Design and Development of *IRIS*: An *Intelligent Robotic Imaging System* - A Low-Cost, 3D-Printed Cinema Robot Arm for Dynamic Shot Automation

by

Jerry (Qilong) Cheng

M.Eng., University of Toronto (2025)
B.A.S., University of Toronto (2023)

Submitted to the Electrical and Computer Engineering Department,
Applied Science and Engineering,
in partial fulfillment of the requirements for the degree of

Master of Engineering in Robotics

at the

THE UNIVERSITY OF TORONTO

September 2025

Author_____

Electrical and Computer Engineering Department
August 27th, 2025

Certified by_____

Ali Bereyhi
Professor at the Electrical and Computer Engineering Department
Applied Science and Engineering
Thesis Supervisor

Certified by_____

Matthew Mackay
Professor at the Mechanical and Industrial Engineering Department
Applied Science and Engineering
Thesis Supervisor

Design and Development of *IRIS*: An *Intelligent Robotic Imaging System* - A Low-Cost, 3D-Printed Cinema Robot Arm for Dynamic Shot Automation

by

Jerry (Qilong) Cheng

Submitted to the Electrical and Computer Engineering Department,
Applied Science and Engineering,
on August 27th, 2025, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Robotics

Abstract

This work presents the design and development of a low-cost, fully 3D-printed, six-degree-of-freedom (6-DOF) cinema robot arm, *IRIS* engineered to deliver professional-grade camera motion to independent filmmakers and small studios. *IRIS* is conceived as a vertically integrated platform, combining custom mechanical design, modular actuation, and advanced control strategies to achieve high precision, smooth dynamics, and intuitive operation at a fraction of the cost of commercial solutions.

Mechanically, *IRIS* employs a lightweight, modular architecture with strategically relocated actuators, timing-belt transmissions, and a differential wrist joint to minimize distal inertia while preserving dexterity. The chosen Unitree BLDC actuators, driven by field-oriented control (FOC), provide precise torque, velocity, and position regulation, with an impedance control layer enabling compliant, repeatable motion.

On the software side, an imitation learning (IL) framework enables autonomous path planning and dynamic obstacle avoidance based on expert demonstrations, supported by vision-based feedback for real-time trajectory correction. The control stack is implemented in ROS with a custom kinematic/dynamic model, allowing seamless integration of simulation, algorithm development, and hardware deployment.

The prototype, costing under $1,000 in materials, achieves a 1.5 kg payload capacity, ± 5 mm repeatability, and competitive motion speeds compared to high-end cinema robots. Validation includes actuator-level tracking, repeatability trials, and end-to-end IL-driven motion tests in real-world scenarios.

This project demonstrates that low-cost, additive-manufactured robotics when combined with modern control and learning-based planning can democratize access to precision cinematic motion control, expanding creative possibilities for a broader range of storytellers.

Thesis Supervisor: Ali Bereyhi
Title: Professor at the Electrical and Computer Engineering Department, Applied Science and Engineering

Thesis Supervisor: Matthew Mackay
Title: Professor at the Mechanical and Industrial Engineering Department, Applied Science and Engineering

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 What is a cinema robot?

Cinema robots are widely used in the film industry, advertising and in studios. Most cinema robots use the exact same technology as industrial robot arms, with the only difference being that the end effector has a cinema camera attached instead (1). The reason cinema robots have gained so much popularity in recent years is due to higher content consumption by the masses and demand from various industries, such as consumer products, food, artistic studios, and more (2).

(a) Cinema robot example 1 (b) Cinema robot example 2 (c) Cinema robot example 3

Figure 1-1: Examples of cinema robots used in the industry.

Cinema robots, while widely used in professional film production for decades, remain unfamiliar to most consumers. These systems share many components with industrial robotic arms used in manufacturing and research, yet they are uniquely tailored for cinematographic purposes. A popular video from MOCO company *Bolt* clearly explains what sets a cinema motion control robot apart from traditional robots (3).

1.2 What can cinema robot do

To summarize the industry's expectations for a professional cinema robot arm, several **core functional requirements** must be met:

Tasks 1.

- ***Programmable Motion Control****: The robot must support highly customizable, keyframe-based motion programming at both the joint and Cartesian levels at the end-effector. This allows the user to precisely choreograph complex camera movements.*
- ***High Repeatability and Accuracy****: Cinema robots must consistently return to sub-millimeter (often pixel-level) precision. This is critical when replicating shots for visual effects (VFX), product commercials, or composite layers.*
- ***High-Speed Capability****: Many shots involve ultra-slow-motion cinematography using high-speed cameras (e.g., 1000 fps). The robot must move quickly and smoothly to create dynamic movement within a few milliseconds of shutter time.*

Figure 1-2: Comparison of control interfaces: Flair software (left) used in cinema robotics for visual keyframing, versus legacy pendant-based interfaces (right) from industrial robotics.

Beyond these primary functions, several **secondary features** enhance usability and broaden creative potential:

Tasks 2.

- ***Intuitive User Interface****: Unlike industrial arms programmed with code or outdated pendant-style controllers (as shown in Figure 1-3), cinema robots often leverage visual, timeline-based GUIs like Flair or MRMC's Academy Award-winning interface, which allow filmmakers to key frame motion intuitively.*
- ***Large Working Space****: To accommodate wide sweeping camera moves, dolly-style tracking, or dramatic arcs, the robot must offer a long reach or be mounted on a motorized track.*
- ***High Payload Capacity****: The end-effector (camera mount) must support large cinema cameras, lenses, wireless transmitters, follow-focus systems, and even additional lighting.*

- ***Safety and Compliance**: Since film crews work in close proximity to these systems, built-in safety measures like emergency stops, soft torque limits, and collision detection are essential.*

Figure 1-3: On the left shown the complex control software for key-framing the cinema robot arm, while on the right shown the outdated pendant-style controller that has been used in the industry for over two decades

1.3 What motivates the project?

Despite the wide usibility and interest in cinema robot arms, there are barriers blocking a normal consumer from accessing cinema robot arms.

The biggest barrier to consumer access to cinema robots is **cost**. On average, a cinema robot, excluding the camera or linear rail system, costs around $100,000 to $250,000, depending on the size and features (4; 5; 6). For instance, the Bolt High-Speed Cinebot costs approximately $200,000 to $250,000, excluding the camera (4). Similarly, the Milo Motion Control Rig, known for its advanced capabilities, is also priced in the same range (5). Another option, the Kira Cinema Robot, falls within the same price bracket, further contributing to the cost barrier for average consumers (6).

Another issue is the **operational barrier** - normally, cinema robots require professional operators trained to use dedicated control software (7). Most controllers for robot arms are still operated, unfortunately, via a teach pendant, the controller pad used for industrial robot arms (8). The lack of intuitiveness is another reason why cinema robots have not become more widespread. Others that use some professional control software, such as Lensmaster and MP Studio, have several notable drawbacks. First, the complexity and the steep learning curve of these systems require specialized training, which can slow down production and limit usability for less experienced operators (9; 10). Moreover, these advanced software solutions are typically bundled with high-end robots, making them costly and inaccessible to smaller studios or independent filmmakers (11). Finally, despite some advancements in user interfaces, such as iPad controls in MP Studio, many users still find these systems lacking in intuitiveness compared to simpler technologies (12).

This project aims to address those two biggest issues.

1.4 Design objectives

Cinema robots are still a relatively new technology for general consumers, the main objective of this project is to make this technology more popularized to the masses by bring down the barrier of entry.

First and foremost is to bring down the cost. Compared to most general consumer electronics, like a DJI consumer graded professional drone to be around 2500 USD, a consumer professional camera like the Sony a7r5 to be around 3000 USD, and consumer graded 3D printers from Bambulab to be around 1500 USD. Driven to be a consumer-level robot arm that everyone can build at home, we set the cost objective to be less than 1000 USD.

Objective 3.

Cost to be lower than 1000 dollars.

The second is to maintain the functions that an industrial cinema robot can achieve as mentioned in section 1.2: being programmable, having high accuracy and high speed. This is necessary as it the goal is to make the product useful for the cinema purpose on budget without sacrificing the functionalities too much.

Objective 4.

Programmable, high accuracy, high speed

Finally, to solve the final barrier of entry being non-intuitive mentioned in section 1.3, the control of the robot arm should be learning-free, meaning the user do not need to learn another programming language, software or complex controller interface. Like the DJI drone, all the low-level balancing controls should be in black-box, without the user needing to interact. The project aims to make the path-planning of the robot arm to be autonomous without needing the user to program themselves. In addition, instead of having a separate hardware controller, like a pendant, or running professional graded custom software on high performance computers, the user should be able to control the cinema robot arm easily by either only having the device itself, or a mobile application without the additional hassels.

Objective 5.

Intuitive user interface and autonomous planning.

In the later sections, the report will be going into details how each objective is being realized, and evaluate their effectiveness.

1.5 Scope of the project

Given the objectives outlined in section 1.4, this project aims to design and develop as many components from scratch as possible to make the cinema robot arm more accessible to hobbyists interested in creating professional-

quality motion-controlled shots. Vertical integration, developing all components in-house, offers two key advantages: it minimizes overall cost and provides full control over both hardware and software. For example, most structural components are designed for 3D printing, significantly reducing manufacturing costs and making the system easier for individual makers to replicate and assemble. This end-to-end integration also enables the development of custom control software, allowing complete control over motion algorithms, communication protocols, and user interface design. To fulfill the third objective, intuitive control, imitation learning is employed. This approach allows the robot arm to learn camera motions from expert demonstrations, mimicking the behavior of professional operators through supervised learning techniques. The resulting system can generate smooth and purposeful camera trajectories autonomously. However, designing all components in-house also presents challenges. Time constraints limit the amount of testing and iteration that can be performed. Furthermore, due to budget limitations and the absence of economies of scale, high-precision machined parts or advanced materials cannot be used. As a compromise, off-the-shelf actuators and motor controllers are selected to minimize development time and cost. These components are integrated into the system but remain outside the scope of custom design. In summary, the project is divided into three major components:

1. **Mechanical Design**: Structural design, 3D modeling, and mechanical assembly of the cinema robot arm.
2. **Firmware Development**: Low-level actuator control, kinematic/dynamic modeling, and ROS integration using an in-house package.
3. **High-Level Control Software**: Training of imitation learning models and development of an intuitive mobile application for teleoperation and user interaction.

The items that are not included in the design are the actuators, and the low-level communication and control for the motors. The project will use the SDK or libraries provided by the actuator companies.

Chapter 2

Related Work

State-of-the-art

Current Cinema Robot

The current landscape of robotic arms in the filming industry is characterized by high-end, sophisticated equipment designed primarily for professional use. These robotic arms, such as the Bolt High-Speed Camera Robot and PhotoRobot's Robotic Camera Arm, are state-of-the-art in terms of precision, speed, and versatility. However, they are also typically large, complex, and expensive, with prices often exceeding tens of thousands of dollars. This makes them inaccessible for smaller production studios, independent filmmakers, and photography enthusiasts who could benefit from such technology.
The only affordable robot arm E-Jib still costs over 9000 dollars, with difficult controls and configuration processes.

A brief summary of the state-of-the-art cinema robots are as follows:

Robot Arm	Pricing	Dimensions (HxWxD)	Weight	Max Reach	Speed	Repeatability
Bolt	$69,400 plus	Max Height: 3.5m	600kg (Bolt + Base)	2.0m	Up to 5m/s	High precision
PhotoRobot(V8)	$100,000 plus	0.8×1.2×1 mm	106 kg	900 - 2906 mm	Not specified	High precision
Loki	$22,450 plus	Not specified	Base: 45kg, Arm: 30kg	1.1 meters	Up to 4m/s	0.05mm
Colossus	$200,000 plus	1.47 × 0.81m	Base: 45kg, Arm: 30kg	1.1 meters	Up to 6.8m/s	0.05mm
E-JIB MINI	$9,000 plus	N/A	N/A	1.8 meters	Up to 1m/s	High

Table 2.1: Comparison of Latest Cinema Robot Arms

Current 3D Printed Robot Arms

There are a lot of attempts on creating a high-precision and reliable 3D printed robot arms too, notably Arctos and Dexter, both are able to provide high precision and relatively high payload. However, those 3D printed robot arms are not customized for cinema purposes, especially on the maximum reach, where normally higher number is more

(a) Bolts

(b) Colossus

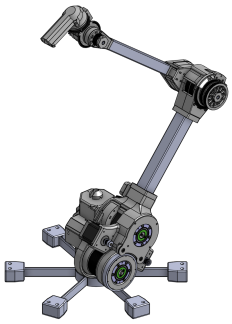
(c) Loki

Figure 2-1: Examples of state-of-the-art cinema robot arms: (a) Bolts, (b) Colossus, and (c) Loki.

desired. In addition, those arms generally uses stepper motors as the actuator, thus have limited speed and not ideal for cinema purposes.

Name	DOF	Payload	Reach	Accuracy	Open Source
Arctos	6	500g	600mm	High	Yes
Dexter	6	3kg	700mm	0.025 mm	Yes
BCN3D Moveo	6	500g	625mm	0.1mm	Yes
Mirobot	6	500g	400mm	0.2mm	Yes

Table 2.2: 3D printed robot sepcs comparison



(a) Dexter

(b) Arctos

(c) BCN3D Moveo

(d) Mirobot

Figure 2-2: Examples of consumer or open-source 3D printed robot arms: (a) Dexter, (b) Arctos, (c) BCN3D Moveo, and (d) Mirobot.

Chapter 3

Design Decisions

The design of *IRIS* follows the goals in Section 1.4: affordability, ease-of-use, and smooth, precise, repeatable motion for cinema work. Achieving these goals requires trade-offs across mechanics, actuators, transmissions, and structure. This section summarizes the key design decisions, why we made them, and the trade-offs involved. In brief, we explain how each choice contributes to *IRIS*'s capabilities (tracking accuracy, payload, workspace) and defines its performance envelope.

3.1 Actuator Selection

In robotic systems, four main types of electric actuators are commonly used: brushed DC motors, stepper motors, servo motors, and brushless DC (BLDC) motors, as illustrated in Figure 3-1. Each category differs in cost, torque density, control requirements, and precision, making actuator selection a trade-off between performance and budget.

For this project, the primary objectives are affordability, while achieving sufficient torque, high precision, and responsiveness for an articulated robot arm. These goals make off-the-shelf actuators an attractive choice, as they reduce development time and integration complexity. Within the available actuator types, BLDC motors with field-oriented control (FOC) offer the best balance of torque output, efficiency, precision, and smoothness, while enabling advanced control modes (torque/velocity/position) over standard interfaces and supporting backdrivability for safe HRI and learning from demonstration (see Table 3.1 for a summary). Recent cost reductions driven by robotics and e-mobility have brought FOC-BLDC pricing close to closed-loop steppers, yet with higher torque density and superior dynamic response. This combination of performance, control flexibility, and cost-effectiveness makes FOC BLDC motors the most suitable choice for this project.

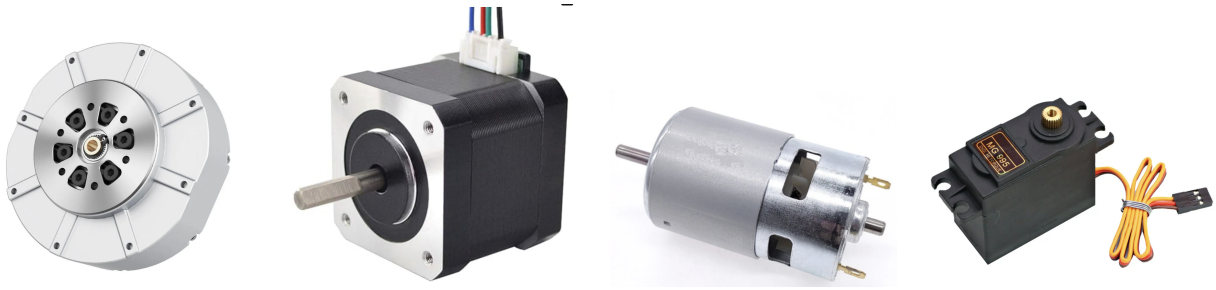


Figure 3-1: Comparison of four common actuator types used in robotics: FOC BLDC, stepper, brushed DC, and servo motors.

Table 3.1: Comparison of Common Motor Types for Robotic Applications

Motor Type	Pros	Cons
Brushed DC	Simple control, low cost	Low efficiency, wears out quickly, low precision
Stepper	High holding torque, precise open-loop control, inexpensive	Noisy operation, reduced torque at higher speeds, high vibration
Servo (RC)	Built-in position control, easy interfacing	Limited torque, non-industrial durability, restricted feedback capabilities
BLDC with FOC	High efficiency, smooth and quiet motion, precise closed-loop control, backdrivability	Higher cost, complex control electronics

3.2 Torque Sizing

IRIS uses two proximal actuator clusters, three at the shoulder and three at the elbow, with no wrist motors. Distal DOFs are driven by cable/belt transmissions, reducing moving mass.

