements are corrected for the cover-glass reflection before being reported over I^2C. Once calibrated in this way, the sensor's distance readings across its 8×8 zones were checked against a set of known reference distances and found to track them consistently, confirming that the crosstalk correction is effective for the specific mounting geometry used in this design.

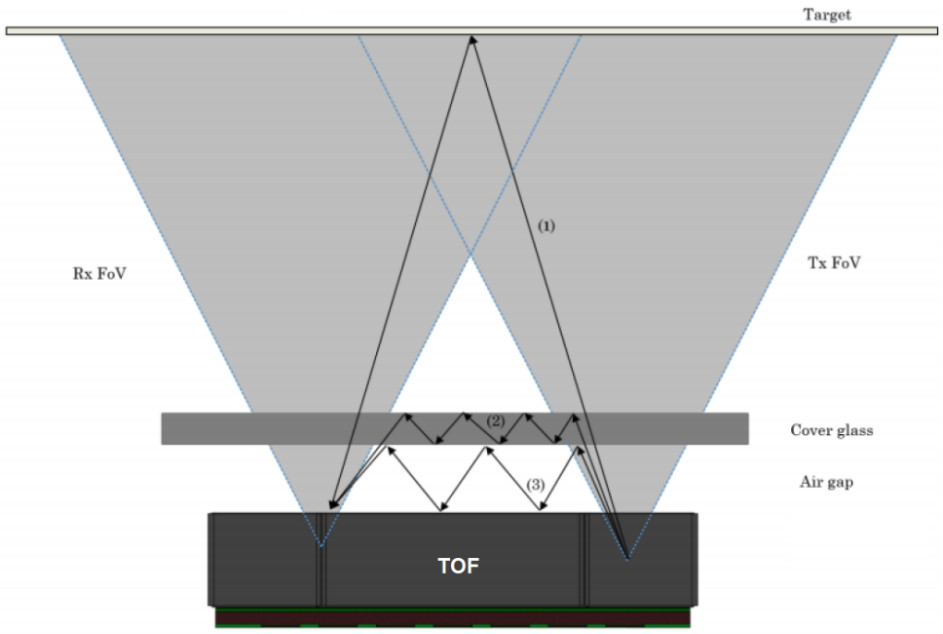


Figure 7.1: The crosstalk phenomenon in the ToF sensor, showing the reflected signal that can interfere with the cover glass.



(a) (b)

Figure 7.2: Calibration setup

7.2 FSR Sensor Calibration

Because the FSR tactile matrices used in this design are, as discussed in Section 5.3.2, subject to unit-to-unit variation, the reading circuit relies on a single divider trimmer, introduced in Section 5.4.2, to bring the sensors' combined response into a usable voltage range. This is made possible since both matrices share the same voltage divider stage through the row/column multiplexing

scheme described in Section 5.4. Calibration therefore consisted of a single adjustment applied jointly to both fingers' sensing chains. In Figure 7.3, a reference weight of 2 kg – matching the target grasping force established in Section 3.1 – was applied directly onto the sensing area of one finger's FSR matrix, and the trimmer potentiometer was adjusted while monitoring the voltage across it with an oscilloscope, until the voltage drop across it reached 3.3 V under this reference load. Because this single trimmer setting governs the divider used by both matrices, the same calibrated configuration was then verified by repeating the reference-weight test on the second finger, confirming that the shared setting produced a comparably usable response on both sensing chains.

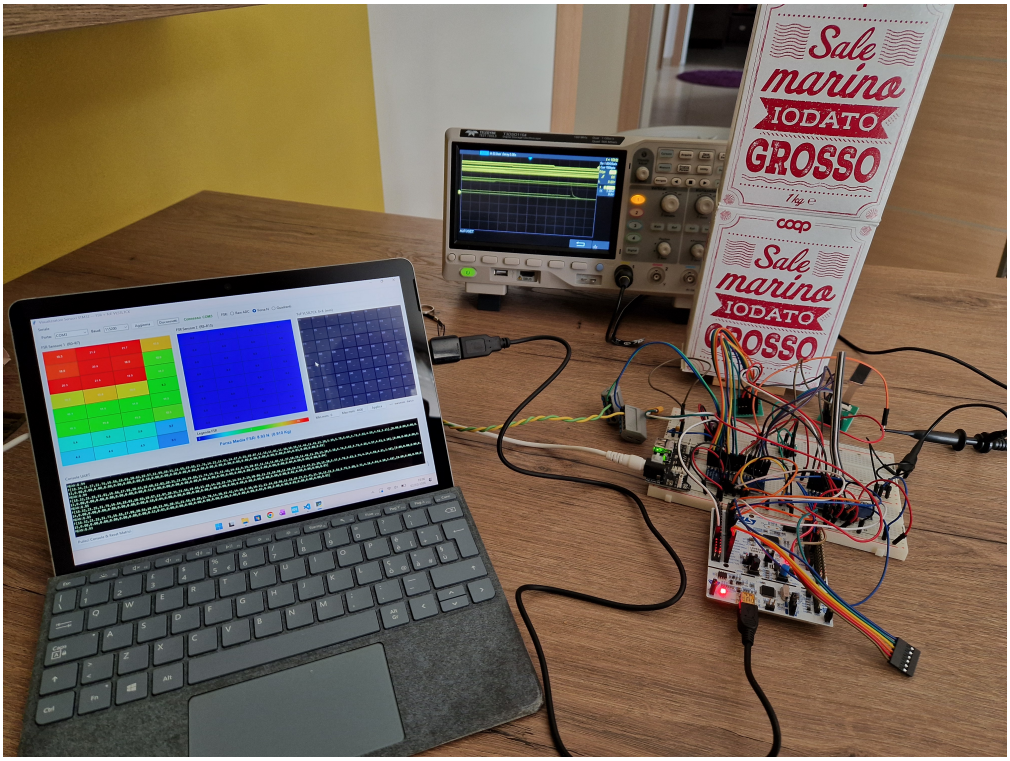


Figure 7.3: FSR sensor calibration: a 2 kg reference weight is applied to the sensing area of one finger's FSR matrix, while the voltage divider trimmer is adjusted to bring the voltage drop across it to 3.3 V. This single trimmer setting governs the divider used by both matrices, and was verified by repeating the test on the second finger.

7.3 System-Level Functional Demonstration

With both sensors calibrated as described above, the complete control loop – tactile sensing, ToF sensing, and the embedded admittance controller described in Chapter 6 – was exercised on the NUCLEO prototype. Following manual tuning of the admittance controller's virtual mass-spring-damper parameters, the closed-loop system was demonstrated to behave correctly. The fingers closed with a visibly compliant response when contact was made with an object, backing off or reducing applied force in response to resistance rather than continuing to close rigidly, consistent with the compliant behavior the admittance law is designed to produce. The ToF sensor was likewise demonstrated to report distances correctly during this testing, with its readings changing consistently as objects were moved into and out of its field of view, in line with the calibration described in Section 7.1. Finally, all sensor data – the tactile pressure maps from both fingers and the ToF sensor's range readings – together with the controller's internal state, were successfully streamed over CAN and displayed in real time on a desktop monitoring application, confirming that the communication architecture described in Chapter 3 supports simultaneous real-time control and telemetry without perceptible interference between the two.