Let L_1, L_2, L_3 be link lengths; m_{L1}, m_{L2}, m_{L3} their masses; m_a actuator mass; m_p payload. Worst-case horizontal static torques are:

$$\begin{aligned} T_{\text{sh}} &= g\big[m_{L1}\tfrac{L_1}{2} + n_e m_{a,e} L_1 + m_{L2}\big(L_1 + \tfrac{L_2}{2}\big) + m_{L3}\big(L_1 + L_2 + \tfrac{L_3}{2}\big) + m_p(L_1 + L_2 + L_3)\big] \\ &= 9.81\,[0.06 + 0.468 + 0.1575 + 0.2125 + 1.275] \\ &\approx 25.8\ \text{Nm}, \end{aligned} \tag{3.1}$$

$$\begin{aligned} T_{\text{el}} &= g\big[m_{L2}\tfrac{L_2}{2} + m_{L3}\big(L_2 + \tfrac{L_3}{2}\big) + m_p(L_2 + L_3)\big] \\ &= 9.81\,[0.0525 + 0.1625 + 0.825] \\ &\approx 10.6\ \text{Nm}, \end{aligned} \tag{3.2}$$

$$\begin{aligned} T_{\text{wr}} &= g\big[m_{L3}\tfrac{L_3}{2} + m_p L_3\big] \\ &= 9.81\,[0.03125 + 0.375] \\ &\approx 4.0\ \text{Nm}. \end{aligned} \tag{3.3}$$

Based on the required torque calculations, several off-the-shelf FOC motors fall within the overall budget constraint

of 1,000 USD for the build: the Damiao motor, the Xiaomi motor, and the Unitree motor, as shown in Figure 3-2. Their key specifications are summarized in Table 3.2.

Table 3.2: Comparison of Off-the-Shelf FOC Motors Within $1000 Budget

Motor	Cost [USD]	Peak Tq [Nm]	Rated Tq [Nm]	Weight [g]	Comm.	Key Features
Damiao DM-S3519-1EC	83.49	9	3	≈300	CAN	Gearbox, encoder, MIT/speed/pos modes
Xiaomi CyberGear	83.49	12	4	≈317	CAN	FOC, 28-pole, 7.75:1 gearbox
Unitree GO-M8010-6	69.69	23.7	15	≈530	RS-485	Built-in FOC, temp/pos sensors

Among these options, the Unitree motor offers a favorable balance of cost and performance, delivering substantially higher torque output at a competitive price. Consequently, it was selected as the primary actuator for *IRIS*. A more detailed discussion of the motor performance and the control paradigm is provided in chapter 6.

Figure 3-2: Comparison of three off-the-shelf FOC motors from Damiao, Xiaomi and Unitree

3.3 Alignment Design

In summary, robot-arm kinematics broadly fall into two categories, *aligned-axis* and *offset-axis*, as illustrated in Figure 3-3. In aligned (collinear) designs, joint axes are arranged to minimize offsets. These arms are often stiffer and better suited to high payloads but can suffer from self-collision and constrained motion in certain poses, which limits reachable orientations. In offset-axis designs, deliberate axis offsets create unobstructed sweeps (often allowing full 360° rotation about selected joints) and enlarge the usable workspace; however, mass asymmetry can reduce structural rigidity and payload margins compared to aligned counterparts.

IRIS prioritizes cinematographic reach and camera freedom, so we adopt the offset-axis layout to maximize unobstructed motion and exploit continuous rotation capabilities of the BLDC-driven joints, trading some stiffness for a larger, more practical workspace for creative shots.

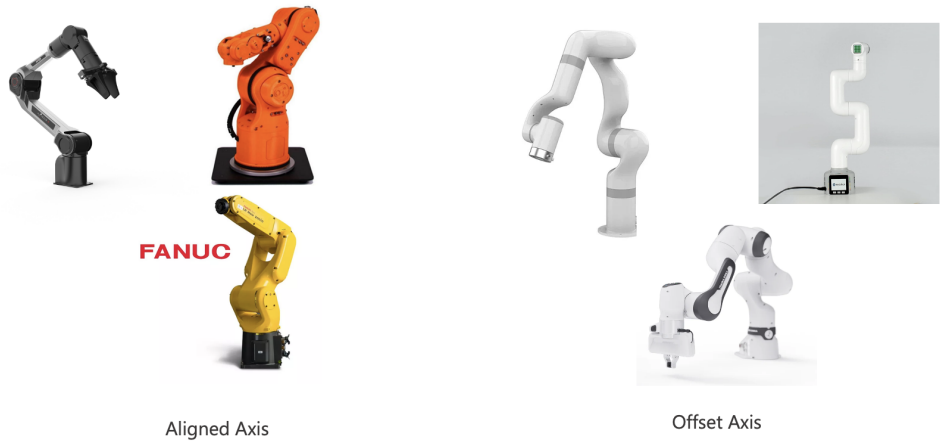


Figure 3-3: Two axis alignment designs

3.4 Transmission Design

We considered four options for the tranmission design: (i) *direct drive*: excellent precision and minimal parts, but poor torque density for our payload; (ii) *gears*: rigid and low backlash with quality gearing, but heavy, costly, and mechanically complex; (iii) *timing belts*: toothed, no slip, low backlash when properly tensioned, low cost; (iv) *cables*: light and low backlash, but sensitive to tension drift and long-term stretch.

Timing belts (GT/HTD profiles) are inexpensive and widely available with off-the-shelf pulleys and idlers; they provide positive engagement (no slip), quiet operation, and low friction. Critically, they allow *remote motor placement*

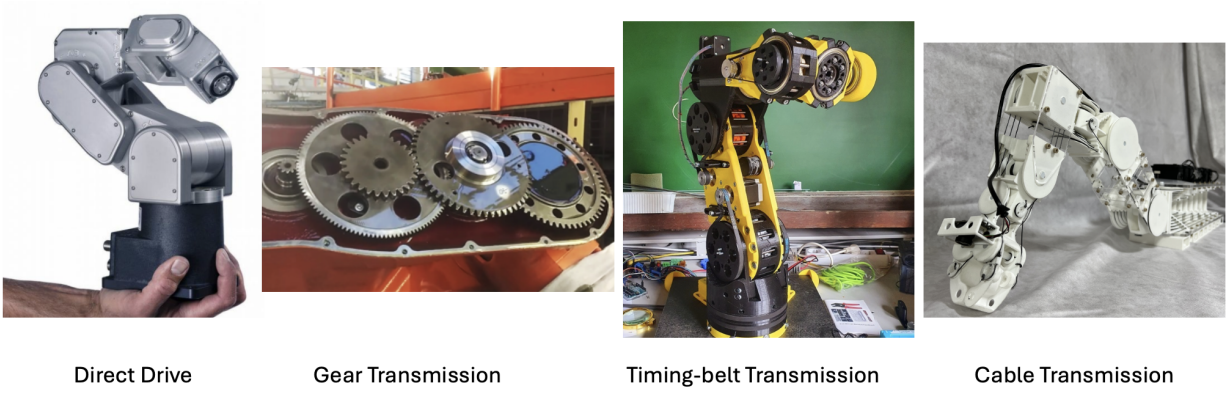
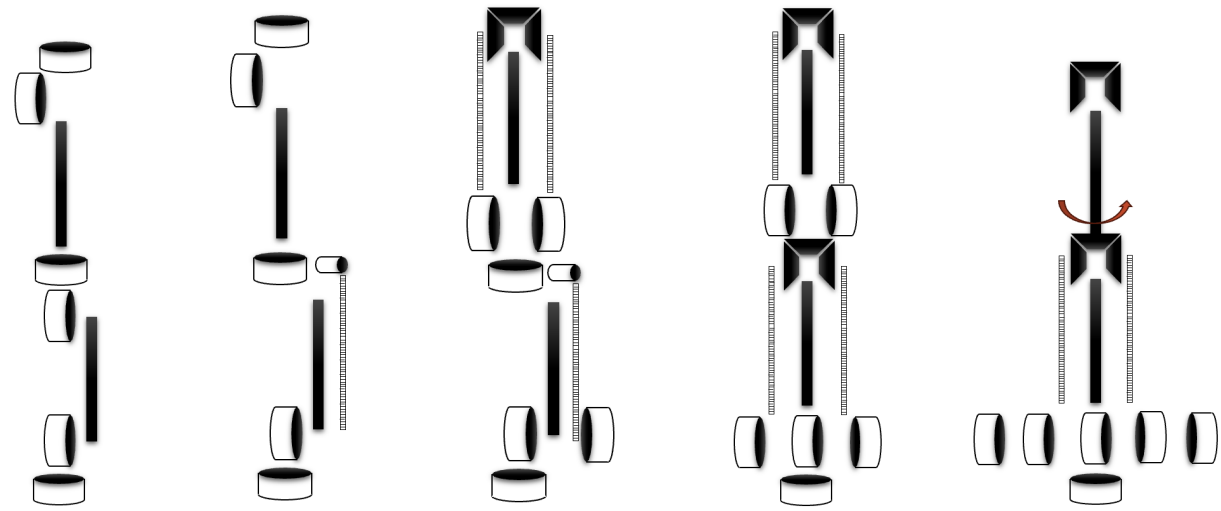


Figure 3-4: Types of mechanism used in robot arm transmission

3.5 Structural Design



(a) Baseline **(b)** 1-Offset **(c)** 3-Offset **(d)** 4-Offset **(e)** 5-Offset

Figure 3-5: Design-iteration snapshots used in the ablation study. All renders are scaled uniformly for direct visual comparison.

We number the revolute joints J_1- J_6 (base joint yaw to wrist joint roll) and collect them in $\boldsymbol{\theta} = [\theta_1, \ldots, \theta_6]^\top$. The design objective is to minimize *reflected inertia* at the flange so a 3 kg cinema payload can be manoeuvred without overloading the drives. This is pursued by progressively relocating actuator mass from distal links to the pedestal while preserving torsional stiffness. The baseline ($0-$offset) in Figure 3-5a is a direct-drive, six-DOF PUMA chain with one motor per joint, whose cumulative moving mass caps payload at 0.6 kg. In the $1-$offset variant (Figure 3-5b), an HTD-5M belt moves the elbow-pitch motor (J_3) to the upper arm, trimming fore-arm inertia by $\sim$40 % while avoiding belt twist through elbow roll. The 3–offset concept (Figure 3-5c) adds a *differential belt stage*: two co-axial motors drive concentric pulleys, where common-mode speed yields wrist pitch (J_5) and differential speed yields wrist roll (J_6), removing both wrist motors from the flange. Extending the same principle to shoulder and elbow pitch gives the $4-$offset layout (Figure 3-5d), which relocates J_2-J_3 drives to the torso, leaving only two motors distal and cutting fore-arm mass by $\approx$83 % at the cost of longer, more compliant belts. The final $5-$offset architecture (Figure 3-5e) migrates *all* actuators to the pedestal: three concentric carbon-fibre shafts traverse the elbow-inner for elbow roll (J_4), outer pair for the belt drives to J_5 and J_6. Although this yields the lowest distal inertia, it introduces maximal transmission complexity and compliance, motivating model-based disturbance rejection in the low-level torque loop.

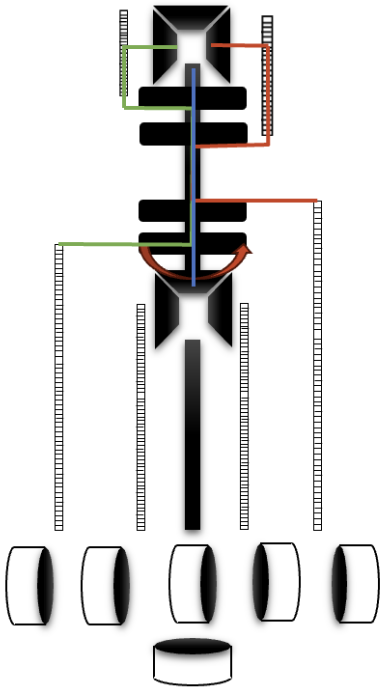


Figure 3-6: The power transmission of the first iteration of the design.

The first physical prototype implemented a partial version of the 5-offset concept, integrating both the remote elbow-pitch actuation and the differential wrist mechanism. A hollow shaft (blue in Fig. 3-6) routes the base yaw

torque (J_1) through the center of the structure. Two orthogonal GT belts drive the shoulder and elbow pitch joints: the green belt conveys torque from the base-mounted shoulder motor to J_2 after a 90° redirection; the orange belt bypasses the shoulder stage entirely and twists vertically to reach J_3.

At the elbow, a compact bevel-gear differential (copper in Fig. 3-6) splits the elbow-pitch torque (J_3) and introduces a second degree of freedom for forearm roll (J_4), leveraging compliance sharing between both axes. A second, smaller differential stack is embedded at the wrist, combining two coaxial belts to actuate wrist pitch and roll (J_5, J_6). This multi-layered design maintains kinematic decoupling and removes all actuators from the arm links, reducing overall moment of inertia and enabling faster, smoother motion suitable for cinematic shots.

Despite its advantages, this design presented significant challenges in precision alignment, cumulative backlash, and belt compliance, which must be addressed in future iterations through both mechanical refinement and advanced control strategies such as torque feedforward and impedance tuning.

Figure 3-7: 3D model of the first design iteration.

The second design iteration adopts an *in-link direct-drive* topology to eliminate mechanical friction and improve torque transparency. This shift was motivated by the observation that timing belts and long transmission paths in the first design introduced significant estimation uncertainty, particularly in high-speed maneuvers or when implementing model-based control.

In this configuration, each joint is actuated by a compact, self-contained module where the **stator** is rigidly mounted to the proximal link, and the **rotor** is directly coupled to the distal link without any intermediate gearing or belt systems. The absence of transmission backlash and friction allows the joint torque to be approximated di-

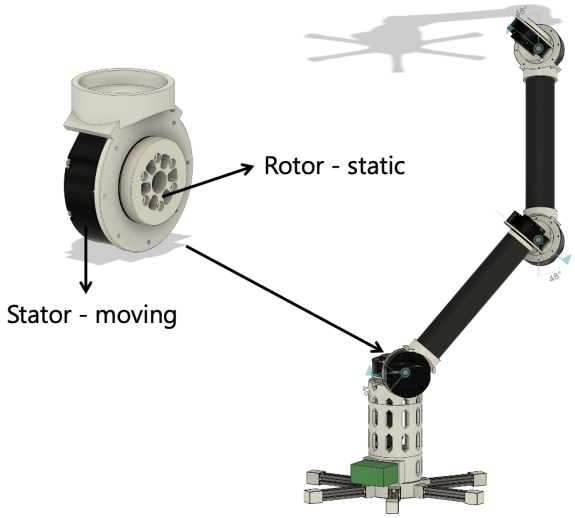


Figure 3-8: Second-iteration CAD: modular in-link direct-drive joints. Each stator is rigidly clamped to the proximal link, while the rotor hub is integrated directly into the distal link. No transmission is used between the motor and link motion.

rectly by the motor current, scaled by a known motor constant, enabling high-fidelity impedance control and back-EMF-based state estimation.

This architecture also simplifies the dynamic model of the robot, making it well-suited for learning-based and model-predictive control methods. Similar direct-drive strategies are used in research platforms such as MIT's Cheetah robot and Agility Robotics' Digit upper-limb joints, where torque control accuracy and backdrivability are critical design priorities (13; 14).

Moreover, by designing each joint as a repeatable modular actuator block, the robot becomes highly reconfigurable. Link lengths and joint modules can be swapped or extended with minimal reassembly, supporting rapid development and experimental prototyping.

The third and final design iteration represents a deliberate compromise between the extremes of mechanical simplicity and maximal weight reduction. Drawing inspiration from the earlier $3-$offset concept (Figure 3-5c), the final configuration integrates a **differential-driven wrist** to reduce the distal inertia while maintaining dexterous end-effector control.

Specifically, a pair of co-axial actuators mounted at the elbow drives the wrist-pitch (J_5) and wrist-roll (J_6) axes via a differential belt stage, significantly reducing the moment of inertia at the tool flange. Simultaneously, the elbow-pitch motor (J_3) is relocated to the shoulder link using an HTD-5M timing belt, which routes torque around the elbow-roll axis without belt twist. The remaining three actuators-responsible for base yaw (J_1), shoulder pitch (J_2), and elbow pitch (J_3)-are consolidated at the robot base, while the differential wrist and elbow-roll (J_4) motors are embedded in the distal chain.

This architecture strategically balances several competing objectives: reducing the weight of the distal limb, preserving mechanical simplicity, and minimizing the number of long, compliance-prone transmission paths. Compared to the 5-offset design, this configuration avoids the need for multi-layered concentric shafts or complex bevel gear assemblies, thereby improving manufacturability and reliability.

The final design supports a $1.5\,\mathrm{kg}$ professional camera payload at the end-effector, while maintaining sufficient backdrivability, kinematic decoupling, and reach for high-speed tracking shots and cinematic motion control. The assembled 3D CAD renders from multiple perspectives are shown in Figure 4-1. In total, there are only 23 3D printed parts to construct the entire robot arm. The detailed components breakdown can be seen in Figure 4-5

3.6 Wrist Joint Design

Several wrist joint designs were considered, including direct-drive/serial joint, differential joint and parallel joint. Serial wrists, which stack multiple motors to achieve roll-pitch-yaw but add weight and inertia at the end-effector; parallel wrists, which offer a more compact design but involve complex mechanics and control; and differential wrists, which couple joint motions to reduce actuator count and shift motor mass toward the base, making them

ideal for lightweight, high-speed applications such as precision camera movement in cinema robotics. To reduce

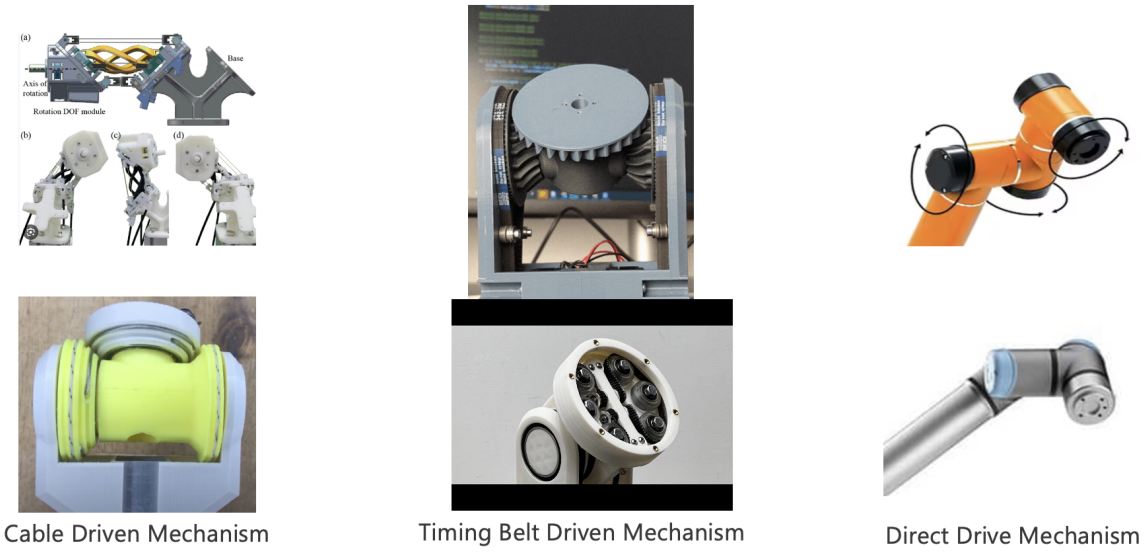


Figure 3-9: Different wrist joint designs

end-effector inertia while preserving full dexterity, the wrist joint was designed using a compact **differential belt mechanism**. This solution allows two coaxial actuators to simultaneously drive the wrist-pitch and wrist-roll axes by combining their motions through a mechanically decoupled differential stage. Such designs are commonly seen in high-performance robotic manipulators, including humanoid wrists and camera gimbals, where weight at the end effector must be minimized. Other transmission systems like using cable

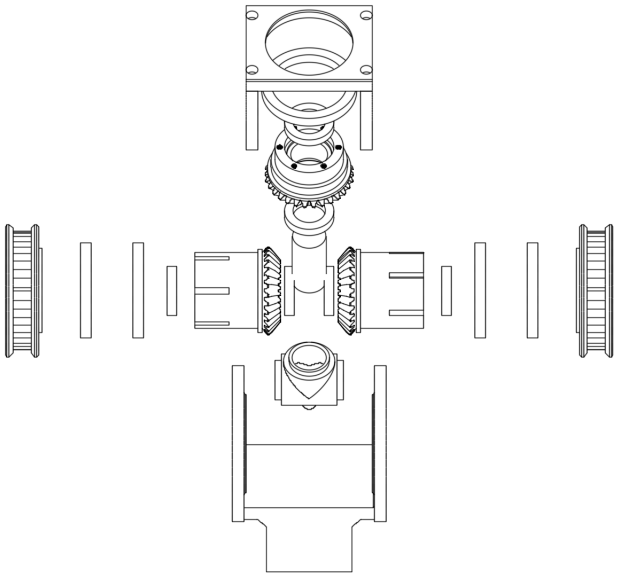


Figure 3-10: Exploded view of the differential wrist joint, showing the co-axial pulley stack, output links, and bearing-supported housings. This architecture supports combined wrist-pitch and roll with minimal backlash and inertia.

The final wrist design was selected for its compactness, symmetry, and compatibility with cable routing through the forearm. An exploded view of the joint is shown in Figure 3-10, illustrating the internal pulley layout and bearing support structure.

3.7 Design Iterations

Despite the major design decisions having been finalized, the arm underwent numerous redesigns to improve rigidity, reduce complexity, and achieve a more polished overall build. The initial prototype is shown in Figure 3-11. The following section discusses key aspects of these design iterations and the improvements made at each stage.

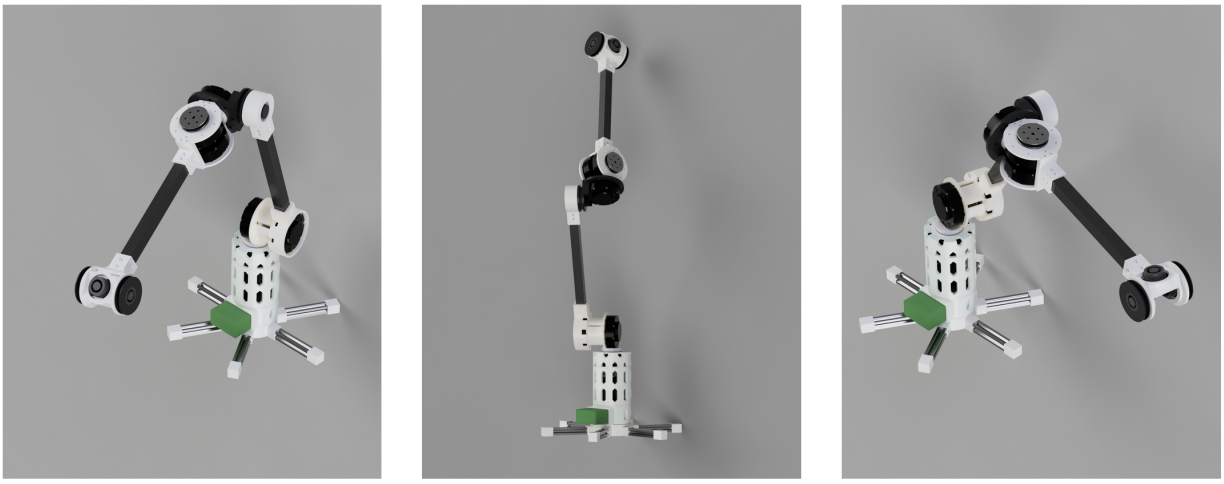


Figure 3-11: Design of the initial prototype

3.7.1 Reduced Length

In the original design, the elbow and forearm are $\ell_e = \ell_f = 500\,\text{mm}$, giving a shoulder-to-tool span $L_{\text{span}} = \ell_e + \ell_f \approx 1.00\,\text{m}$. The static shoulder torque is

$$\tau = mgL_{\text{eff}}.$$

Worst case (all distal mass at the tool) gives a torque-per-kg of $gL_{\text{span}} = 9.81 \times 1.00 = 9.81\ \text{Nm/kg}$, so for $m = 3.0\,\text{kg}$, $\tau \approx 29.4\ \text{Nm}$, far above the $\sim 10\ \text{Nm}$ continuous limit. A more realistic distributed-mass estimate places the COM at $\ell_e + \frac{1}{2}\ell_f = 0.75\,\text{m}$, yielding $9.81 \times 0.75 = 7.36\ \text{Nm/kg}$ and $\tau(m{=}3\,\text{kg}) \approx 22.1\ \text{Nm}$, still beyond the continuous rating (and near the 23.7 Nm peak), causing overheating.

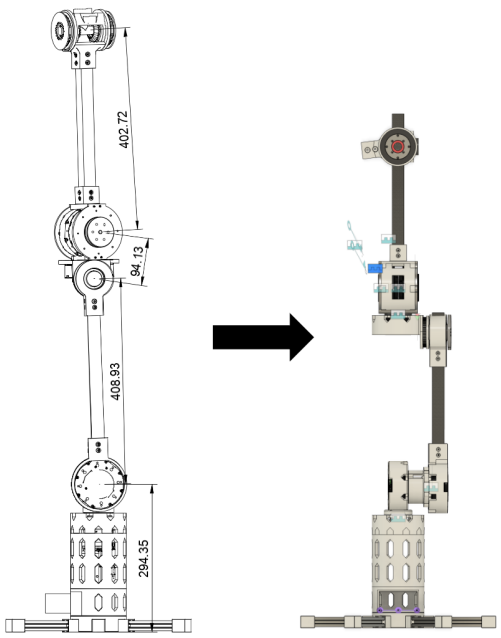


Figure 3-12: Dimension table of the shortened design

Under the 10 Nm continuous limit, the allowable distal mass is

$$m_{\max} \approx \frac{10}{9.81} \approx 1.02\ \text{kg} \quad (\text{worst case, } 1.00\ \text{m}), \qquad m_{\max} \approx \frac{10}{7.36} \approx 1.36\ \text{kg} \quad (\text{distributed, } 0.75\ \text{m}),$$

leaving minimal payload margin once link and actuator masses are included. This motivates shortening the links to reduce L_{eff} proportionally, bringing operation within the continuous torque envelope and recovering usable payload.

3.7.2 Timing Belt Flipped Direction

The initial design experienced self-collision issues when certain joints rotated, which restricted the range of motion. This was most apparent when the elbow joint rotated toward the base, causing the end-effector to interfere with the timing belt that drives the elbow. Such collisions not only limited joint travel but also posed a risk of mechanical wear and reduced operational reliability. To address this limitation, the timing belt was repositioned to the opposite side of the joint. This modification provided an additional 15 degrees of elbow motion, expanding the reachable workspace and improving the arm's versatility. Furthermore, relocating the belt shifted the lower arm's center of gravity closer to its rotational axis, which reduces torque demands on the joint actuator and contributes to smoother, more energy-efficient movement.

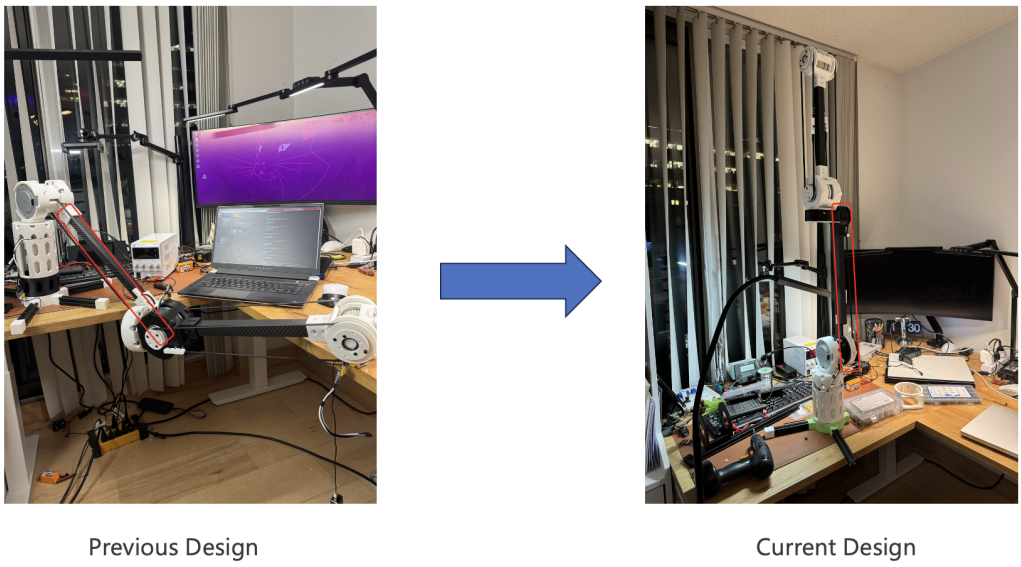


Figure 3-13: Illustration shows that flipping the direction of the timing belt to avoid self-collision

3.7.3 Change of Timing Belt

Originally, the design employed the widely available GT2 timing belt due to its accessibility and low cost. However, GT2 belts proved problematic for this application: they lack sufficient strength to prevent slippage under high payloads, require precise tensioning to achieve their rated capacity. In addition, it is tend to stretch over time, reducing their service life and necessitating frequent maintenance. To overcome these limitations, the design now uses an HTD-5M timing belt, which offers a higher torque capacity to prevent slippage under load, greater resistance to stretching for improved durability, and more reliable performance during sustained high-load operation, thereby extending the lifespan and enhancing the overall robustness of the cinema robot arm.

(a) GT2 timing belt

(b) HTD-5M timing belt

Figure 3-14: Comparison of the two timing belts

3.7.4 Carbon Fiber Rod Linkage

To mitigate bending and reduce weight, we use carbon-fiber (CF) tubes for the linkages connecting actuator sub-assemblies. We also evaluated aluminum (Al) options (20 mm × 20 mm extrusions and smooth round rods). Al is inexpensive, readily available, and offers off-the-shelf brackets, but slender sections are prone to bending and buckling; increasing cross-section improves stiffness at the cost of mass, which directly reduces end-effector payload. Practically, CF tubes cut link mass by ∼30-60% versus comparable Al while maintaining or increasing bending stiffness; they also damp vibrations better, improving motion quality.

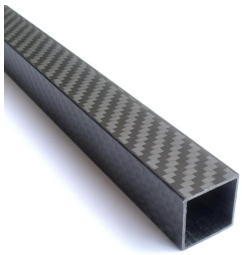


Figure 3-15: 20*20mm carbon fiber tube used in *IRIS*.

3.7.5 Integration of Many Parts

Initially, the major sections of the cinema arm were composed of numerous small parts, which significantly increased assembly complexity. A high part count not only prolongs the initial build but also complicates maintenance-replacing a motor or renewing a damaged component often required extensive disassembly. In some cases, the assembly sequence was interdependent; for example, accessing a damaged base component required removing the entire elbow section. Such sequential dependencies slowed down repairs and made rapid design iterations difficult, shifting valuable time away from testing and improvement toward lengthy assembly work.

To address these issues, multiple sub-components were consolidated into single, integrated parts wherever possible. By redesigning critical sections as monolithic single-print 3D-printed pieces, the number of fasteners and mating surfaces was reduced, eliminating the need for assembly in those areas entirely. For example, structural housings that previously consisted of multiple plates and brackets were combined into one unified shell, and actuator mounts were integrated directly into link bodies, as seen in Figure 3-16 This approach not only reduced assembly time and part misalignment risk but also enabled direct replacement of major modules without disturbing adjacent sections, greatly accelerating both maintenance and iterative prototyping.

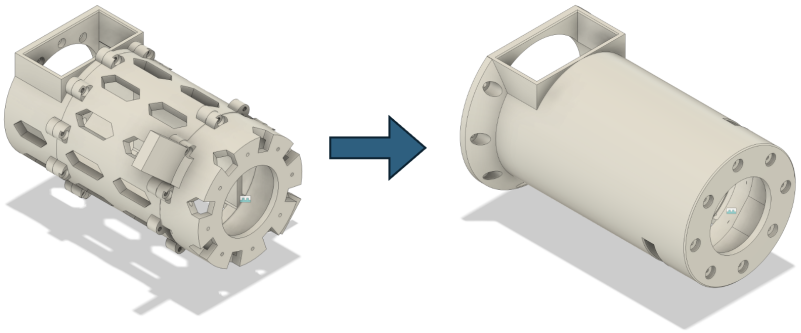


Figure 3-16: An example of a merged part during the iterative design process. The original base consists of three separated 3D printed parts, while the final design only consists of one merged part. This greatly reduced the required hardware and assembly time.

3.7.6 Thickened and Reinforced Parts

Testings revealed that certain sections of the robot arm were prone to failure, particularly during high-load or high-torque operations. This is particularly true at the motor mounting sections, where the mounting plate needs to sustain the substantial shear force generated at the rotational plane. To minimize repair frequency and improve reliability, several components were significantly reinforced. Besides the motor mounting parts, sections at the linkage-to-3D printed part interfaces where a press-fit connection was used were also extremely critical and were prone to fractures. When printed with thin walls, these press-fit sections tended to crack under load, leading to repeated failures. By increasing wall thickness and redesigning the connection geometry for greater rigidity, these components became far more resistant to high torque and external disturbances, substantially improving the armâĂŹs durability in experimental use. Examples of those reinforced parts can be seen in Figure 3-17.