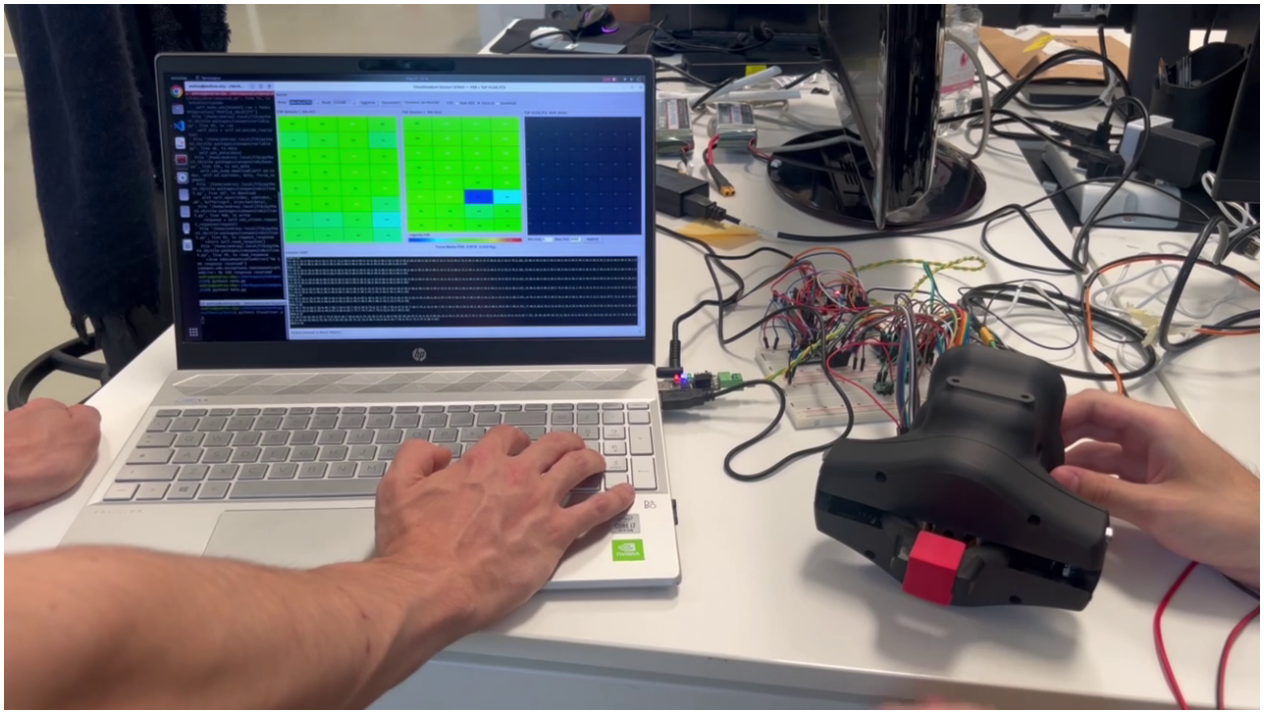


Figure 7.4: The complete control loop – tactile sensing, ToF sensing, and the embedded admittance controller – was exercised on the NUCLEO prototype. The fingers closed with a visibly compliant response when contact was made with an object, and the ToF sensor reported distances correctly as objects were moved into and out of its field of view. All sensor data and the controller's internal state were streamed over CAN and displayed in real time on a desktop monitoring application.

8. Virtual model development in Isaac Sim

Having described the physical gripper and its embedded sensing and control architecture in the previous chapters, this chapter turns to the second pillar of the platform introduced in Chapter 1: a simulation environment that reproduces the gripper, its sensor modalities, and eventually the full manipulator it is mounted on, closely enough to support training a learned grasping and pick-and-place policy. The virtual model developed here is built in NVIDIA Isaac Sim and used, together with Isaac Lab, as the environment in which the reinforcement learning policy described in Chapter 9 is trained.

The development of this virtual model followed a deliberately staged approach, rather than building the full arm-and-gripper system in a single step. The gripper was first modeled and instrumented with its sensors – the FSR tactile matrices and the ToF proximity sensor – as a standalone articulated object, so that its kinematics, contact behavior, and sensor outputs could be inspected and debugged in isolation, without the added complexity of a six-joint manipulator moving it around. Only once this standalone sensor modeling was judged structurally sound was the end effector mounted onto the full robotic arm assembly, forming the complete manipulator used for policy training. This chapter follows that same order: Sections 8.1–8.3 describe the standalone gripper model and its sensors, Section 8.4 describes its integration onto the robotic arm, and Sections 8.5 discuss domain randomization.

8.1 Modeling the gripper

The gripper's kinematic and dynamic representation is imported directly from a URDF description generated from the CAD assembly presented in Chapter 4. This preserves the prim hierarchy of the mechanical design – the base link and the two finger links, together with their associated joints – as a single articulated object, and it is what allows the same physical design decisions discussed in Section 3.5 (the rack-and-pinion transmission, the single actuated degree of freedom, the palm-mounted ToF sensor) to be reflected directly in simulation. The root of the articulation is defined by a fixed joint carrying both an articulation-root API and PhysX's articulation extension, which

anchors the gripper to the world frame and lets the physics engine treat it as one coupled multibody system rather than a collection of independently simulated parts.

The single mechanical degree of freedom of the physical rack-and-pinion transmission, discussed in Section 4.5, is reproduced in simulation as two prismatic joints – one per finger – both translating along the same local axis. Only one of the two, corresponding to the actuated finger, carries an active position drive, implemented as a force-type drive with a stiffness of 5000 N/m and a damping of 50 Ns/m, capped at a maximum force of 20 N, with its travel bounded between 0 and 0.0455 m – matching the finger stroke established as a design requirement in Section 3.1. The second joint is not driven independently; instead, it is constrained through a PhysX mimic-joint relationship to the driven joint, with a gearing factor of -1, so that it moves by exactly the same amount in the opposite direction at every simulation step. Rather than explicitly simulating the pinion's gear teeth meshing with both racks – which would be computationally expensive and difficult to tune – the mimic-joint constraint reproduces the physical coupling enforced by the real transmission directly at the joint level, giving the correct kinematic behavior (one motor, two symmetric fingers) without paying the cost of simulating gear contact.

Mass properties are likewise assigned from the actual CAD design, in line with the material choices discussed in Section 4.1. The base link, incorporating the carbon-fiber mounting plate and the printed shell, is assigned a mass of approximately 0.427 kg, while each of the two symmetric finger links carries a mass of approximately 0.063 kg. Both finger links additionally carry PhysX's contact-reporting extension, configured with a reporting threshold of zero so that every contact event, however small, is exposed to the simulation – this is what makes the tactile sensing scheme described next possible.

8.2 Simulating the FSR Tactile Arrays

Reproducing the tactile sensing modality in simulation is the most novel, and arguably the most consequential, part of the virtual model, since it is what ultimately determines whether a policy trained on simulated tactile data can meaningfully inform grasping behavior on the real hardware. Consistent with the discussion in Section 2.5, tactile simulation is considerably less mature than rigid-body simulation, and the strategy adopted here follows the penalty-based, contact-point family

of approaches rather than the optical-rendering approach used for vision-based sensors such as GelSight, since the physical sensor being reproduced is a resistive matrix and not a camera-based device.

Concretely, each finger's tactile matrix is represented in the scene, as shown in Figure 8.1, as a dense grid of individual taxel prims, arranged across eight rows and four columns – 32 taxels per finger, 64 in total across the two matrices – matching the 8×4 layout of the real FSR sensor described in Section 5.3.2. Each taxel is not a purely geometric marker but is instantiated as its own small rigid body, carrying both a collision shape and a rigid-body API, and it is rigidly attached to its parent finger link through a dedicated fixed joint. Giving every taxel its own rigid body is what allows PhysX's per-body contact reporting to be queried independently for each of the 64 cells, yielding a full spatial force map across the fingertip surface.

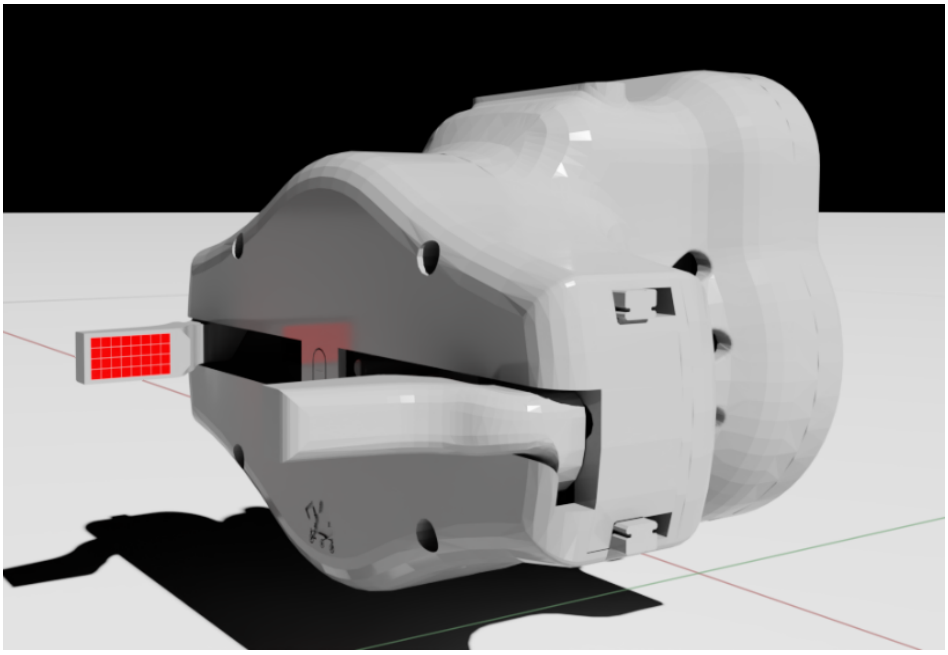


Figure 8.1: The FSR tactile arrays are represented in the scene as a grid of 32 individual taxel prims per finger, each instantiated as its own rigid body and rigidly attached to the parent link through a fixed joint. The taxels are arranged in eight rows and four columns, matching the layout of the real sensor.

This modeling choice is carried through directly into the training environment. The tactile signal used by the reinforcement learning agent, described further in Chapter 9, is obtained through two

dedicated contact sensors, one per finger, that monitor exactly the 32 taxel bodies of each matrix and report the net contact force exchanged specifically with the probe object, filtered so that incidental contact with other parts of the scene does not register as a tactile reading.