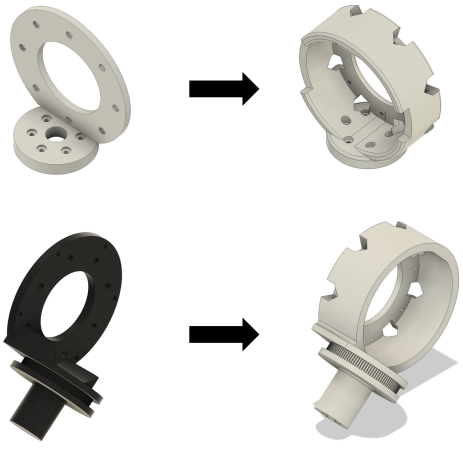


Figure 3-17: Examples of the thickened parts.

Chapter 4

Final Design

Figure 4-1: Final design: 3D CAD views from multiple angles showing actuator placement, elbow belt transmission, and the integrated wrist differential. This configuration balances weight reduction, mechanical simplicity, and dexterity, suitable for cinema-grade robotic motion.

4.1 Specifications

The final design, shown in Figure 4-4, supports a 1.5 kg professional camera payload at the end-effector while maintaining sufficient backdrivability, kinematic decoupling, and reach for high-speed tracking shots and cinematic motion control. The assembled 3D CAD renders from multiple perspectives are illustrated in Figure 4-1. The robot arm is constructed from only 23 custom 3D-printed parts, significantly reducing assembly complexity. A detailed

breakdown of these components is provided in Figure 4-5. The overall specifications of the final robot arm are summarized in Table 4.1.

Table 4.1: Specifications of the Final Cinema Robot Arm Design

Specification	**Value / Description**
Degrees of Freedom (DoF)	6
Total Mass (incl. electronics)	$\sim$8.5 kg
Payload Capacity	1.5 kg
Workspace Dimensions	Reach $\sim$0.7 m, total sweep diameter $\sim$1.4 m
Repeatability	$\pm$ 5 mm
Power Consumption	$\sim$60-100 W (average), $\leq$300 W peak (all joints at max torque)
Estimated Cost	\$800-\$1000
Assembly Time	$\sim$1-2 hours

4.2 DH Table

Table 4.2: Denavit-Hartenberg Parameters of the Final Robot Arm

i	α_{i-1}	a_{i-1} **(mm)**	d_i **(mm)**	θ_i
1	-90°	0	314.5	θ_1
2	0°	66.5	0	θ_2
3	0°	298.53	0	θ_3
4	0°	325.12	0	θ_4
5	$+90^\circ$	0	0	θ_5
6	0°	0	42.82	θ_6

The kinematic specification follows the DH parameters in Table 4.2. The wrist is a *spherical* (intersecting-axes) differential design: the roll, pitch, and yaw axes intersect at the end-effector's wrist center. Consequently, inverse kinematics decomposes cleanly into (i) a **position subproblem** for the first three (shoulder elbow) joints to place the wrist center, and (ii) an **orientation subproblem** for the last three wrist joints to realize the desired end-effector rotation. With all three wrist link lengths and offsets zero (ideal spherical wrist: $a_3 = a_4 = a_5 = 0$, $d_4 = d_5 = d_6 = 0$ except any final tool flange offset), the rotation matrix factors as

$$ {}^0R_6 = {}^0R_3 \, {}^3R_6, $$

allowing ${}^0R_3$ to be obtained from the wrist center position solution and ${}^3R_6$ solved directly for the three wrist joint

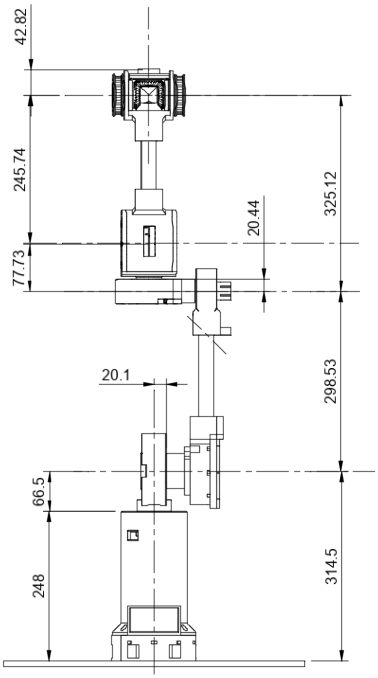


Figure 4-2: DH table dimension illustration of the final cinema robot design

angles. This structural decoupling simplifies analytic inverse kinematics, improves numerical conditioning, and reduces computational cost.

4.3 Final Build

The final build of the cinema robot arm utilizes all-black filament to resemble professional commercial cinema equipment. As the robot is intended for general consumer use, the design emphasizes a non-intimidating and user-friendly appearance. Several minor aesthetic enhancements were incorporated: a black protective sleeve was added to conceal the motor wiring, all electronic components were enclosed within the structure, and a standard hot shoe mount was integrated to facilitate the attachment of camera equipment using a quarter-inch screw interface.

Figure 4-3: Final construction of the cinema robot arm

4.4 Bill of Materials (BOM)

The hardware of the design focuses on using the off-the-shelf, easily accessiable componments. Those includes bearings, carbon fiber rods, timing belts, screws and nuts. All the componments can be found either on Amazon or Aliexpress. The detailed BOM is listed as follows:

Table 4.3: Preliminary Bill of Materials (BOM) for cinema robot arm prototype.

Category	Item / Spec	Qty	Unit (USD)	Link
Actuators	Unitree Go-1 actuator	6	$69.65	Taobao
Linkages	Carbon fiber square tube 25 mm width, 2 mm thickness, 500 mm length	1	$27.40	AliExpress
Bearings	Deep groove 26mm×17mm×5mm (OD×ID×Depth)	2	$1.59	AliExpress
	Deep groove 50mm×40mm×6mm (OD×ID×Depth)	6	$2.61	Amazon
	Deep groove 42mm×30mm×7mm (OD×ID×Depth)	5	$2.43	AliExpress
Transmission	HTD-5M rubber timing belt, 150 teeth (750 mm)	1	$15.19	Amazon
	HTD-5M rubber timing belt, 160 teeth (800 mm)	2	$15.56	Amazon
Fasteners	M4 screws and nuts set	1	$16.69	Amazon
	M3.5 screws and nuts set	1	$10.35	Amazon
Sensors	Intel RealSense	1	$163.63	AliExpress
Misc.	Wire sleeving / braided loom	1	$9.26	Amazon
Electronics	Jetson Nano	1	$216.15	AliExpress
	RS-485 hub / adapter	1	$0.99	AliExpress
	Main power supply ($\geq$300 W)	1	$34.97	AliExpress
3D Printed Material	PLA filament, 30% infill, honeycomb interior (Bambulab)	1	$16.99	Bambulab
		Total	**$991.63 USD**	

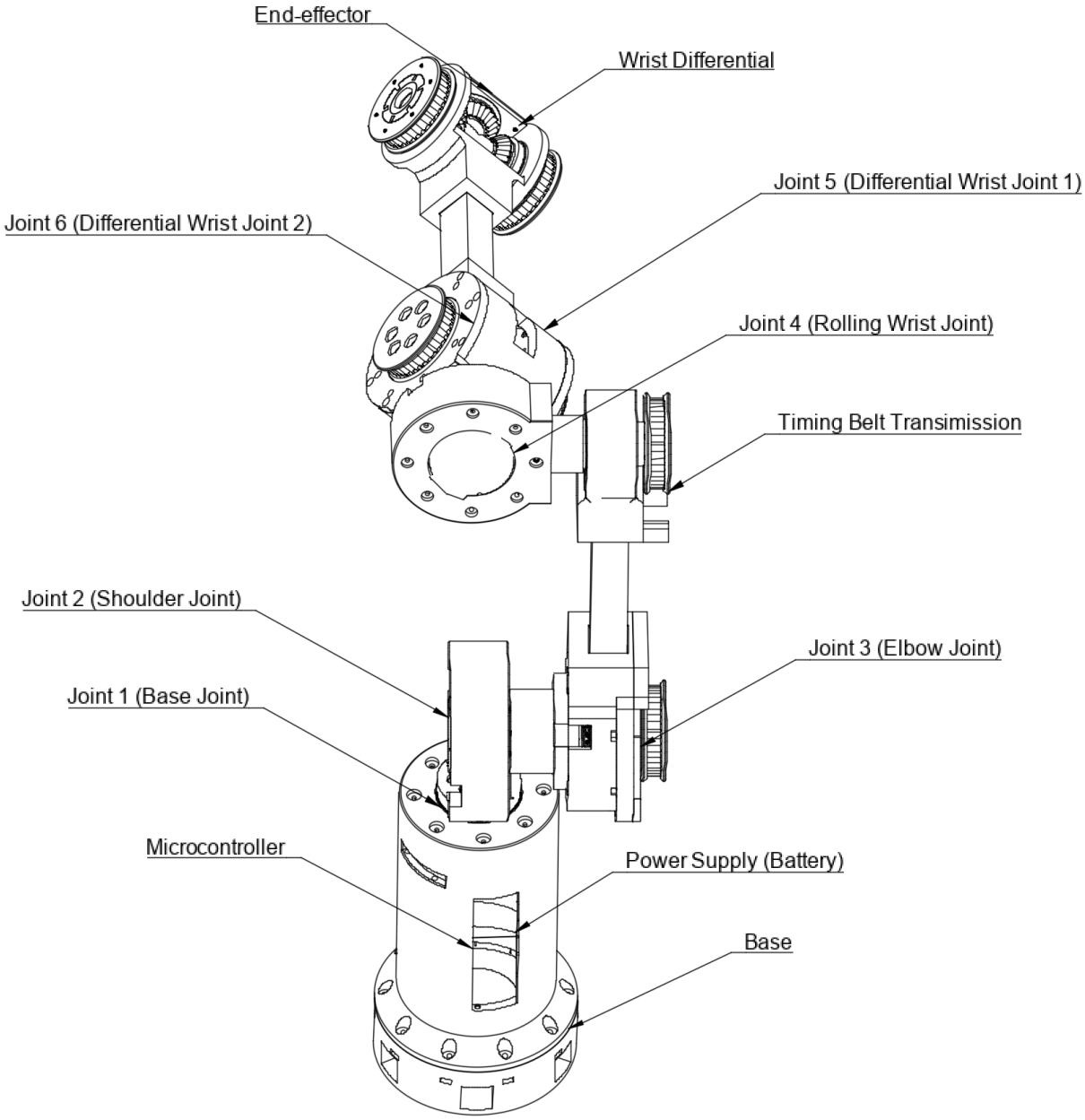


Figure 4-4: Final design components and layout illustrations

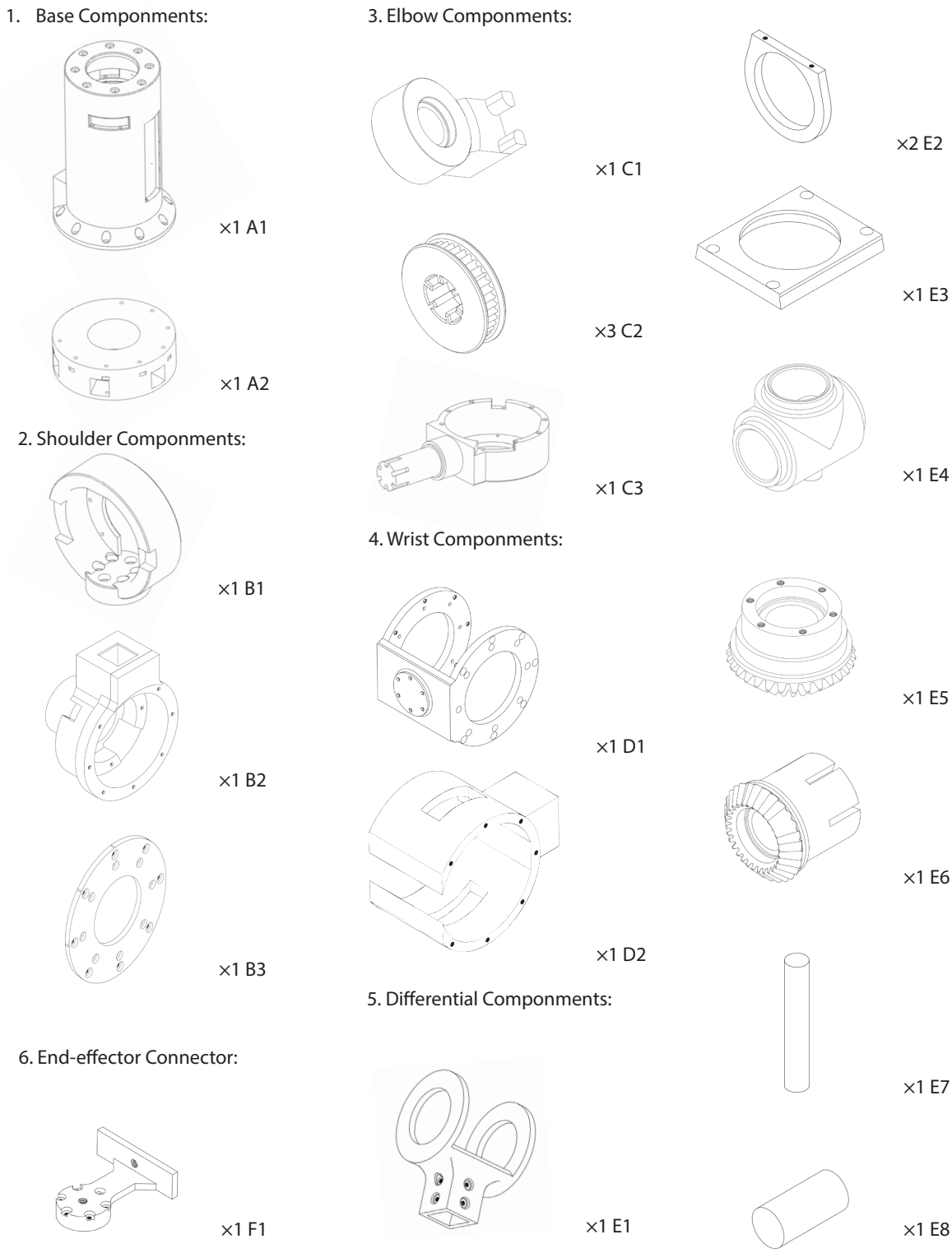


Figure 4-5: Final design all 3D printed components

Figure 4-6: Final build of *IRIS*

Chapter 5

Kinematics

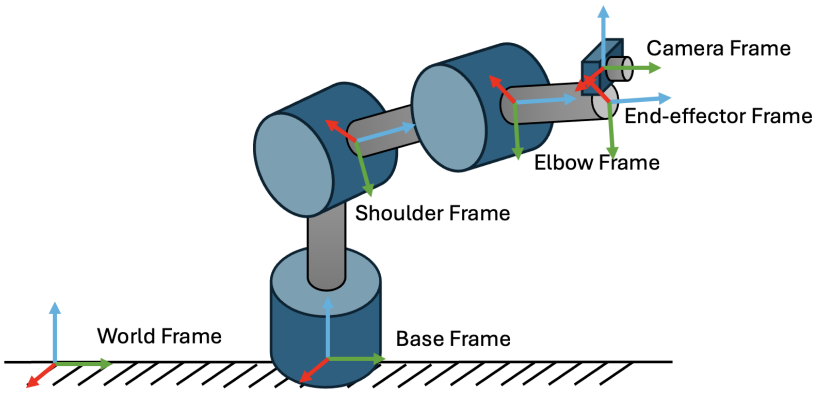


Figure 5-1: Illustration of the robot frames, camera frame, and end-effector frame. Joint frames are simplified for clarity.

5.1 Forward Kinematics

Forward kinematics (FK) maps joint angles $\mathbf{q} = [\theta_1, \ldots, \theta_6]^\top$ to the end-effector pose.

Definition.

$$\mathbf{T}_0^6(\mathbf{q}) = \begin{bmatrix} \mathbf{R}_0^6(\mathbf{q}) & \mathbf{p}_{EE}(\mathbf{q}) \\ \mathbf{0}^\top & 1 \end{bmatrix}, \quad \mathbf{R}_0^6 \in SO(3), \;\; \mathbf{p}_{EE} \in \mathbb{R}^3. \tag{5.1}$$

Position (joints 1–3).

$$\mathbf{T}_0^3(\theta_1, \theta_2, \theta_3) \;=\; \mathbf{T}_0^1\, \mathbf{T}_1^2\, \mathbf{T}_2^3, \tag{5.2}$$

$$\mathbf{p}_w(\theta_1, \theta_2, \theta_3) \;=\; \left[\mathbf{T}_0^3\right]_{1:3,\,4}, \tag{5.3}$$

$$\mathbf{p}_{EE}(\mathbf{q}) \;=\; \mathbf{p}_w(\theta_1, \theta_2, \theta_3) \;+\; d_6\, \mathbf{R}_0^6(\mathbf{q})\, \hat{\mathbf{z}}_6. \tag{5.4}$$

Orientation (joints 4–6).