8.3 Simulating the ToF Sensor

The palm-mounted proximity sensor introduced in Section 5.3.1 is reproduced in the standalone gripper model as a dedicated sensor prim attached at the sensor's real mounting location on the palm, between the two fingers, exactly as positioned on the physical hardware described in Section 4.2. Once the gripper was integrated into the full training environment (Figure 8.2), this prim is driven by Isaac Lab's ray-casting sensor infrastructure, configured to emulate the VL53L7CX's multi-zone ranging behavior. The ray pattern is configured with eight vertical channels spanning a $\pm 22.5^\circ$ vertical field of view, and a matching horizontal sweep at a resolution of about 6.43° across the same $\pm 22.5^\circ$ range, together producing an 8×8 grid of independent range measurements – 64 values in total – directly analogous to the real sensor's 8×8 zone output described in Section 5.3.1. The maximum sensing range is set to 4 m, comfortably covering the distances relevant to pre-grasp detection of the probe.

8.4 Integration with the Robotic Arm

Once the standalone gripper model and its sensors were judged structurally functional, the end effector was mounted onto the full robotic arm assembly used by ProjectRED. The arm itself is a six-joint serial manipulator, imported into the scene as its own articulated asset together with the gripper. The end effector's base link is attached to the arm's last link through a fixed joint, producing a single combined articulation with seven actuated degrees of freedom in total – the arm's six revolute joints plus the gripper's one prismatic degree of freedom – matching exactly the action space used by the reinforcement learning policy described in Chapter 9.

This combined arm-and-gripper articulation is what is actually spawned in the Isaac Lab training environment, configured with gravity enabled, self-collision detection enabled at the articulation level, and realistic contact and collision margins consistent with the scale of the hardware. The

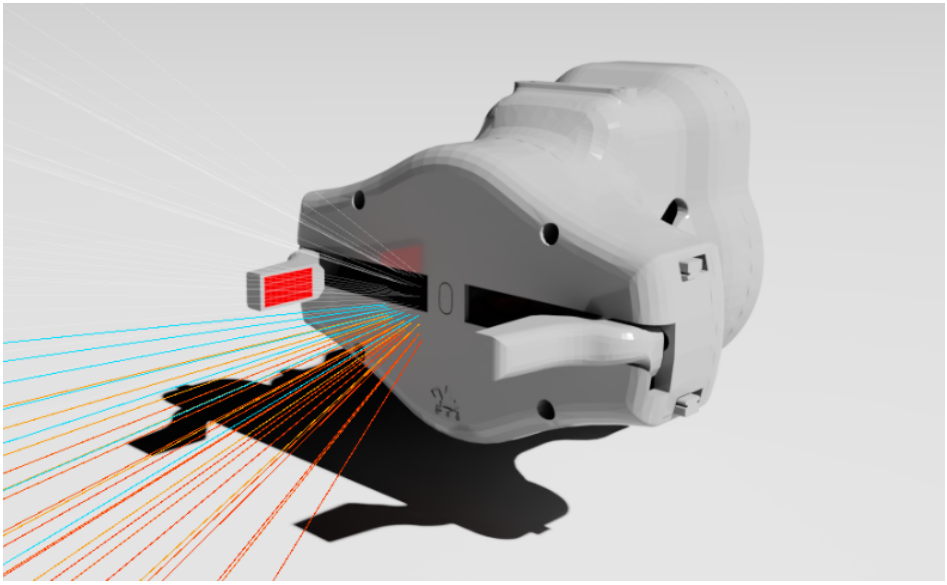


Figure 8.2: The palm-mounted ToF sensor is represented in the simulation by a dedicated Lidar prim, attached at the sensor's real mounting location on the gripper's palm. The sensor's ray-casting pattern is configured to emulate the VL53L7CX's multi-zone ranging behavior, producing an 8×8 grid of independent range measurements, directly analogous to the real sensor's output.

two halves of the articulation are actuated through separate implicit actuator models with distinct dynamic characteristics, reflecting the fact that the arm's joints and the gripper's finger joint are driven by physically different actuators: the six arm joints are configured with a stiffness of approximately 206 N/m and a damping of approximately 3.3 Ns/m, with an effort limit of 400 Nm, values intended to approximate the dynamic response of the arm's own joint controllers, while the gripper's finger joint is configured separately, with a much lower stiffness of approximately 0.55 N/m and damping of approximately 0.0002 Ns/m and an effort limit of 20 N.

Combining a six-joint arm with a gripper whose fingers and finger-mounted sensors sit in close proximity to the arm's own last links introduces a practical simulation concern: spurious self-collisions between adjacent, physically connected links, which would otherwise be flagged by the physics engine even though they correspond to normal, expected contact between consecutive joints. To avoid this, collision detection is explicitly disabled between every pair of adjacent links along the kinematic chain – from the arm's base link through each successive link, and finally between the arm's last link and the gripper's base link – using PhysX's filtered-pairs mechanism, applied

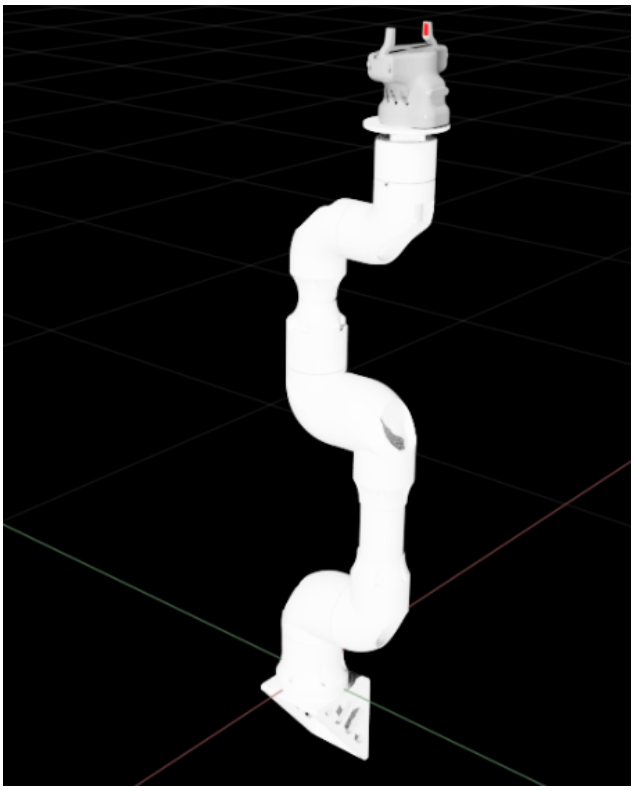


Figure 8.3: The robotic arm assembly, with the gripper mounted on the end effector.

identically across every parallel environment instance used during training. Genuine self-collisions elsewhere on the arm, which would indicate an undesirable or unsafe configuration rather than expected joint-to-joint contact, are instead monitored through a dedicated contact sensor spanning all of the arm's links, whose readings are used, as described in Chapter 9, to detect and penalize self-collision events during training.

Mounting the gripper on the full arm, was a necessary step toward the eventual ERC deployment discussed in Chapter 1. The Probing Task requires the rover's arm to actively search for, approach, and grasp a probe within its workspace, which is a behavior that cannot be captured by a gripper alone, since the reaching and approach motion is itself part of the skill the policy needs to learn. The seven-dimensional joint configuration resulting from this integration together with the sensor suite described in Sections 8.2 and 8.3, forms the complete observation and actuation interface used for policy training in Chapter 9.

8.5 Domain Randomization

Domain randomization has been implemented targeting the pose of the probe object, the target of the Probing Task. At the start of every training episode, the probe's position and orientation are re-sampled rather than reset to a fixed pose: its horizontal position is drawn uniformly within a rectangular region in front of the gripper, spanning roughly 10 *cm* along the depth axis and 20 *cm* laterally, while its height above the ground is held fixed at its resting value. Its orientation is randomized about the vertical axis only. A yaw angle is sampled uniformly over the full 360° range, while roll and pitch are left at zero, so that the probe always lies flat on the table but is presented to the gripper at an arbitrary heading on every episode. Because each of the parallel training environments is reset independently, this per-episode randomization allows a single training run, executed across many environments simultaneously, to expose the policy to a wide range of probe positions and orientations without requiring a correspondingly larger number of training episodes to do so sequentially.

This choice of randomization axis directly targets the operational uncertainty relevant to the ERC Probing Task introduced in Chapter 1. On the real Mars Yard, the exact position and heading at which a probe will be found cannot be known in advance, so training the policy against a distribution of plausible probe poses, instead of a single fixed one, is intended to produce an approach-and-grasp behavior that generalizes across the range of placements the rover is likely to encounter.