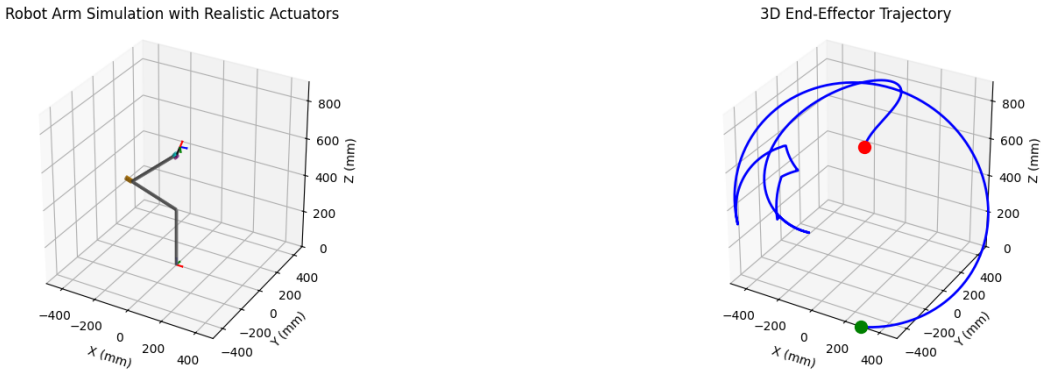
$$\mathbf{R}_3^6(\theta_4, \theta_5, \theta_6) \;=\; \mathbf{R}_x(\theta_4)\, \mathbf{R}_y(\theta_5)\, \mathbf{R}_z(\theta_6), \tag{5.5}$$

$$\mathbf{R}_0^6(\mathbf{q}) \;=\; \mathbf{R}_0^3(\theta_1, \theta_2, \theta_3)\, \mathbf{R}_3^6(\theta_4, \theta_5, \theta_6). \tag{5.6}$$

Final form.

$$\mathbf{T}_0^6(\mathbf{q}) \;=\; \begin{bmatrix} \mathbf{R}_0^3(\theta_1, \theta_2, \theta_3)\, \mathbf{R}_3^6(\theta_4, \theta_5, \theta_6) & \mathbf{p}_w(\theta_1, \theta_2, \theta_3) + d_6\, \mathbf{R}_0^6(\mathbf{q})\, \hat{\mathbf{z}}_6 \\ \mathbf{0}^\top & 1 \end{bmatrix}. \tag{5.7}$$

This explicitly decouples FK: joints 1-3 determine the wrist-center position, while joints 4-6 determine orientation about that point. Furthermore, the forward kinematics was verified and testing on both simulation in Python and *IRIS*. The simulation results can be seen in Figure 5-2, where by commanding individual joints, the end-effector changes accordingly. This forward kinematics algorithm is also the backbone of the later simulation for inverse kinematics, and test for singularities. To deploy in the real-world testings, all the torque output at each joint is set to be zero. The actuators are only used for encoder position purpose. The result of the real-world test can be seen in Figure 5-3.



Simulation of the forward kinematics

3D visualization for the simulated FK

Figure 5-2: Forward kinematics simulation

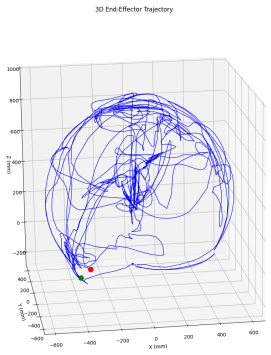


Figure 5-3: 3D visualization for real-world deployment of FK on *IRIS*

5.2 Inverse Kinematics

IRIS has a spherical wrist, so IK decouples: joints $1-3$ solve for the wrist-center position; joints $4-6$ solve for the orientation about that point.

Wrist center.

$$\mathbf{p}_w = \mathbf{p}_{EE} - d_6\,\mathbf{R}_0^6\hat{\mathbf{z}}_6, \qquad \mathbf{p}_w = [x_w,\, y_w,\, z_w]^\top. \tag{5.8}$$

Position (joints 1–3).

$$r = \sqrt{x_w^2 + y_w^2}, \qquad s = z_w - d_1, \tag{5.9}$$

$$\theta_1 = \operatorname{atan2}(y_w, x_w), \tag{5.10}$$

$$\cos\theta_3 = \frac{r^2 + s^2 - a_2^2 - a_3^2}{2a_2a_3}, \tag{5.11}$$

$$\theta_3 = \operatorname{atan2}\big(\pm\sqrt{1-\cos^2\theta_3}, \cos\theta_3\big), \tag{5.12}$$

$$\theta_2 = \operatorname{atan2}(s, r) - \operatorname{atan2}\big(a_3\sin\theta_3,\; a_2 + a_3\cos\theta_3\big). \tag{5.13}$$

Orientation (joints 4–6).

$$\mathbf{R}_3^6 = (\mathbf{R}_0^3)^\top\mathbf{R}_0^6 = \mathbf{R}_x(\theta_4)\,\mathbf{R}_y(\theta_5)\,\mathbf{R}_z(\theta_6), \tag{5.14}$$

$$\theta_5 = \operatorname{atan2}\big(r_{13}, \sqrt{r_{11}^2 + r_{12}^2}\big), \tag{5.15}$$

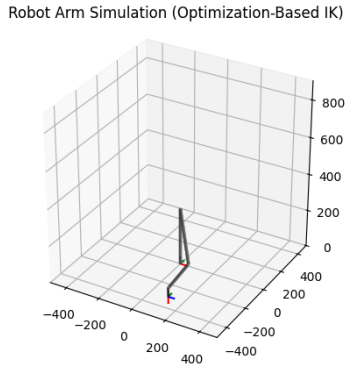
$$\theta_4 = \operatorname{atan2}(-r_{23}, r_{33}), \tag{5.16}$$

$$\theta_6 = \operatorname{atan2}(-r_{12}, r_{11}), \tag{5.17}$$

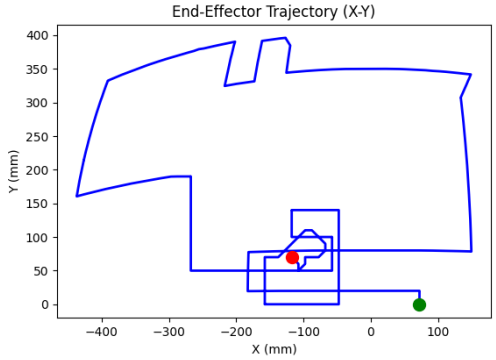
where r_{ij} are the entries of $\mathbf{R}_3^6$. At the wrist singularity ($\cos\theta_5 \approx 0$), fix θ_6 (e.g. 0) and recover θ_4 from $\operatorname{atan2}(r_{21}, r_{22})$ or use damped least squares.

Final analytical solution.

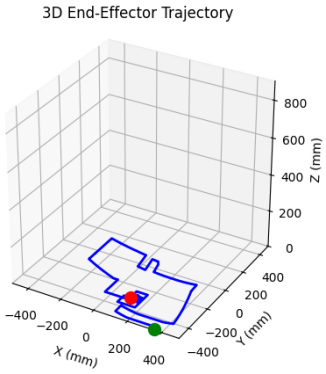
$$\begin{bmatrix}\theta_1\\ \theta_2\\ \theta_3\\ \theta_4\\ \theta_5\\ \theta_6\end{bmatrix} = \begin{bmatrix} \operatorname{atan2}(y_w, x_w) \\ \operatorname{atan2}(s, r) - \operatorname{atan2}\big(a_3\sin\theta_3,\; a_2 + a_3\cos\theta_3\big) \\ \operatorname{atan2}\big(\pm\sqrt{1-\cos^2\theta_3}, \cos\theta_3\big) \\ \operatorname{atan2}(-r_{23}, r_{33}) \\ \operatorname{atan2}\big(r_{13}, \sqrt{r_{11}^2 + r_{12}^2}\big) \\ \operatorname{atan2}(-r_{12}, r_{11}) \end{bmatrix}. \tag{5.18}$$



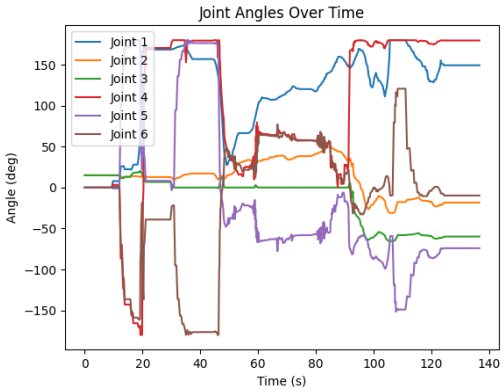
Simulation of *IRIS* teleoperated using inverse kinematics



2D view of the end-effector trajectory



3D view of the end-effector trajectory



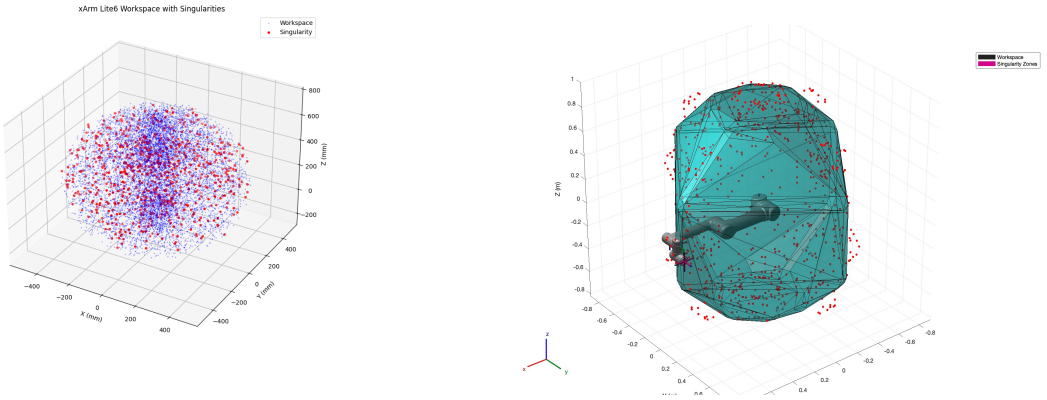
Joint states of *IRIS* under inverse kinematics

Figure 5-4: Simulated inverse kinematics using analytical solution

We also evaluated the IK performance in a simulated environment developed in MATLAB, as seen in Figure 5-4. In the simulation, the end-effector was teleoperated via keyboard inputs to various target positions, and the resulting end-effector trajectory, both in 3D and 2D, along with the corresponding joint angles, was plotted over time. The simulation results demonstrate the overall stability of the IK solver. However, in the presence of singularities, sharp spikes appear in the joint angle plots, indicating instability and non-ideal behavior. These singularity-related issues are further analyzed in a subsequent section using the MATLAB simulation framework.

5.3 Singularities

Singularities are a critical bottleneck in manipulation: they can amplify joint velocities and torques, risking actuator damage, loss of control, and unsafe behavior. Ensuring a safe workspace for *IRIS* therefore requires a clear understanding of where singularities arise and how to avoid them. Although robust handling of singularities remains an active research problem, this section outlines the common cases relevant to our 6-DoF arm with a spherical wrist, leveraging its position-orientation decoupling to localize (i) elbow/planar singularities in the positioning chain and (ii) wrist gimbal-lock in the orientation chain. We also did a simulation on the singularities based on the analytical solutions of the analysis.



(a) Velocity/torque blow-up near $w(\mathbf{q}) \to 0$.

(b) Workspace (blue) and singular samples (red).

Figure 5-5: Singularity analysis: (a) hardware-oriented simulation; (b) joint-space sampling with manipulability thresholding.

A configuration is *singular* when the geometric Jacobian $J(\mathbf{q})$ loses row rank:

$$\dot{\mathbf{x}} = J(\mathbf{q})\,\dot{\mathbf{q}}, \qquad \boldsymbol{\tau} = J(\mathbf{q})^{\top}\mathbf{w}, \qquad J < m \;\Leftrightarrow\; \sigma_{\min}(J) = 0. \tag{5.19}$$

For a 6-DoF arm with a spherical wrist, singularities typically arise in three familiar situations: (i) **elbow/planar** alignment when the upper arm and forearm become collinear, (ii) **boundary** configurations at the edge of reachable workspace where geometric leverage collapses, and (iii) **wrist gimbal-lock** when two wrist axes align (e.g., the middle wrist at $\pm 90^\circ$ in an XYZ wrist), removing one rotational DoF. The following sections analyze each case and present the corresponding closed-form relations at singularity, together with numerically robust remedies used in our system.

Wrist Singularity

Wrist gimbal-lock happens when two wrist axes align (e.g., the middle wrist joint at $\pm 90^\circ$ in an XYZ wrist). One rotational DOF is lost, orientation changes become ambiguous, and IK may cause θ_4 and θ_6 to spin in opposite directions unless the orientation task is relaxed or constrained. Nominal extraction from $\mathbf{R}_3^6 = \mathbf{R}_x(\theta_4)\mathbf{R}_y(\theta_5)\mathbf{R}_z(\theta_6)$ with entries r_{ij}:

$$\theta_5 = \operatorname{atan2}\!\big(r_{13}, \sqrt{r_{11}^2 + r_{12}^2}\big), \quad \theta_4 = \operatorname{atan2}(-r_{23}, r_{33}), \quad \theta_6 = \operatorname{atan2}(-r_{12}, r_{11}). \tag{5.20}$$

At *gimbal lock*, $\cos\theta_5 = 0$ ($\theta_5 = \pm\frac{\pi}{2}$, thus $r_{11} = r_{12} = 0$), only one combination of (θ_4, θ_6) is observable:

$$\theta_5 = +\tfrac{\pi}{2}: \qquad \theta_4 + \theta_6 = \psi^{+} \triangleq \operatorname{atan2}(r_{21}, r_{22}), \tag{5.21}$$

$$\theta_5 = -\tfrac{\pi}{2}: \qquad \theta_4 - \theta_6 = \psi^{-} \triangleq \operatorname{atan2}(-r_{21}, r_{22}). \tag{5.22}$$

Equivalent free-parameter form ($\gamma \in \mathbb{R}$):

$$\theta_5 = +\tfrac{\pi}{2}: \qquad \theta_4 = \gamma, \quad \theta_6 = \psi^{+} - \gamma, \tag{5.23}$$

$$\theta_5 = -\tfrac{\pi}{2}: \qquad \theta_4 = \gamma, \quad \theta_6 = \gamma - \psi^{-}. \tag{5.24}$$

If enforcing a specific $\theta_6^{\star}$ (continuity), then

$$\theta_5 = \pm\tfrac{\pi}{2}, \qquad \theta_4 = \begin{cases} \psi^{+} - \theta_6^{\star}, & \theta_5 = +\tfrac{\pi}{2}, \\ \psi^{-} + \theta_6^{\star}, & \theta_5 = -\tfrac{\pi}{2}. \end{cases} \tag{5.25}$$

Planar/Elbow Singularity

Elbow (planar) singularities occur when the forearm aligns with the upper arm (fully extended or folded). In this state, one in-plane rotational DOF collapses, so small Cartesian motions demand very large joint velocities, making the arm appear 'stiff' or hard to move sideways. Boundary singularities occur at the edge of the workspace, when the wrist center reaches its maximum reach. Here, the geometry saturates, leverage is poor, and controllers may chatter as they struggle to maintain motion. For $\mathbf{p}_w = [x_w, y_w, z_w]^{\top}$, $r = \sqrt{x_w^2 + y_w^2}$, $s = z_w - d_1$:

$$\sin\theta_3 = 0 \ (\theta_3 = 0, \pi) \ \Rightarrow \ \theta_2 = \operatorname{atan2}(s, r), \qquad \theta_1 = \operatorname{atan2}(y_w, x_w), \tag{5.26}$$

and one in-plane rotational DOF is lost.

Chapter 6

Dynamics

6.1 Field Oriented Control (FOC)

The brushless DC (BLDC) actuator used in *IRIS* is driven by a field-oriented control (FOC) scheme to achieve high-precision motion, with tracking accuracy on the order of millimeters. Although this project did not cover the design of the FOC driver or the algorirthm, I think it is important to outline the working of the FOC algorithm to help understand the low-level feedback control done in the later chapter.