Other axes of randomization discussed conceptually in Section 2.5 – object mass and friction, tactile contact stiffness and damping, and ToF sensor noise – have not yet been implemented at the time of writing. The probe's mass and its collision and contact properties are currently fixed and the tactile and ToF sensor models described in Sections 8.2 and 8.3 are used as configured, without injected noise or per-episode variation in their response. Extending domain randomization to these additional parameters is identified as part of the roadmap discussed in Chapter 10

9. Reinforcement Learning Policy Training

9.1 Task Formulation

The grasping problem addressed in this thesis is formulated as a Markov Decision Process (MDP), following the standard formalism used throughout the reinforcement learning literature discussed in Chapter 2. This section defines the three elements of that formulation as actually implemented in the Isaac Lab training environment – the observation space, the action space, and the structure of an episode – before the reward function that ties them together is discussed separately in Section 9.2.

9.1.1 Observation Space

At every control step, the policy receives a 154-dimensional observation vector, assembled by concatenating five groups of normalized signals directly available from the sensors and state variables described in Chapter 8. The largest component, 64 values, is the flattened 8×8 range image produced by the simulated ToF sensor, normalized against the sensor's 4 m maximum range. The next 64 values are the tactile signal: the 32 per-taxel contact-force magnitudes from the left finger's FSR matrix, followed by the 32 from the right finger's, each normalized against a reference force of 20 N – matching the maximum drive force configured for the gripper's finger joint in Section 8.4. The remaining 26 values are proprioceptive and task-relevant state: the 7 joint positions and 7 joint velocities of the combined arm-and-gripper articulation, the end effector's position, the probe's position, the vector from the end effector to the probe, and the fixed placement-goal position, each expressed relative to the environment's own origin. All five groups are concatenated into a single flat vector, and any residual invalid values (NaN or infinite, which can arise transiently from the ray-caster or from numerical instability) are replaced with safe defaults before being passed to the policy, so that the observation is always well-formed regardless of momentary sensor glitches.

9.1.2 Action Space

The policy outputs a 7-dimensional continuous action vector, bounded to $[-1, 1]$ in every component: six actions correspond to the arm's six revolute joints, and the seventh corresponds to the gripper's single degree of freedom. Rather than commanding absolute joint positions directly, each action is interpreted as an incremental position delta added to the current joint position target. The six arm actions are scaled by a fixed step size of 0.03 rad per control step, while the gripper action is scaled by a smaller step size of 0.01 m and applied symmetrically to both finger joints, consistent with the single-DOF, mimic-jointed transmission described in Section 8.1. The resulting joint position targets are clamped against the articulation's own soft joint limits before being sent to the implicit position drives, which prevents the policy from commanding physically infeasible configurations regardless of how large or repeated its raw action outputs are.

9.1.3 Episode Structure

Each episode follows the reach–grasp–lift–place structure introduced conceptually in Chapter 1 for the ERC Probing Task. Starting from a fixed initial arm configuration with the probe placed at a randomized pose in front of the gripper (Section 8.5), the policy must approach the probe, close the fingers around it, lift it clear of the floor, and carry it to a fixed placement-goal position located elsewhere in the workspace. An episode terminates successfully once the probe's position comes within 4 cm of this placement goal; it terminates in failure either if the probe falls or if a self-collision is detected, per the self-collision monitoring described in Section 8.4; and it is otherwise truncated after a fixed episode length of 25 simulated seconds if neither condition is reached first. This termination logic gives every episode an unambiguous outcome – success, failure, or timeout – which is tracked directly for the success-rate metrics reported in Section 9.5.

9.2 Reward Design

The reward function combines four phase-specific components – alignment, grasp, lift, and place – with three penalty terms, weighted and gated through an explicit, automatically-advancing four-stage curriculum.

Alignment reward. Rather than rewarding proximity to the probe using privileged, ground-truth Euclidean distance, the alignment term is derived entirely from the simulated ToF sensor's own 8×8 range grid making this the one reward component that could, in principle, be computed identically on the real hardware. The 8×8 grid is split into a central 4×4 block, treated as the region directly in front of the gripper, and an outer ring of edge pixels. A reach term, $1 - \tanh(5 \cdot d_{\min})$, rewards minimizing the closest reading within the central block; a centering term rewards the central block reading closer than the surrounding edges, encouraging the probe to be framed in the middle of the sensor's field of view rather than off to one side; a top-down term combines the end effector's own forward-kinematics orientation (rewarding the gripper pointing downward) with a ToF-based tilt-symmetry measure (comparing top/bottom and left/right edge readings, via $\exp(-10 \cdot \text{asymmetry})$) to encourage a perpendicular approach to the probe; and a fourth term uses the local gradient structure of the ToF grid to reward alignment with the probe's own geometry. These three shaping terms are combined into a single modulation factor and multiplied by the reach term, so that alignment reward only accrues once the gripper is actually getting close.

Grasp reward and premature-closure penalty. Grasp reward is only granted once the end effector is very close to the probe according to the ToF sensor (below 3 cm) and the gripper is actively closing, and it rewards double contact – both fingers' tactile matrices registering a force above a 0.2 N threshold, chosen to reject spurious low-level contact noise from the physics engine – three times as much as single-finger contact. A separate penalty term is applied whenever the gripper closes while the ToF sensor reports the probe as not yet close (beyond 5 cm), discouraging indiscriminate closing regardless of proximity.

Lift and place rewards. A grasp is considered secure, for reward purposes, once the ToF sensor confirms proximity, both fingers register contact, and the probe has been lifted more than 1.5 cm off the table. The lift reward scales with height gained up to a 10 cm target, and the place reward, active only once the probe has cleared 4 cm, rewards reducing the remaining distance to the fixed placement goal.

Additional penalties. Two further, stage-independent penalties are applied: a fixed -0.20 penalty if any part of the arm collides with the floor, and a small constant time penalty of -0.005 per step.

The total reward is clamped to $[-1, 1]$ at every step and sanitized against NaN or infinite values before being returned to the training loop.

The weighting of these components is not fixed for the whole training run, but follows an explicit four-stage curriculum, advanced automatically by a dedicated monitoring callback. Table 9.1 summarizes the reward composition and success criteria for each stage, which are designed to gradually increase the task's difficulty while still providing a dense shaping signal at every step. The curriculum is adaptive, advancing only when the policy demonstrates sufficient competence at the current stage.

Stage	Name	Reward	Success
0	Alignment	100% align.	ToF close to probe
1	Grasp	30% align. + 70% grasp	Stage 0 + both fingers contact
2	Lift	15% align. + 35% grasp + 50% lift	Stage 1 + above lift threshold
3	Pick & place	10% align. + 30% grasp + 20% lift + 20% place	Probe within 4 cm of goal

Table 9.1: Reward weighting and success criteria across the four curriculum stages.

At every stage, floor-collision and step penalties are always active, and from Stage 1 onward the premature-closure penalty is active as well. The callback tracks a rolling window of the last 100 episode outcomes and checks, every 2,048 steps, whether the success rate over that window has reached 70%. Once this condition is met, the environment advances to the next stage and the outcome window is cleared, so that the success-rate criterion for the new stage is evaluated on fresh data.

9.3 Algorithm and Training Setup

The model free pick-and-place policy is trained using Proximal Policy Optimization (PPO), chosen for its established use in Isaac Lab-based manipulation tasks. It handles continuous action spaces well and scales naturally to the large batch sizes produced by massively parallel GPU simulation. At each update, PPO improves the policy parameters θ by maximizing a clipped surrogate objective over the batch of trajectories $\mathcal{D}_k$ collected under the previous policy π_{θ_k},

$$\theta_{k+1} = \arg\max_{\theta} \frac{1}{|\mathcal{D}_k|T} \sum_{\tau \in \mathcal{D}_k} \sum_{t=0}^{T-1} \min\left(\frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_k}(a_t \mid s_t)} A^{\pi_{\theta_k}}(s_t, a_t),\ g\left(\epsilon, A^{\pi_{\theta_k}}(s_t, a_t)\right) \right) \tag{9.1}$$

where $A^{\pi_{\theta_k}}(s_t, a_t)$ is the advantage estimate for state-action pair (s_t, a_t) under the previous policy, and $g(\epsilon, A)$ clips the probability ratio to the interval $[1 - \epsilon, 1 + \epsilon]$, preventing any single update from moving the policy too far from π_{θ_k} regardless of how large the estimated advantage is. The clipping threshold ϵ corresponds directly to the clipping range reported in Table 9.2, and this clipped objective is what allows PPO to perform multiple epochs of gradient updates on the same batch of rollout data. The implementation used is Stable-Baselines3's PPO, wrapped around the custom `GripperPpoEnv` environment described in Sections 8.4 and 9.1 through Isaac Lab's `Sb3VecEnvWrapper`, and further wrapped in a `VecMonitor`, which tracks standard per-episode reward and length statistics alongside the environment's own custom metrics.

Training runs across 256 parallel environment instances simulated simultaneously on the GPU, each spaced 12.5 m apart, with PhysX's rigid-body patch budget increased to accommodate the large number of individually contact-reporting FSR taxels introduced in Section 8.2. Each control step corresponds to two physics steps (a decimation factor of 2), and episodes run for up to 25 simulated seconds before being truncated, per the termination logic defined in Section 9.1.

The optimization hyperparameters used for PPO are summarized in Table 9.2. The learning rate follows a linear decay schedule rather than a fixed value, allowing larger, more exploratory updates early in training while stabilizing learning as the policy approaches convergence; the discount factor and GAE parameter are set slightly below the more common defaults, consistent with the relatively short, densely-shaped episodes discussed in Section 9.2; and a comparatively high entropy coefficient is used to sustain exploration throughout training, which was found necessary given the multi-phase nature of the task.

The policy processes its 154-dimensional observation through the custom multi-modal feature extractor described in Section 9.4, producing a 256-dimensional shared representation that feeds into separate actor ($[128, 64]$) and critic ($[128, 128]$) heads with ReLU activations. Rather than a fixed weighting applied for the whole run, the reward is instead shaped by the four-stage curriculum described in Section 9.2, whose progression is entirely performance-gated: no fixed total-timestep budget is set for training, and the run is instead allowed to continue indefinitely, terminating only

Hyperparameter	Value
Rollout buffer size (n_{steps})	2,048 steps/environment
Mini-batch size	1,024
Epochs per update	10
Learning rate	linear decay, $3 \times 10^{-4} \rightarrow 1 \times 10^{-5}$
Discount factor γ	0.99
GAE parameter λ	0.95
Clipping range	0.2
Max gradient norm	0.5
Entropy coefficient	0.05
Parallel environments	256
Random seed	42

Table 9.2: PPO hyperparameters used for training.

once the curriculum-advancement callback detects that the final stage's success-rate target has been met. This ties the length of training directly to how quickly the policy actually masters each phase of the task, rather than to a predetermined schedule.

Three callbacks run alongside training: one logs custom episodic metrics – end-effector-to-probe and probe-to-goal distances, the fraction of environments actively gripping, and per-batch timeout and success rates; a second logs the run's reward-weight configuration once at the start, so that the specific settings behind any given training curve remain traceable; and the curriculum-advancement callback itself, described in Section 9.2, which additionally logs the current curriculum stage at every step. The final trained policy is saved once the callback signals that training is complete.

9.4 Policy Architecture

The 154-dimensional observation described in Section 9.1 is split into three streams as it enters the network. The 64 ToF values are reshaped into an 8×8 single-channel image, matching the spatial layout of the sensor's own ranging grid described in Section 8.3, and passed through a small convolutional branch: a first 3×3 convolution and normalization layer followed by max-pooling, then a second convolutional layer, before being flattened and projected down to a 64-dimensional

feature vector. The 64 tactile values are similarly reshaped, but as two 8×4 single-channel images – one per finger – stacked into a two-channel input, so that the convolutional filters can learn spatial patterns within each finger's taxel grid while also relating the two fingers' contact patterns to one another; this branch mirrors the ToF branch's convolution–normalization–pooling structure and likewise produces a 64-dimensional feature vector. The remaining 26 proprioceptive and task-state values (joint positions and velocities, end-effector and probe positions, and the placement goal) are processed by a small fully-connected branch instead, since this data has no natural 2D spatial structure to exploit, and are projected to a further 64-dimensional feature vector.

The three 64-dimensional feature vectors are concatenated and passed through a final fusion layer that projects them into a single 256-dimensional shared feature representation. Normalization is applied throughout using group and layer normalization rather than batch normalization, a choice suited to reinforcement learning training, where batch statistics can be unstable across the sequence of on-policy rollouts collected during PPO updates. From this shared 256-dimensional representation, separate actor and critic heads branch off as standard multi-layer perceptrons – a smaller two-layer head (128, 64 units) producing the policy's action distribution, and a slightly larger two-layer head (128, 128 units) producing the value function estimate – both using ReLU activations, following the conventional PPO actor-critic design.

This convolutional, modality-aware architecture is a deliberate departure from feeding the flat observation vector directly into a generic multi-layer perceptron: by giving the ToF and tactile channels their own spatially-aware processing branches before fusing them with the proprioceptive state, the network is structured to recognize local patterns – such as a cluster of nearby ToF readings indicating the probe's direction, or a localized force response on specific taxels indicating where contact is occurring – that a fully-connected network would have to learn to reconstruct from an unstructured concatenation of the same 128 raw values.

The overall network is kept deliberately compact relative to what is typically seen in manipulation-RL literature, where feature extractors and policy heads with several times as many parameters are common. This is not an incidental simplification but a design constraint tied to the long-term goal, discussed in Section 1.3, of eventually running policy inference directly on the STM32F446 microcontroller described in Section 5.5, rather than depending on a host computer at deployment time. The microcontroller's limited flash and RAM, and the absence of any dedicated hardware

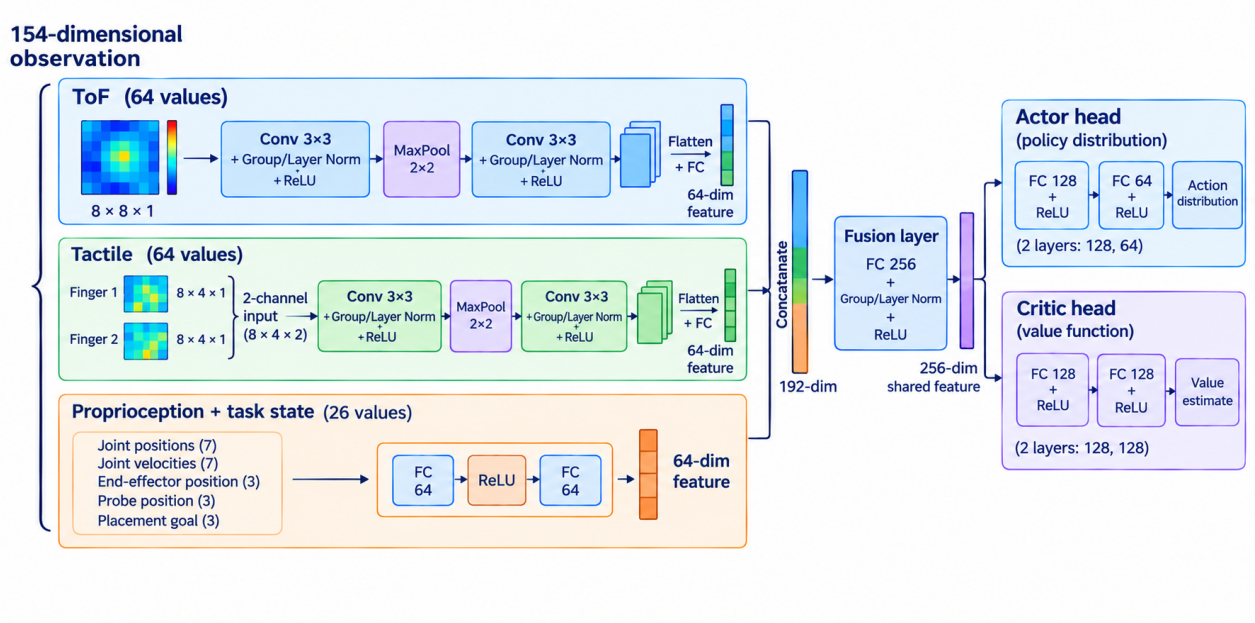


Figure 9.1: The policy architecture is a convolutional, modality-aware design that processes the ToF and tactile channels through separate convolutional branches before fusing them with the proprioceptive state. The resulting shared representation is then passed to separate actor and critic heads for action selection and value estimation.

acceleration for neural network inference, mean that only a small model – a handful of narrow convolutional layers and small fully-connected heads, as used here – could realistically be quantized and executed.

9.5 Training Progress and Preliminary Results

Training is still in progress at the time of writing, and this section reports the behavior observed and the learning trends recorded up to the current checkpoint.