FOC operates by transforming the three-phase stator currents into a rotating reference frame aligned with the rotor's magnetic flux, thereby decoupling torque-producing and flux, producing current components.

This decoupling allows independent regulation of torque and flux via two orthogonal current control loops, enabling smooth torque output, fast dynamic response, and low torque ripple even at low speeds.

An outer impedance control loop computes the desired torque from position and velocity errors (plus feedforward terms), while the inner FOC current loops ensure that the commanded d and q axis currents are accurately tracked. This architecture combines the dynamic compliance of impedance control with the precision and responsiveness of FOC, making it well suited for robotic joint actuation.

Clarke & Park transforms. Measured phase currents are mapped to $\alpha\beta$ and then to dq using

$$\begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix} = \frac{2}{3} \begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \end{bmatrix} \begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix}, \tag{6.1}$$

$$\begin{bmatrix} i_d \\ i_q \end{bmatrix} = \begin{bmatrix} \cos\theta_e & \sin\theta_e \\ -\sin\theta_e & \cos\theta_e \end{bmatrix} \begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix}. \tag{6.2}$$

The inverse transforms generate v_a, v_b, v_c from the commanded v_d^*, v_q^*.

PMSM model in the dq frame. The electrical dynamics (per phase-equivalent) are

$$v_d = R_s i_d + L_d \tfrac{di_d}{dt} - \omega_e L_q i_q, \tag{6.3}$$

$$v_q = R_s i_q + L_q \tfrac{di_q}{dt} + \omega_e (L_d i_d + \lambda_m). \tag{6.4}$$

The electromagnetic torque is

$$\tau = \frac{3}{2} n_p \big(\lambda_m i_q + (L_d - L_q) i_d i_q\big). \tag{6.5}$$

For a surface-PMSM (SPMSM) with $L_d \approx L_q$, this reduces to

$$\tau \approx \frac{3}{2} n_p \lambda_m \, i_q. \tag{6.6}$$

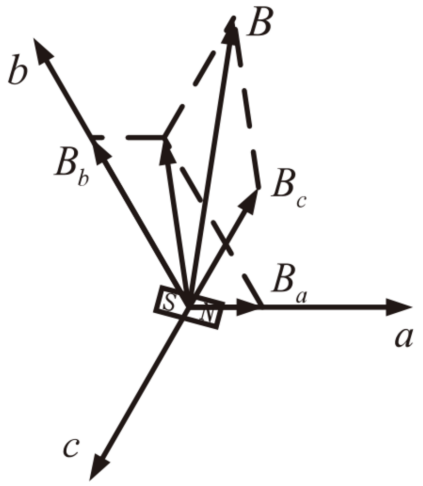


Figure 6-1: Illustration of the FOC vector transform.

Inner current control with decoupling. Let i_d^*, i_q^* be current references. Two PI regulators track them and add feedforward decoupling/back-EMF compensation:

$$v_{d,\mathrm{PI}} = K_{p,d}(i_d^* - i_d) + K_{i,d} \int (i_d^* - i_d)\, dt, \tag{6.7}$$

$$v_{q,\mathrm{PI}} = K_{p,q}(i_q^* - i_q) + K_{i,q} \int (i_q^* - i_q)\, dt, \tag{6.8}$$

$$v_d^* = v_{d,\mathrm{PI}} + R_s i_d - \omega_e L_q i_q, \tag{6.9}$$

$$v_q^* = v_{q,\mathrm{PI}} + R_s i_q + \omega_e (L_d i_d + \lambda_m). \tag{6.10}$$

The commanded $\boldsymbol{v}_{dq}^*$ is inverse-Park/Clarke transformed to $\boldsymbol{v}_{abc}^*$ and synthesized via SVPWM. Voltage and current limits enforce DC-bus and thermal constraints with anti-windup.

Torque mapping and references. For SPMSM (max-torque-per-ampere at low speed): $i_d^* = 0$ and

$$i_q^* = \frac{2}{3\, n_p\, \lambda_m}\, \tau^*. \tag{6.11}$$

For an interior PMSM (IPMSM), use an MTPA law to set $i_d^* < 0$; in field-weakening at high speed, further reduce i_d^* to satisfy $|\boldsymbol{v}_{dq}^*| \le V_{\max}$.

6.2 Impedance Control

Each BLDC motor employs impedance control to accurately track the desired position, velocity, and torque. Conceptually, impedance control can be modeled as a serial spring-damper system, where the stiffness term governs

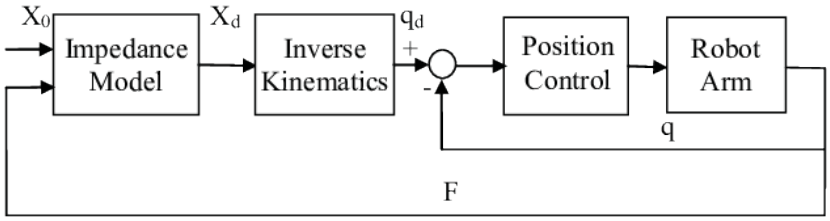


Figure 6-2: Overview of the impedance control flow chart in an actuator. This diagram does not include gravity compensation, in other words, there is no inverse dynamics compensation in the control loop

position tracking and the damping term governs velocity tracking. In more advanced implementations, if the inverse dynamics can be computed, a torque feedforward term can be incorporated into the total torque command to further enhance tracking performance. Overall, the control law can be expressed as:

$$\tau_{\text{cmd}} = K_p\big(\theta_d - \theta\big) + K_d\big(\dot{\theta}_d - \dot{\theta}\big) + \tau_{\text{ff}}, \tag{6.12}$$

$$i_q^* = \frac{\tau_{\text{cmd}}}{K_t} = \frac{1}{K_t}\Big[K_p(\theta_d - \theta) + K_d(\dot{\theta}_d - \dot{\theta}) + \tau_{\text{ff}}\Big], \tag{6.13}$$

where τ_{cmd} is the desired torque output from the motor; and i_q^* is the desired current output from the FOC motor. This equation provides an intuitive mapping from the required current input to the resulting torque output.

The overview diagram of how each components flow to each can be seen from the diagram below: To validate the control algorithm, a dedicated actuator test stand was designed and constructed, as shown below:

The test stand serves two major purposes:

1. Test the individual actuator's repeatability
2. Test the individual actuator's position tracking

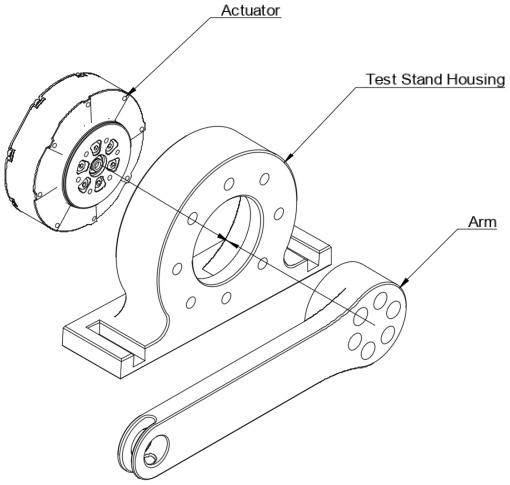


Figure 6-3: Actuator test stand illustration

1. To conduct the repeatability test, a dial is placed at the end of the test stand arm, so that the actuator can move the arm (first test without load, second test without load) to the preset location. The dial would then measure the difference between each iterations. The real-life test stand for repeatability is shown in Figure 6-4, where the test stand housing and the arm are 3D-printed components. The iterations and the amount of error per trail is plotted in Figure 9-2 in Section 9, where it shows a progressively increasing error of the motor.

Figure 6-4: The repeatability test stand of the actuator.

2. For the Position tracking test, we evaluate how the actuator's position-tracking performance degrades under externally applied loads at the end effector (arm tip). Specifically, a known payload is mounted at the tool flange and the joint is commanded to follow reference trajectories (steps, ramps, and sinusoids across a frequency sweep). We record the measured joint angle at high rate and quantify tracking via steady-state error, peak overshoot, rise/settling time, RMS error over a cycle, and closed-loop bandwidth. To isolate load effects, we repeat the protocol across multiple payloads (including the nominal camera mass) and vary the trajectory amplitude/speed to probe both low- and high-velocity regimes. The controller (PID with feedforward gravity/viscous terms unless otherwise stated) and all gains are held fixed across conditions. Together, these trials characterize the actuator's ability to maintain precise positioning under realistic end-effector loads and provide operating envelopes for safe, repeatable motion.

Figure 6-5: Dial used in the accuracy testings

6.3 Gravity Compensation

Having only impedance control works well under light or no load, where tuning the stiffness and damping gains can yield sub-degree rotational accuracy. However, performance degrades with heavy or load-varying conditions (e.g., along the motion), where unmodeled gravity, friction, and coupling terms dominate. This was a major bottleneck in our system. To address it, we add torque feedforward to provide gravity and disturbance compensation on top of impedance control.

The implementation is as follows: Based on Lagrangian physics, the full dynamics of the 6 DOF robot arm can be expressed as:

$$\mathbf{M}(\mathbf{q})\,\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\,\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}) = \boldsymbol{\tau}, \tag{6.14}$$

where $\mathbf{M}$ is the inertia matrix, $\mathbf{C}\dot{\mathbf{q}}$ the Coriolis/centrifugal vector and $\mathbf{G}$ the gravity vector.

Because static positioning accuracy is limited mainly by $\mathbf{G}$, we cancel it in feedforward and leave the feedback loop to correct only residual modeling error.
For the present arm-links modelled as mass-less tubes and actuators treated as point masses located on their joint axes-the closed-form gravity term is

$$\mathbf{G}(\mathbf{q}) = g \begin{bmatrix} 0 \\ (m_3 + m_w) L_1 \cos q_2 + m_w L_2 \cos(q_2 + q_3) \\ m_w L_2 \cos(q_2 + q_3) \\ 0 \\ 0 \\ 0 \end{bmatrix}, \tag{6.15}$$

with $m_w = m_4 + m_5 + m_6$, link lengths L_1, L_2, and standard trigonometric shorthands. The resulting hybrid control law is

$$\boldsymbol{\tau}_{\text{cmd}} = \mathbf{G}(\mathbf{q}) + \mathbf{K}_p(\mathbf{q}_d - \mathbf{q}) + \mathbf{K}_d(\dot{\mathbf{q}}_d - \dot{\mathbf{q}}), \tag{6.16}$$

where diagonal gains $\mathbf{K}_p, \mathbf{K}_d$ are tuned just high enough for the desired closed-loop bandwidth because gravity no longer creates steady-state error.

In software (1 kHz loop) we evaluate (6.15) symbolically, divide each component by its gearbox ratio to obtain rotor-side torques, clamp to the motor limits, and finally add the PD term (6.16).

In real-world testing however, several issues led to unsatisfactory results. First, the system dynamics model was inaccurate. Because *IRIS* uses 3D-printed components, the part masses and inertia are not well characterized, introducing modeling discrepancy. Second, assembly tolerances are non-negligible. Rotational axes, such as the elbow and shoulder, can be misaligned, producing residual errors in the Denavit-Hartenberg (DH) parameters. Together, these factors yield incorrect model-based (theoretical) torque feedback.

Chapter 7

Classical Controls

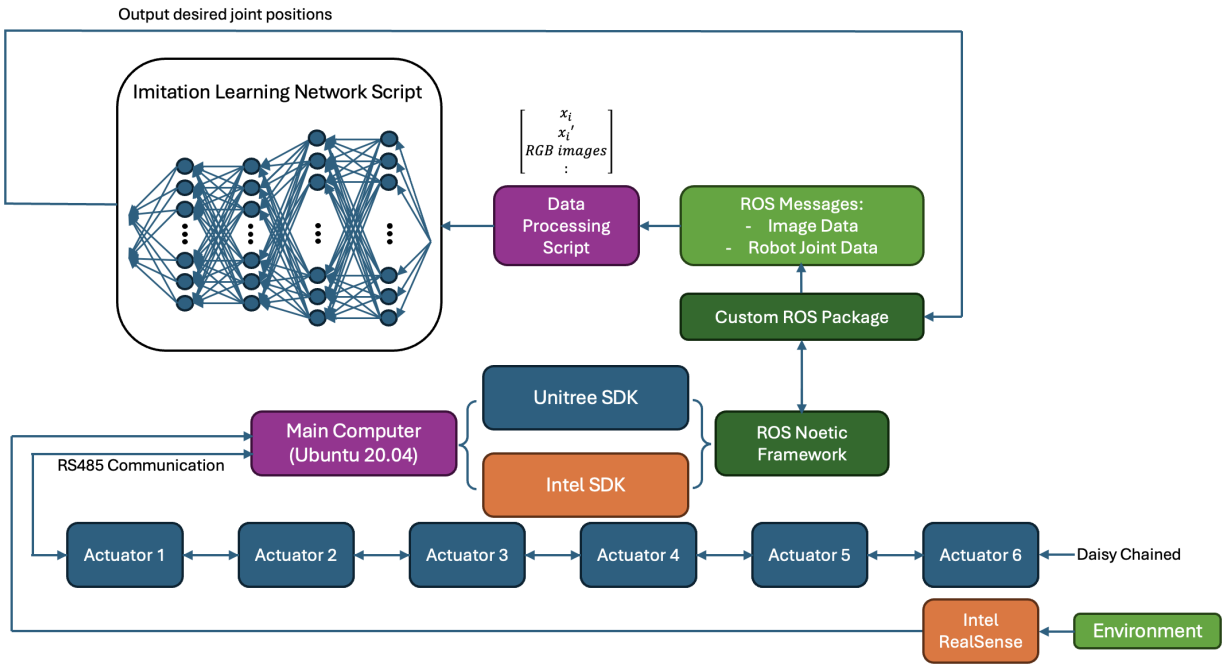


Figure 7-1: Full stack of the cinema robot arm framework illustration

Before fully realizing autonomous path planning on *IRIS*, foundational teleoperation and baseline path-execution capabilities must be in place. This chapter details the teleoperation pipeline and a teach-and-repeat mechanism that leverages *IRIS*'s high-resolution encoders.

7.1 Framework Overview

The control system runs on Ubuntu 20.04 with ROS Noetic. A Python control layer interfaces with the Unitree SDK to issue low-level FOC commands, while each *IRIS* actuator integrates BLDC drive electronics, encoder-based position/velocity sensing, and torque estimation for precise, smooth motion. The full stack comprises six integrated

actuators, a depth/RGB camera, vendor SDKs for sensing and control, and a custom ROS package for data collection and inter-process communication; a high-level control layer can be instantiated as either imitation-learning policies or classical controllers. A detailed system diagram is shown in Figure 7-1. In the configuration depicted, high-level actions are produced by an imitation-learning policy (see chapter 8); for baseline operation, the same interface accepts classical control outputs (e.g., joint setpoints from teleoperation or IK-based Cartesian commands), allowing modules to be swapped without changes to the low-level interface on *IRIS*.

7.2 ROS Integration

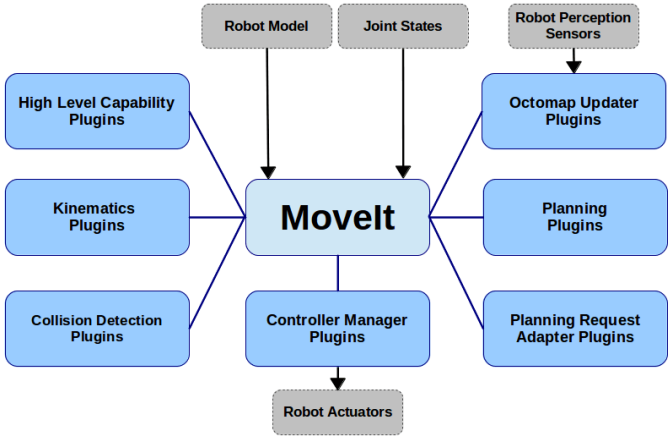


Figure 7-2: MoveIt! framework overview

ROS Noetic is used on Ubuntu 20.04 to provide a mature ecosystem (RealSense drivers, MoveIt! IK, RViz, rosbag), standardized message interfaces (sensor_msgs/Image, sensor_msgs/JointState, geometry_msgs/PoseStamped), and reproducibility tooling (tf2 for transforms, message_filters for time sync). These choices reduce integration effort and let the same node graph support both classical controllers and learned policies on *IRIS*.

On *IRIS*, ROS bridges vendor SDKs and high-level control. Joint states are published as JointState; RGBD streams are published as Image with synchronized CameraInfo, aligned via message_filters. A tf2 tree (base → links → end-effector → camera) derived from URDF/Xacro keeps frames consistent for Cartesian commands and logging. Teleoperation and imitation-learning nodes emit joint or Cartesian targets that an IK layer maps to trajectories. RViz provides live visualization, while rosbag records synchronized RGBD and joint data for datasets. A watchdog monitors temperatures, torque estimates, and limits, preempting motion on faults.