Qualitatively, the policy has clearly learned the approach behavior targeted by the first stage of the curriculum described in Section 9.2. Guided by the ToF sensor's own range readings, the arm reliably identifies the direction of the probe and drives the end effector toward it, correctly orienting and closing the distance regardless of the probe's randomized position and heading. Grasp behavior, corresponding to the second curriculum stage, is present but not yet reliable. In a small number of episodes, the policy has been observed to successfully close both fingers around the probe and

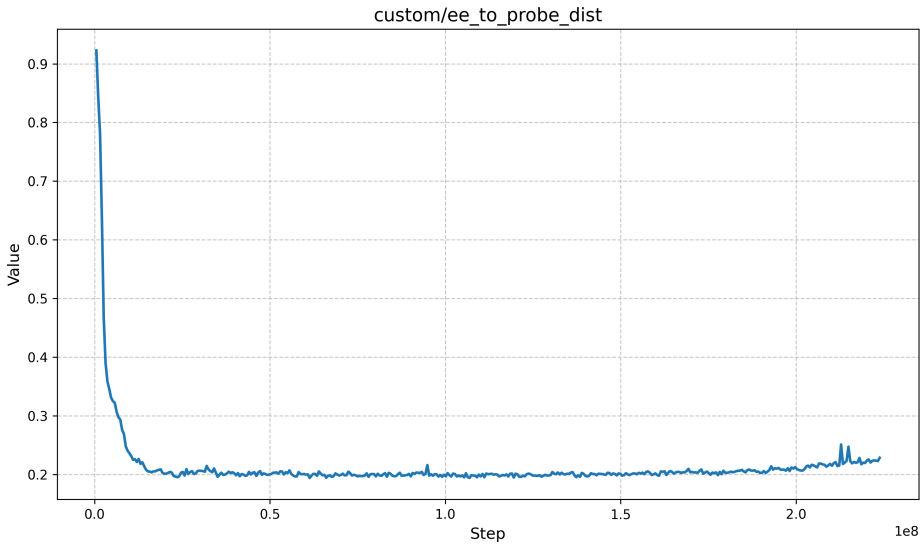
briefly lift it off the ground, but these successes are sporadic, occurring occasionally and not across the majority of episodes. This pattern is consistent with a policy that has not yet accumulated enough simulated experience at this stage to reliably clear the 70% success-rate threshold required by the curriculum callback (Section 9.2) to advance further, rather than indicating a fundamental flaw in the task or reward formulation.

This qualitative picture is corroborated by the training curves logged to TensorBoard over the course of the run. The mean distance between the end effector and the probe shows a clear, consistent downward trend across training, reflecting the increasingly accurate approach behavior described above, and the mean episode reward increases correspondingly as training progresses, tracking the growing contribution of the alignment reward term once the policy began reliably closing this distance (Figure 9.2). Neither curve has yet plateaued at the time of writing, suggesting that continued training is likely to further improve both metrics.

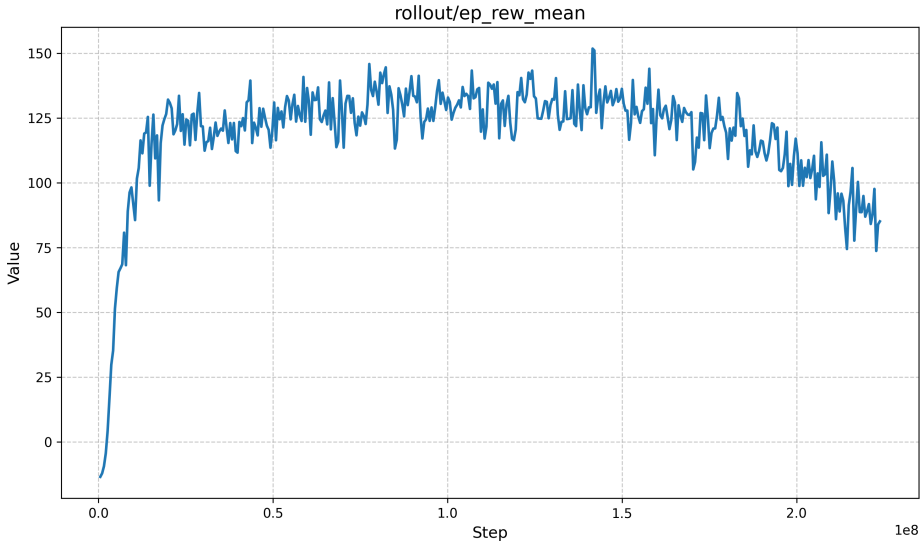
The PPO optimizer minimizes a single scalar objective combining three terms: the clipped policy-gradient loss, the value-function loss weighted by `vf_coef`, and the entropy bonus weighted by `ent_coef` (Section 9.3),

$$L_{total} = L_{policy} + c_{vf} \cdot L_{value} - c_{ent} \cdot H(\pi). \tag{9.2}$$

The total loss recorded during training (Figure 9.3) shows a noisy but downward trend over the course of the run. This overall trend is not, on its own, a clean convergence signal, since it results from the combination of three terms that are each moving for different reasons. The value-function loss has been steadily increasing throughout training which is consistent with the critic continually being asked to fit the return of an evolving, non-stationary policy as it progresses through the curriculum stages described in Section 9.2. The policy loss fluctuates close to zero, as expected under PPO's clipping mechanism and the entropy term decreases steadily and close to linearly over training, indicating that the policy's action distribution is becoming less random as it specializes. Taken together with the reward and distance curves discussed above, the loss curves are consistent with an optimization process that is still actively adapting to a moving curriculum target.



(a) Mean distance between end effector and probe



(b) Mean episode reward

Figure 9.2: The mean distance between the end effector and the probe shows a consistent downward trend across training, reflecting the increasingly accurate approach behavior while the mean episode reward increases correspondingly as training progresses.

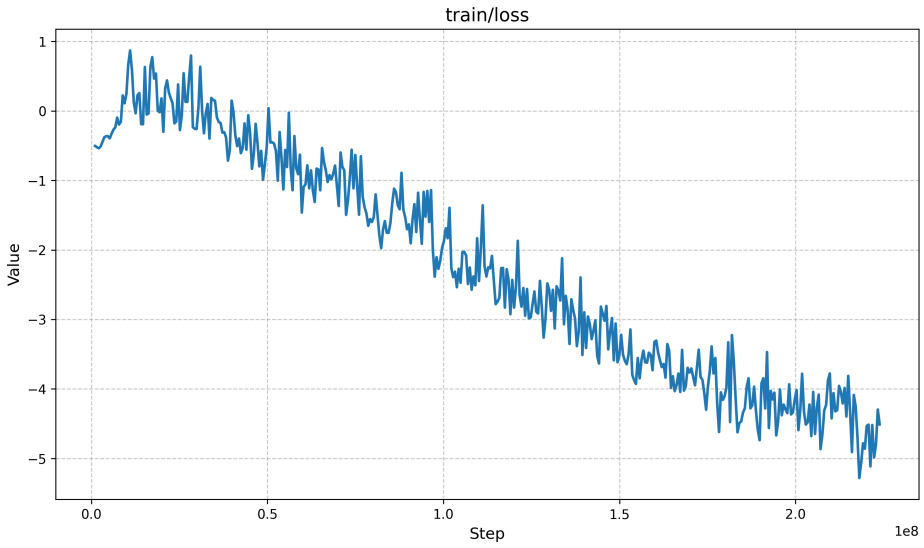


Figure 9.3: The total loss recorded during training shows a noisy but downward trend over the course of the run.

10. Conclusions and Future Work

10.1 Summary of Contributions

This thesis set out to design and build a sensorized two-finger gripper for the ERC Maintenance and Probing tasks, and to extend the resulting platform into a learning-based pipeline built on the same sensor modalities. Revisiting the contributions listed in Section 1.4, a two-finger end effector integrating dual FSR tactile matrices and a multi-zone ToF sensor was designed, built, and instrumented with an embedded, real-time sensing and control architecture. A CANopen-based communication scheme was implemented to share telemetry and control traffic on a single bus with the rover's motor driver and controller and an embedded admittance controller was implemented and demonstrated on the NUCLEO prototype. In a second phase, a sensor-faithful virtual model of the gripper, integrated with a full robotic-arm model in Isaac Sim, was developed and used to train a reinforcement learning pick-and-place policy targeting the same Probing Task the physical hardware was designed for.

10.2 Current Status

At the time of writing, the physical gripper's mechanical, electronic, and firmware layers are functional and running together as a complete system. Both sensor types have been calibrated – the ToF sensor's crosstalk compensation and the FSR matrices' voltage-divider trimmer – and, once calibrated, both were demonstrated to behave correctly. The ToF sensor reports distances consistently as objects move in and out of its field of view, and the tactile matrices produce a sensible force response under load. The embedded admittance controller was tuned specifically for the task of grasping the ERC probe, and the complete control loop was demonstrated to run in real time on the prototype, closing compliantly around the probe while streaming live over CAN all sensor telemetry to a desktop monitoring application.

On the learning side, the virtual model reproduces the gripper's kinematics, its tactile matrices at the same taxel resolution as the physical sensor, and the ToF sensor's ranging behavior, and it has

been integrated with a full model of the robotic arm to support the pick and place task the Probing Task requires. Although the training of the reinforcement learning policy is still in progress at the time of writing, the policy has clearly learned to approach the probe reliably, guided by the ToF sensor, but grasp success remains sporadic and the curriculum has not yet advanced beyond.

10.3 Limitations

Several limitations of the current work should be stated explicitly. The admittance controller has been tuned and demonstrated specifically for the probe-grasping task. It has not been formally benchmarked, and its behavior on other object types or under the panel-manipulation conditions of the Maintenance Task has not yet been characterized. The reinforcement learning policy, as noted above, had not converged at the time of writing, and the results reported reflect a training run still in progress.

The virtual model's domain randomization is also limited in scope. Only the probe's position and orientation are randomized across training episodes, while object mass, friction, and other physical properties remain fixed. Equally, no noise has been injected into either simulated sensor. The tactile taxels and the ToF range readings used during training are effectively idealized, free of the sensor noise, drift, and per-unit variation that characterize the real FSR matrices and the real VL53L7CX. Because the policy has therefore only ever been trained against clean, noise-free sensor signals and a single randomized object property, its ability to tolerate the imperfections of real sensor data, or to generalize to different object properties, has not been established and should not be assumed. No system identification or calibration of the virtual model's tactile contact response or sensor noise model against real hardware measurements has been carried out, and no sim-to-real transfer has been attempted yet.

A further limitation concerns the resolution of the ToF sensor itself. While the VL53L7CX was demonstrated to be adequate for the coarse pre-grasp detection role it plays in this thesis, its 8×8 zone output is a comparatively low-resolution representation of the space in front of the gripper, and this likely places a ceiling on how finely the probe's position and orientation can be estimated from range data alone – a factor discussed further as a candidate improvement in Section 10.4.

The slow, still-incomplete convergence observed in training is also, at least in part, attributable

to a hardware-driven design constraint. As discussed in Section 9.4, the policy network was deliberately kept small – a handful of narrow convolutional layers and compact fully-connected heads – because the long-term intent is to eventually run inference directly on the STM32F446RE microcontroller already used for the embedded control loop, and this microcontroller's limited flash, RAM, and lack of any dedicated neural-network acceleration hardware would not support a substantially larger model. This deliberately constrained network capacity likely limits how quickly the policy can fit the multi-phase reach-grasp-lift-place task, and is a plausible contributing factor to the sporadic, still-inconsistent grasp behavior reported in Chapter 9, independent of the curriculum and reward design themselves.

10.4 Future Work

The most direct next step is to let the reinforcement learning training run to completion, allowing the curriculum described in Chapter 9 to progress through full pick-and-place stages, and to report the resulting success rate once training converges. In parallel, the domain randomization strategy should be extended beyond the probe's pose to cover object mass and friction, and realistic noise models should be added to both the simulated sensors, so that the policy is exposed to a training distribution that better reflects the imperfections of the real sensors.

On the sensing side, a higher-resolution proximity sensor is worth investigating as a replacement for the VL53L7CX. A higher-resolution multi-zone ToF sensor, such as ams OSRAM's TMF8829 – a direct time-of-flight sensor supporting configurable resolutions up to 48×32 zones (1,536 measurement points) over an 80° field of view [6, 42] – would provide substantially finer spatial detail on the probe's surface and edges, and is a natural candidate for improving both object detection and orientation estimation beyond what the current sensor's central-block and edge-symmetry heuristics (Section 9.2) can extract from a 64-point range grid.

On the learning side, it is also plausible that at least part of the slow convergence observed during training (Chapter 9) is attributable to the deliberately small size of the policy network described in Section 9.4. As discussed there, the network was kept compact by design, specifically so that it could eventually be quantized and deployed for on-device inference on the STM32F446 microcontroller already used for the embedded admittance controller – a platform whose limited flash, RAM, and

lack of any dedicated neural-network acceleration hardware make a larger policy impractical to run. Migrating to a more capable microcontroller, such as STMicroelectronics' STM32N6 family, would remove this constraint. The STM32N6 pairs an 800 MHz Arm Cortex-M55 core with a dedicated Neural-ART neural processing unit running at 1 GHz and delivering up to 600 GOPS of inference throughput together with up to 4.2 MB of on-chip RAM, compared to the 128 kB available on the STM32F446 [34]. Adopting this platform would allow a substantially larger and more expressive policy network to be trained and deployed on-device, potentially easing the current training difficulty while still preserving the long-term goal.

Once these steps are complete, the natural continuation is to close the loop between simulation and hardware: calibrating the virtual model's tactile and ToF behavior against real sensor measurements, and formally benchmarking the admittance controller – across repeated trials and a defined set of objects – to obtain the quantitative baseline against which a transferred learned policy can eventually be compared.

Bibliography

[1] "A Method of Human-Like Compliant Assembly Based on Variable Admittance Control for Space Maintenance." In: *Cyborg and Bionic Systems* (2024). DOI: 10.34133/cbsystems.0046. URL: https://spj.science.org/doi/10.34133/cbsystems.0046.

[2] "A New SMART Gripper with Soft Fingers and Integrated Force Sensors for Adaptive Robotic Tasks." In: *Mechatronics* (2025). URL: https://www.sciencedirect.com/science/article/pii/S092188902500315X.

[3] Fares J. Abu-Dakka and Matteo Saveriano. "Variable Impedance Control and Learning—A Review." In: *Frontiers in Robotics and AI* Volume 7 - 2020 (2020). ISSN: 2296-9144. DOI: 10.3389/frobt.2020.590681. URL: https://www.frontiersin.org/journals/robotics-and-ai/articles/10.3389/frobt.2020.590681.

[4] Hayder F. N. Al-Shuka et al. "Active Impedance Control of Bioinspired Motion Robotic Manipulators: An Overview." In: *Applied Bionics and Biomechanics* 2018 (2018), p. 8203054. DOI: 10.1155/2018/8203054.

[5] Elie Aljalbout et al. "The Reality Gap in Robotics: Challenges, Solutions, and Best Practices." In: *Annual Review of Control, Robotics, and Autonomous Systems* 9 (2026). arXiv:2510.20808, pp. 403–432. URL: https://arxiv.org/abs/2510.20808.

[6] *ams OSRAM – TMF8829 48x32 Multi-Zone Direct Time-of-Flight Sensor*. Future Electronics. URL: https://www.futureelectronics.com/npi/ams-osram-tmf8829-48x32-multizone-time-of-flight-sensor.

[7] "Compliant Grasp Control Method for the Underactuated Prosthetic Hand Based on the Estimation of Grasping Force and Muscle Stiffness with sEMG." In: *IEEE Access / PMC* (2024). URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC11592086/.

[8] Gabriel Dulac-Arnold, Daniel Mankowitz, and Todd Hester. *Challenges of Real-World Reinforcement Learning*. 2019. arXiv: 1904.12901 [cs.LG]. URL: https://arxiv.org/abs/1904.12901.

[9] Stefan Escaida Navarro et al. “Proximity Perception in Human-Centered Robotics: A Survey on Sensing Systems and Applications.” In: *IEEE Transactions on Robotics* 38.3 (2022), pp. 1599–1620. DOI: 10.1109/TRO.2021.3111786.

[10] Wen Fan, Haoran Li, and Dandan Zhang. *MagicGripper: A Multimodal Sensor-Integrated Gripper for Contact-Rich Robotic Manipulation*. 2025. DOI: 10.48550/arXiv.2505.24382. arXiv: 2505.24382 [cs.RO]. URL: https://arxiv.org/abs/2505.24382.

[11] *FlexiTac: A Low-Cost, Open-Source, Scalable Tactile Sensing Solution for Robotic Systems*. 2026. arXiv: 2604.28156 [cs.RO]. URL: https://arxiv.org/abs/2604.28156.

[12] “Force Admittance Control of an Underactuated Gripper with Full-State Feedback.” In: *2025 IEEE International Conference on Robotics and Automation (ICRA)*. 2025. URL: https://rai-inst.com/wp-content/uploads/2025/05/ICRA_Underactuated_Gripper.pdf.

[13] Neville Hogan. “Impedance Control: An Approach to Manipulation, Parts I–III.” In: *ASME Journal of Dynamic Systems, Measurement, and Control* 107 (1985), pp. 1–24.

[14] Julian Ibarz et al. “How to Train Your Robot with Deep Reinforcement Learning: Lessons We Have Learned.” In: *The International Journal of Robotics Research* 40.4-5 (2021), pp. 698–721. DOI: 10.1177/0278364920987859.

[15] *Impedance Control vs Admittance Control: Key Differences*. Patsnap Eureka. 2025. URL: https://eureka.patsnap.com/article/impedance-control-vs-admittance-control-key-differences.

[16] *Impedance Control vs. Admittance Control: Safety & Performance Tradeoffs*. Patsnap Eureka. 2025. URL: https://eureka.patsnap.com/article/impedance-control-vs-admittance-control-safety-performance-tradeoffs.

[17] *Introducing ST Neural-ART Accelerator*. STMicroelectronics. URL: https://www.st.com/resource/en/product_presentation/st-neural-art-accelerator-introduction.pdf.

[18] *Isaac Lab: A GPU-Accelerated Simulation Framework for Multi-Modal Robot Learning*. NVIDIA High Fidelity Simulation Research. 2025. URL: https://research.nvidia.com/labs/prl/publication/isaaclab2025/.

[19] Dmitry Kalashnikov et al. "Scalable Deep Reinforcement Learning for Vision-Based Robotic Manipulation." In: *Proceedings of the 2nd Conference on Robot Learning (CoRL)*. Vol. 87. Proceedings of Machine Learning Research. 2018. URL: https://proceedings.mlr.press/v87/kalashnikov18a/kalashnikov18a.pdf.