ROS 1 is preferred over ROS 2 in this build due to broader package availability on Ubuntu 20.04 (RealSense, existing Unitree wrappers, MoveIt!), which minimizes integration risk. The node graph cleanly separates low-level I/O, kinematics/supervision, and high-level modules (teleoperation, teach-and-repeat, imitation learning), and namespacing keeps the design ready for multi-*IRIS* operation.

7.3 Teleoperation

Teleoperation provides a simple, reliable means to position *IRIS* and to validate accuracy before fully autonomous operation. Two modes are implemented:

(1) *Individual Joint Control* - Each joint can be commanded independently to a desired position, with optional simultaneous moves across multiple joints. This mode is used for setup, calibration, and fine adjustments.

(2) *End-Effector Control* - Desired Cartesian pose commands are mapped to joint angles via the inverse-kinematics solver derived in chapter 5 (Chapter 4). This mode supports precise manual framing for cinematic shots.

7.4 Teach-and-Repeat

Algorithm 1: Teleoperation and Teach-and-Repeat Control Flow on *IRIS*

Data: ROS topics: joint states, Cartesian targets, joint targets; Unitree SDK API for low-level FOC
Result: Real-time teleoperation or recorded-trajectory playback
Initialization:
  Start ROS master (Ubuntu 20.04);
  Launch Python control node;
  Connect to Unitree SDK over Ethernet (UDP);
  Initialize joint-state subscribers and command publishers;
  Home and calibrate all joints on *IRIS*;
Teleoperation Mode:
  if `mode = JointControl` **then**
    Read joint position commands from UI;
    Publish joint setpoints to Unitree SDK (position control);
  else
    `mode` $\leftarrow$ `EndEffectorControl`;
    Read Cartesian pose commands from UI;
    Solve IK (chapter 5) for target joint angles;
    Publish joint setpoints to Unitree SDK;
  end
Teach-and-Repeat:
  Teach phase:
    Enable backdrivable/low-stiffness mode on all joints;
    Record $\{\mathbf{q}(t), \dot{\mathbf{q}}(t), t\}$ from encoders;
  Repeat phase:
    Move to home for a collision-free start;
    Replay recorded trajectory: stream joint setpoints synchronized to recorded timestamps;
Continuous Safety Supervision:
  Monitor joint states and SDK status;
  Abort on over-torque, over-temperature, or user stop;

Teach-and-repeat lets an operator physically demonstrate a camera move once and have *IRIS* reproduce it consistently. Using the Unitree actuator FOC controllers with high-resolution encoders (approximately 0.1° per joint under no load), *IRIS* records joint positions, velocities, and timestamps during the demonstration. After confirmation, *IRIS* returns to a safe home configuration to avoid collisions, then executes the captured trajectory with

time-synchronized playback. This workflow is useful for repeatability testing and for recreating identical shots across takes while preserving expert-designed motion paths.

The pseudocode in algorithm 1 summarizes the control logic used by both teleoperation modes and the teach-and-repeat routine.

This architecture leverages the real-time capabilities of the Unitree SDK for low-level actuation while ROS provides message passing, kinematics, safety supervision, and mode switching. The Python layer serves as the high-level coordinator on *IRIS*, translating user inputs or learned policy outputs into precise joint commands and ensuring reproducible trajectory execution via teach-and-repeat.

Chapter 8

Imitation Learning

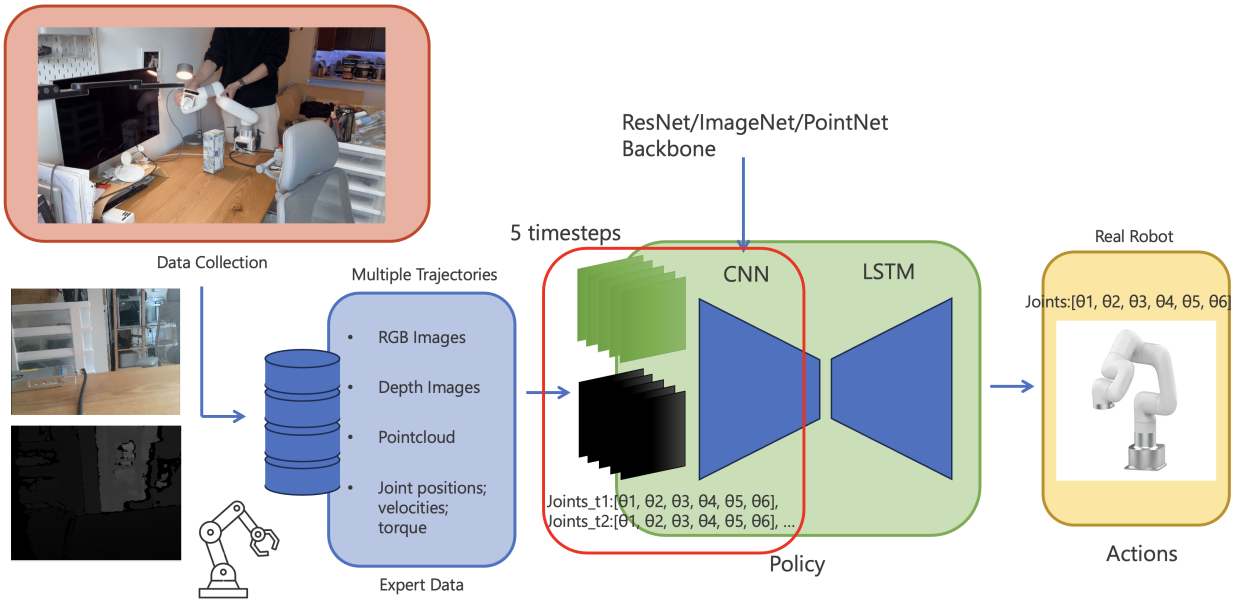


Figure 8-1: Imitation learning pipeline illustration

The primary objective of the control system for *IRIS* is to achieve autonomous, real-time path planning and obstacle avoidance while maintaining smooth, cinematic motion in unstructured environments.

Traditional Model Predictive Control (MPC), though powerful for constrained optimal control, requires solving sizeable optimizations at each control step, which burdens on-board compute, especially when paired with high-rate perception. Even with recent advances in sampling-based and information-theoretic MPC that push real-time performance, the computational load and tuning effort remain significant on embedded platforms (15; 16). Moreover, MPC performance depends on accurate models; for *IRIS*'s fully 3D-printed, custom-built manipulator, variability in link compliance, joint friction, and payloads makes faithful identification challenging.

Reinforcement Learning (RL) offers another approach, but robust policies typically rely on a high-fidelity 'digital twin' and large-scale data generation (17; 18). World-model approaches reduce interaction cost yet still face transfer

issues when contact and actuator effects are imperfectly captured (19). For *IRIS*, constructing a precise digital twin is difficult due to mechanical tolerances, material flexibilities, leading to a nontrivial sim-to-real gap and risky on-hardware fine-tuning.

In contrast, IL avoids explicit modeling and large-scale simulation by learning directly from expert demonstrations. Deployed on *IRIS*, IL can also run within the available compute budget and support real-time inference without costly online optimization, while reproducing smooth, collision-free camera motions. Given its lack of explicit modeling requirements and low online compute cost, imitation learning (IL) is a natural candidate for autonomous path planning on *IRIS*. A more comprehensive comparison between the three approaches are listed in Table 8.1.

Table 8.1: Concise comparison of control paradigms for *IRIS*.

Method	Compute (train / infer)	Online opt.	Model reliance	Robustness	Zero-shot
MPC	- / *High*	Yes (QP/NLP)	*High* (dyn. + contacts)	Good under constraints; sensitive to model mismatch	Low-Mod.
RL (policy)	*Very high* / *Low*	No (eval)	*Med.* (sim fidelity)	Domain rand. helps; sim→real gaps persist	Mod.
IL / Diffusion	*High* / *Low-Mod.*	No	*Low* (model-free)	Stable in-distribution; diffusion smooths actions	Mod.

Notes. 'Compute' reports offline training vs. on-board inference cost. 'Zero-shot' = behavior on unseen scenes without task-specific finetuning. Representative refs: MPC (15; 16; 20); RL (17; 18; 19); IL/Diffusion (21; 22).

8.1 Dataset Collection

Why collect our own data on *IRIS*? Although large public datasets for autonomous manipulator path planning exist (see Table **??**), they do not reflect *IRIS*'s particulars. *IRIS* is a custom, largely 3D-printed manipulator with an end-effector-mounted RGB-D camera, off-the-shelf datasets cannot capture its morphology, cable routing, joint friction, and camera extrinsic. Collecting demonstrations yields image-action pairs from the correct moving viewpoint and implicitly encodes the arm's reachable workspace, self-collision envelope, and compliance without hand-crafted models or calibration priors. In practice, this lets the policy learn a robot-specific visuomotor mapping and local obstacle geometry directly from data, avoiding brittle state estimation about kinematics, link flex, or scene layout.

Table 8.2: Representative public datasets for robot-arm visuomotor IL/path planning. Abbreviations: MV = multi-view, D = depth, lang. = language.

Dataset	Scale	Modalities	Format	Platform(s)
Open X-Embodiment (OXE) (23)	1M+ traj.	RGB(+D), proprio, lang.	RLDS episodes	22 robot embodiments
DROID (24)	76k traj./350 h	MV RGB, D, proprio, lang.	Episodic loaders	Multi-lab real arms
BridgeData V2 (25)	60,096 traj., 13 skills	RGB(/D), proprio, lang./goal img.	Episodic loaders	Low-cost 6-DoF arm
RoboNet (26)	15M frames	RGB, proprio	TFDS (TensorFlow Datasets)	Sawyer, Baxter, Franka, KUKA, *etc.*
BC-Z (27)	∼12k demos (door), 100+ tasks	RGB, proprio, lang. goals	Episodes (Kaggle)	Mobile base + arm; tabletop
RLBench (sim) (28)	100+ tasks (sim)	RGB-D, state (sim)	NPZ/API	Franka-like arm (sim)

All the dataset used during training is collected via manual export demonstrations. To mimic how humans will move their arms while filming, an expert operator will drag the robot arm's end-effector to move from point A to point B following a desired trajectory. In the absence of obstructions, the end-effector traced an approximately straight line between the two waypoints. When obstacles intersected the nominal path, the demonstrator executed collision-avoidance manoeuvres, either skirting the object laterally or elevating the end-effector to pass above it. A separate data collection script is written so that the motors would provide zero torque, and would only be used for encoding the data only. The data collected from the actuators includes: timestamps of each data entry t, the position

at the timestep p, the velocity v and the output torque τ. Initially, to validate the imitation algorithm, an xArm Lite 6 is used for the data collection process coupled with an Intel RealSense D415 (Figure 8-2a). xArm Lite 6 contains 6 degrees of freedom with a payload at the end-effector of 500 grams (more details can be referred to Figure 8-2b). Each joint of the xArm is able to have full position, velocity and torque control, or the mix of all three. In addition, each joint is able to record the proprioceptive data. The Intel RealSense D415 has a resolution of 1920 $\times$ 1080 for RGB image and 1280 $\times$ 720 for depth image, each at 30 FPS and 90 FPS respectively. The detailed specs of the two hardware used for data collection are listed in the table 8.4 under.

Table 8.3: Proprioceptive packet logged at 100 Hz

Field	Dim.	Units	ROS type	Description
Timestamp t	1	µs	`int64`	UNIX epoch synchronised to camera clock
Joint position $\mathbf{q}$	6	rad	`sensor_msgs/JointState.position`	Absolute joint angles
Joint velocity $\dot{\mathbf{q}}$	6	rad s^{-1}	`JointState.velocity`	First-order finite difference of $\mathbf{q}$
Joint torque $\boldsymbol{\tau}$	6	N m	`JointState.effort`	Current-derived torque estimate

(a) Intel RealSense RGB-D camera

(b) xArm Lite 6 manipulator

Figure 8-2: Data collection hardware: (a) Intel RealSense camera; (b) xArm Lite 6 robot arm.

Table 8.4: Hardware specifications of the data-collection setup

Property	UFactory Lite 6 (29)	Intel RS D415 (30)	Notes
DoF	6	—	All revolute, ISO 9409-1-50 flange
Payload / Reach	0.6 kg / 440 mm	—	Desktop-scale cobot
Repeatability	$\pm$0.5 mm	—	Harmonic drives + abs. encoders
Max tip speed / joint speed	500 mm/s / 180 deg/s	—	Cartesian / axis limits
Joint ranges (J1–J6)	$\pm360°$, $\pm150°$, $-3.5°$:300°, $\pm360°$, $\pm124°$, $\pm360°$	—	Firmware soft limits
Encoders	Multi-turn abs. 16-bit	—	Exposed in SDK
Weight	7.2 kg	72 g	Camera = 13 % of arm payload
Interface / SDK	TCP (Eth.), Py/C++/ROS 2	USB 3.2 Gen 1, librealsense 2	Native Linux drivers
Camera-specific:			
RGB res./fps	—	1920×1080 @ 30 fps	Rolling-shutter sensor
Depth res./fps	—	1280×720 @ 90 fps	Active–IR stereo (55 mm baseline)
FoV (H $\times V$)	—	69.4° × 42.5°	Datasheet value
Range / Accuracy	—	0.5 m to 3 m / <2 % @ 2 m	Indoor/outdoor capable

The resulting dataset combines time-synchronized joint-space proprioception from the xArm, including six-axis's position, velocity, and output torque, with RGB-D images from Intel RealSense D415. This produces a sensor-complete record of hand-guided motion suitable for high-fidelity supervision in downstream imitation learning pipelines. To ensure diversity and improve generalization, the lighting conditions of the collection workspace were

systematically varied. Dataset were collected in different days with different natural light conditions, and artificial lights. And in addition, the robot arm was repositioned to alter the visual background. In addition, the expert-demonstrated trajectories were diversified, as illustrated in Figure 8-3, with motions directed toward different regions of the workspace to promote a more robust learned policy.

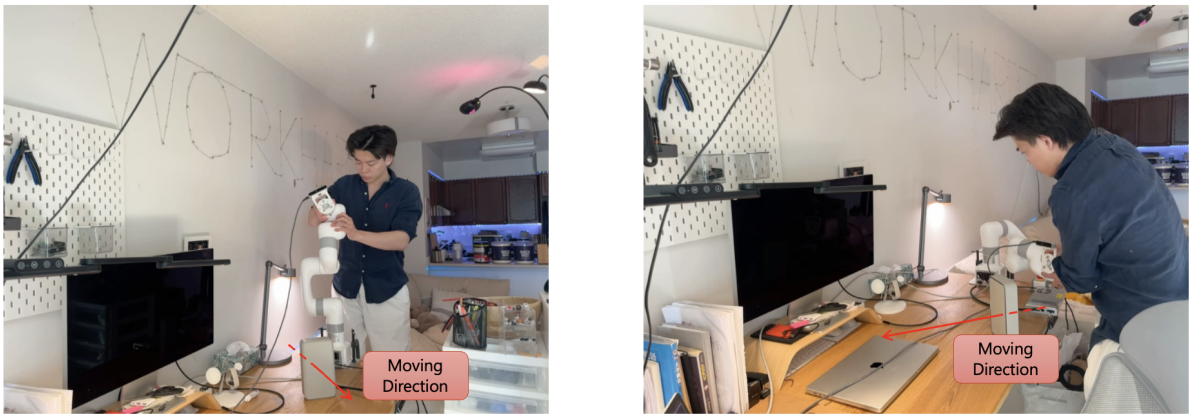


Figure 8-3: xArm data collection in various directions. In the figure two sample directions collected are shown as examples

8.2 Dataset Processing

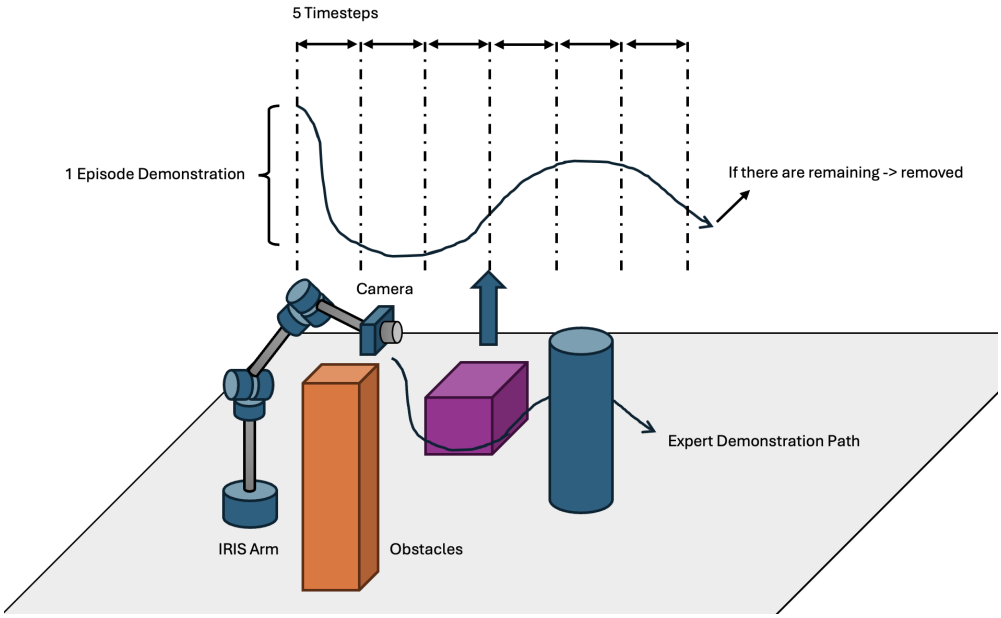


Figure 8-4: Illustration on how the data is being collected and processed.