[20] Rickmer Krohn et al. *Multi-Resolution Tactile Imitation Learning for Contact-Rich Robotic Manipulation*. 2026. DOI: 10.48550/arXiv.2606.06281. arXiv: 2606.06281 [cs.RO]. URL: https://arxiv.org/abs/2606.06281.

[21] Luca Lach et al. "Bio-Inspired Grasping Controller for Sensorized 2-DoF Grippers." In: *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2022. URL: https://academia.skadge.org/publis/lach2022bioinspired.pdf.

[22] *Leveraging Contact Forces for Learning to Grasp*. 2018. arXiv: 1809.07004 [cs.RO]. URL: https://arxiv.org/abs/1809.07004.

[23] Viktor Makoviychuk et al. *Isaac Gym: High Performance GPU-Based Physics Simulation For Robot Learning*. 2021. arXiv: 2108.10470 [cs.RO]. URL: https://arxiv.org/abs/2108.10470.

[24] MarkTechPost. *Meta AI Introduces ReSkin: A Touch-Sensing Skin for AI Tactile Perception Research*. 2021. URL: https://www.marktechpost.com/2021/11/07/meta-ai-introduces-reskin-a-touch-sensing-skin-for-ai-tactile-perception-research-along-with-a-python-sensor-library-to-interface-with-reskin-sensors/.

[25] *Mass-Adaptive Admittance Control for Robotic Manipulators*. 2025. arXiv: 2504.16224 [cs.RO]. URL: https://arxiv.org/abs/2504.16224.

[26] Meta AI and Carnegie Mellon University. *ReSkin: Versatile, Replaceable, Lasting Tactile Skins*. 2021. URL: https://www.cmu.edu/news/stories/archives/2021/november/reskin.html.

[27] *ORCA: An Open-Source, Reliable, Cost-Effective, Anthropomorphic Robotic Hand for Uninterrupted Dexterous Task Learning*. 2025. DOI: 10.48550/arXiv.2504.04259. arXiv: 2504.04259 [cs.RO]. URL: https://arxiv.org/abs/2504.04259.

[28] Radhen Patel, Juan Canardo Alastuey, and Nikolaus Correll. "Improving Grasp Performance Using In-Hand Proximity and Dynamic Tactile Sensing." In: *2016 International Symposium on Experimental Robotics (ISER)*. Vol. 1. Springer Proceedings in Advanced Robotics. 2017, pp. 185–194. DOI: 10.1007/978-3-319-50115-4_17.

[29] "Pretouch Sensing for Robotic Grasping." MA thesis. University of Washington. URL: https://digital.lib.washington.edu/researchworks/items/7a730b47-26de-46fb-ad58-562753e6c174.

[30] Ran Qin et al. "RGB-D Grasp Detection via Depth Guided Learning with Cross-modal Attention." In: (2023). arXiv: 2302.14264 [cs.RO]. URL: https://arxiv.org/abs/2302.14264.

[31] Robotiq. *Physical AI*. 2026. URL: https://robotiq.com/physical-ai.

[32] Robotiq. *Tactile Sensor Fingertips TSF-85*. 2026. URL: https://robotiq.com/tactile-sensor-fingertips.

[33] Zilin Si and Wenzhen Yuan. *Taxim: An Example-Based Simulation Model for GelSight Tactile Sensors*. IEEE Robotics and Automation Letters. 2022. URL: https://publications.ri.cmu.edu/taxim-an-example-based-simulation-model-for-gelsight-tactile-sensors-and-its-sim-to-real-applications.

[34] *STM32N6 Series – High-Performance MCU with Neural-ART Accelerator NPU*. STMicroelectronics. URL: https://www.st.com/en/microcontrollers-microprocessors/stm32n6-series.html.

[35] S. Sulaiman et al. "Force Sensing Resistor (FSR): A Brief Overview and the Low-Cost Sensor for Active Compliance Control." In: *Proc. SPIE* 10011 (2016). URL: https://ui.adsabs.harvard.edu/abs/2016SPIE10011E..12S/abstract.

[36] Markku Suomalainen, Yiannis Karayiannidis, and Ville Kyrki. “A Survey of Robot Manipulation in Contact.” In: *Robotics and Autonomous Systems* 156 (2022), p. 104224. DOI: 10.1016/j.robot.2022.104224.

[37] SynTouch LLC. *BioTac – Multimodal Tactile Sensor*. URL: https://www.syntouchllc.com/Products/BioTac/.

[38] *TaCauchy: An Extensible FEM Framework for Vision-Based Tactile Simulation*. 2026. arXiv: 2606.20426 [cs.RO]. URL: https://arxiv.org/abs/2606.20426.

[39] “TacFR-Gripper: A Reconfigurable Fin-Ray-Based Gripper with Tactile Skin for In-Hand Manipulation.” In: *Actuators* 13.12 (2024), p. 521. DOI: 10.3390/act13120521.

[40] *TacRefineNet: Tactile-Only Grasp Refinement Between Arbitrary In-Hand Object Poses*. 2025. arXiv: 2509.25746 [cs.RO]. URL: https://arxiv.org/abs/2509.25746.

[41] *TACTO: A Fast, Flexible, and Open-Source Simulator for High-Resolution Vision-Based Tactile Sensors*. IEEE Robotics and Automation Letters. 2022. URL: https://www.researchgate.net/publication/358255422_TACTO_A_Fast_Flexible_and_Open-source_Simulator_for_High-resolution_Vision-based_Tactile_Sensors.

[42] *TMF8829 – 48x32 Multi-Zone Time-of-Flight Sensor*. ams-OSRAM. URL: https://ams-osram.com/products/sensor-solutions/direct-time-of-flight-sensors-dtof/ams-tmf8829-48x32-multi-zone-time-of-flight-sensor.

[43] Josh Tobin et al. *Domain Randomization for Transferring Deep Neural Networks from Simulation to the Real World*. 2017. arXiv: 1703.06907 [cs.RO]. URL: https://arxiv.org/abs/1703.06907.

[44] *UM3038 – A Guide to Using the VL53L7CX Time-of-Flight Multizone Ranging Sensor with 90° FoV*. STMicroelectronics. URL: https://www.st.com/resource/en/user_manual/um3038-a-guide-to-using-the-vl53l7cx-timeofflight-multizone-ranging-sensor-with-90-fov-stmicroelectronics.pdf.

[45] USC Viterbi School of Engineering. *Multimodal Tactile Sensor*. URL: https://viterbi.usc.edu/pdfs/gloeb/90894.pdf.

[46] *VL53L7CX Time-of-Flight Ranging Sensor – Datasheet (DS13865)*. STMicroelectronics. URL: https://www.st.com/resource/en/datasheet/vl53l7cx.pdf.

[47] Lilian Weng. *Domain Randomization for Sim2Real Transfer*. Lil'Log. 2019. URL: https://lilianweng.github.io/posts/2019-05-05-domain-randomization/.

[48] Achu Wilson et al. *Design of a Fully Actuated Robotic Hand With Multiple GelSight Tactile Sensors*. 2020. DOI: 10.48550/arXiv.2002.02474. arXiv: 2002.02474 [cs.RO]. URL: https://arxiv.org/abs/2002.02474.

[49] Yinghao Wu et al. *TranTac: Leveraging Transient Tactile Signals for Contact-Rich Robotic Manipulation*. 2025. DOI: 10.48550/arXiv.2509.16550. arXiv: 2509.16550 [cs.RO]. URL: https://arxiv.org/abs/2509.16550.

[50] Zhen Xie, Xinquan Liang, and Canale Roberto. "Learning-based robotic grasping: A review." In: *Frontiers in Robotics and AI* Volume 10 - 2023 (2023). ISSN: 2296-9144. DOI: 10.3389/frobt.2023.1038658. URL: https://www.frontiersin.org/journals/robotics-and-ai/articles/10.3389/frobt.2023.1038658.

[51] Wenzhen Yuan, Siyuan Dong, and Edward H. Adelson. "GelSight: High-Resolution Robot Tactile Sensors for Estimating Geometry and Force." In: *Sensors* 17.12 (2017), p. 2762. URL: https://dspace.mit.edu/entities/publication/91f85d41-73e2-4433-af3e-c9a4f5bc982c.

[52] Brayan S. Zapata-Impata, Pablo Gil, and Fernando Torres. "Tactile-Driven Grasp Stability and Slip Prediction." In: *Robotics* 8.4 (2019), p. 85. DOI: 10.3390/robotics8040085. URL: https://www.mdpi.com/2218-6581/8/4/85.

[53] Yuhao Zhou et al. *In-Hand Singulation and Scooping Manipulation with a 5 DOF Tactile Gripper*. 2024. DOI: 10.48550/arXiv.2408.00610. arXiv: 2408.00610 [cs.RO]. URL: https://arxiv.org/abs/2408.00610.