The expert demonstration dataset was recorded as *ROS bag* files, each containing multiple consecutive *trajectories*. An offline parsing pipeline segmented each bag into discrete *episodes*, defined as single end-effector point-to-point motions whose duration depended on the selected Cartesian path. Each episode preserved the complete multi-modal state vector detailed in Table 8.3. Joint-state feedback from the xArm was streamed at 100 Hz, while the Intel RealSense RGB-D camera published image streams at 30 Hz. To obtain temporally consistent samples, the

proprioceptive triplet $[\mathbf{q},\ \dot{\mathbf{q}},\ \boldsymbol{\tau}]$ was linearly interpolated to the RGB-D camera timestamps, producing temporally aligned tuples of the form

$$\langle \mathbf{q}_k, \dot{\mathbf{q}}_k, \boldsymbol{\tau}_k, I_k^{\mathrm{RGB}}, I_k^{\mathrm{D}} \rangle, \quad k = 1, \ldots, N.$$

Initially, both RGB and depth modalities were considered as inputs to the perception network and processed jointly. As the raw RGB and depth frames were not hardware-synchronized, each RGB frame was matched to the nearest depth frame based on timestamp proximity. Raw depth images from the Intel RealSense sensor exhibited imperfections, including missing pixels and localized voids. To mitigate these effects, an initial preprocessing stage was applied, consisting of Intel's built-in hole-filling filter followed by a temporal smoothing filter to suppress abrupt pixel-wise depth variations. The joint-state data from the xArm was included without additional preprocessing, containing positions, velocities, and output torques for all joints. For ingestion by the imitation-learning policy, each episode was segmented using a sliding horizon of five consecutive frames, producing fixed-size spatio-temporal tensors that capture short-term dynamics while conforming to the input dimensionality requirements of the downstream network.

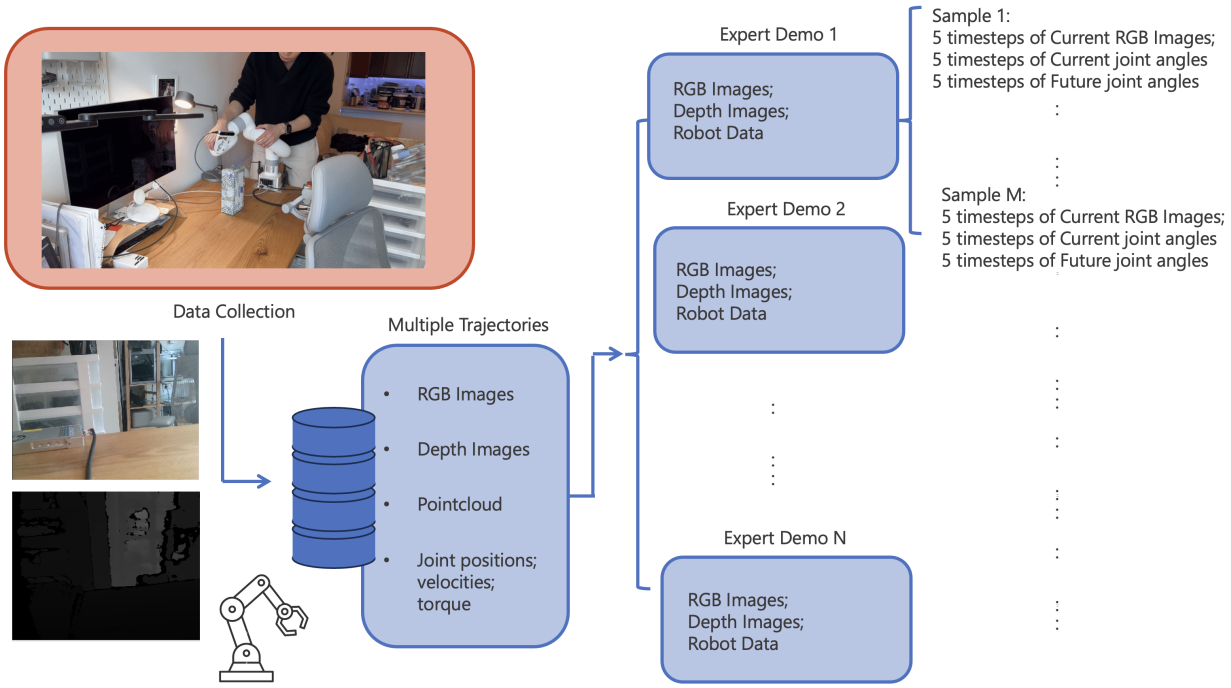


Figure 8-5: Pipeline for data processing

8.3 Loss Functions

The goal is to predict the future joint states given a sequence of past observations. Two loss terms are used: **Mean Squared Error (MSE) Loss**: Let $\hat{q}_{t+i}$ denote the predicted joint state at future timestep i, and q_{t+i} the corresponding ground truth. The standard MSE loss is given by:

$$\mathcal{L}_{\mathrm{MSE}} = \frac{1}{T_{\mathrm{future}}} \sum_{i=1}^{T_{\mathrm{future}}} \|\hat{q}_{t+i} - q_{t+i}\|^2.$$

Continuity Loss: To ensure that the predicted sequence is continuous with the observed past, we add a continuity loss that enforces the first predicted state to be close to the last observed state:

$$\mathcal{L}_{\text{cont}} = \lambda_{\text{cont}} \, \|\hat{q}_{t+1} - q_t\|^2,$$

where λ_{cont} is a weighting factor (set to 0.1 in our experiments).

The total loss becomes:

$$\mathcal{L} = \mathcal{L}_{\text{MSE}} + \mathcal{L}_{\text{cont}}.$$

Future loss functions to consider We keep the joint-space setting and avoid costly kinematics/maps. Let $\Delta\hat{q}_{t+i} = \hat{q}_{t+i} - \hat{q}_{t+i-1}$ with $\hat{q}_t \equiv q_t$, and similarly $\Delta q_{t+i} = q_{t+i} - q_{t+i-1}$.

(1) Early-Weighted MSE (focus on first steps).

$$\mathcal{L}_{\text{wMSE}} = \frac{1}{Z} \sum_{i=1}^{T_{\text{future}}} \alpha^{\,i-1} \, \|\hat{q}_{t+i} - q_{t+i}\|_2^2, \quad Z = \sum_{i=1}^{T_{\text{future}}} \alpha^{\,i-1}, \; \alpha \in (0,1).$$

(2) First-Step Activation Margin (hinge). Encourage a non-zero first move when not near completion (cheap gate via joint-space distance to a known goal q_g, if available).

$$\mathcal{L}_{\text{act}} = \mathbf{1}\{\|q_g - q_t\|_2 > \rho\} \, \max\bigl(0, \; m - \|\Delta\hat{q}_{t+1}\|_2\bigr),$$

with small margin m (e.g., 1-2% of per-joint vel. limit) and gate ρ.

(3) Directional Progress (cosine, joint-space). If a joint-space goal q_g is available, align the first predicted step toward q_g:

$$d_g = q_g - q_t, \qquad \mathcal{L}_{\text{dir}} = 1 - \frac{\langle \Delta\hat{q}_{t+1}, \, d_g \rangle}{\|\Delta\hat{q}_{t+1}\|_2 \, \|d_g\|_2 + \epsilon}.$$

(If q_g is not available, replace d_g with the demo's short-horizon direction $q_{t+h} - q_t$, small h.)

(4) Cheap Smoothness (acceleration L2).

$$\mathcal{L}_{\text{acc}} = \frac{1}{T_{\text{future}} - 1} \sum_{i=2}^{T_{\text{future}}} \|\Delta\hat{q}_{t+i} - \Delta\hat{q}_{t+i-1}\|_2^2.$$

8.4 Architecture

This section situates our design in the landscape of imitation-learning (IL) policies for vision-based control, motivates our choice of a lightweight fully connected (FCL) temporal head, compares major architectural families, and then specifies the proposed *VisionJointPlanner* in detail.

Table 8.5: Architectural families for IL-based, vision-conditioned control (typical online inference characteristics).

Family	Temporal	Pros / Typical use	Cons / Compute
FCL / 1D Conv	Short window; MLP or shallow conv	Lowest latency, tiny footprint; easy quantization; reliable online control (31)	Limited long-horizon context; weaker for strongly multi-modal futures
Transformers	Causal self-attention over vision/joint/goal tokens	Long context; multi-task scaling (RT-1/RT-2, PerAct); flexible goal injection (32; 33; 34; 35)	Higher lat./mem. ($\mathcal{O}(T^2d)$); tuning/data hungry; embedded deployment harder
Diffusion policies	Iterative action denoising	Handles multi-modality; precise rollouts (21)	Multi-step sampling (10–50) increases latency; nontrivial for hard real-time

8.4.1 State of the Art Architectures

Modern IL controllers for visuomotor control cluster into three families. First, **lightweight feedforward** heads fuse short visual histories with proprioception (and optionally a numeric or keypoint goal) and directly regress near-term actions using multilayer perceptrons (MLPs) or temporal 1D convolutions; such designs remain competitive for short horizons and excel when real-time latency is a constraint (31). Second, **Transformer-based sequence models** encode tokens from vision, proprioception, and goal modalities using self-attention, capturing long-range temporal structure and supporting multitask learning at scale (e.g., RT-1/RT-2, Decision Transformer, Perceiver-Actor/PerAct) (32; 33; 34; 35). Third, **diffusion-policy heads** generate action sequences via iterative denoising conditioned on observations and goals, offering strong multi-modality and precision at the cost of iterative inference (21).

When the task intent is best expressed visually, goal images or crops can be injected and aligned to the current observation through cross-correlation (Transporter) or text-conditioned pathways (CLIPort) (36; 37); numeric 3D goals remain standard in control-centric deployments. For datasets lacking explicit goals, goal-conditioned supervised learning with hindsight relabeling (GCSL) provides a principled recipe to retrofit goals from achieved states (38).

8.4.2 Architecture for *IRIS*

Our deployment target is a cinema robot arm executing smooth, high-speed camera motions under tight control deadlines on embedded hardware. The principal constraint is deterministic worst-case latency rather than asymptotic expressivity. An FCL temporal head with fixed input shapes yields a compact compute graph, minimal memory traffic, straightforward quantization/pruning, and stable cache behavior, thereby reducing jitter and bounding execution time. While Transformers and diffusion heads offer longer context and multi-modal competence, their attention scaling and multi-step sampling respectively impose latency and memory costs that are misaligned with on-rig, hard real-time requirements without substantial additional engineering. We therefore adopt an FCL temporal head and keep interfaces (modalities and tensor shapes) compatible with later upgrades should compute budgets expand.

We now describe the proposed goal-conditioned FCL architecture. At time t, the inputs comprise a numeric goal point $g \in \mathbb{R}^3$, a window of five RGB images $\{\mathbf{I}_{t-4}, \ldots, \mathbf{I}_t\}$ with $\mathbf{I}_\tau \in \mathbb{R}^{3\times H\times W}$, and the corresponding joint states

$\{q_{t-4}, \ldots, q_t\}$ with $q_\tau \in \mathbb{R}^6$. The model predicts a five-step future joint sequence $\hat{Q}_{\text{future}} = \{\hat{q}_{t+1}, \ldots, \hat{q}_{t+5}\} \in \mathbb{R}^{5\times 6}$.

Encoders Each image is processed by a ResNet-18 backbone (ImageNet-pretrained; final fully connected layer removed, optionally frozen), producing a 512-dimensional feature:

$$\phi(\mathbf{I}_\tau) \in \mathbb{R}^{512}. \tag{8.1}$$

Joint states and the goal are embedded via small MLPs:

$$\psi(q_\tau) = \text{MLP}_q(q_\tau) \in \mathbb{R}^{64}, \qquad \gamma(g) = \text{MLP}_g(g) \in \mathbb{R}^{32}. \tag{8.2}$$

Per-timestep tokens are formed by concatenation:

$$e_\tau = \left[\ \phi(\mathbf{I}_\tau)\ ;\ \psi(q_\tau)\ ;\ \gamma(g)\ \right] \in \mathbb{R}^{608}. \tag{8.3}$$

Temporal fusion and prediction head We stack tokens over the history $E_t = [e_{t-4}, \ldots, e_t] \in \mathbb{R}^{5\times 608}$, flatten, and pass them through a lightweight temporal MLP with GELU activations and LayerNorm:

$$z_t = \text{MLP}_{\text{temporal}}\big(\text{vec}(E_t)\big) \in \mathbb{R}^h. \tag{8.4}$$

A final MLP maps z_t to a fixed-size output that is reshaped into the future joint sequence (absolute or deltas):

$$\hat{Q}_{\text{future}} = \text{reshape}\Big(\text{MLP}_{\text{head}}(z_t),\ 5,\ 6\Big). \tag{8.5}$$

Fixed input shapes enable operator fusion in compiled runtimes, reducing latency and jitter at deployment.

8.5 Trainings

In the beginning, to validate the network's effectiveness and if it is able to output viable path for the robot arm, a small dataset is used to train the model. The dataset only consist of 25 expert demonstrations, and is only been trained over 50 epoches. The training results can be seen in Figure 8-7

The initial results from the small dataset conclude the following findings:

- The training is able to converge
- The output path-plan is feasible

However, when examining the generated path plans down to individual joint movements, some joint angle outputs are erratic and non-periodic. As illustrated in Figure 8-8, joint angles 4 to 6 significantly deviate from the ground

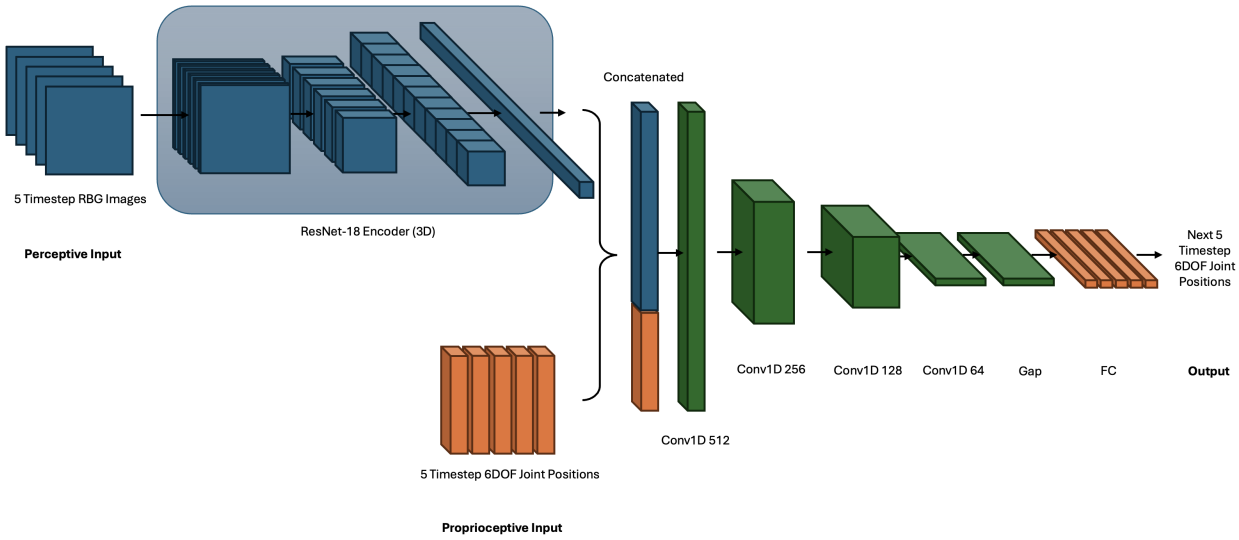


Figure 8-6: VisionJointPlanner: compact 3D ResNet-18 encoder → 512-D visual feature, concatenated with 6-D joints; a temporal window is processed by three Conv1D layers, pooled over time, then mapped by an FC head to $\hat{Q}_{\text{future}}$.

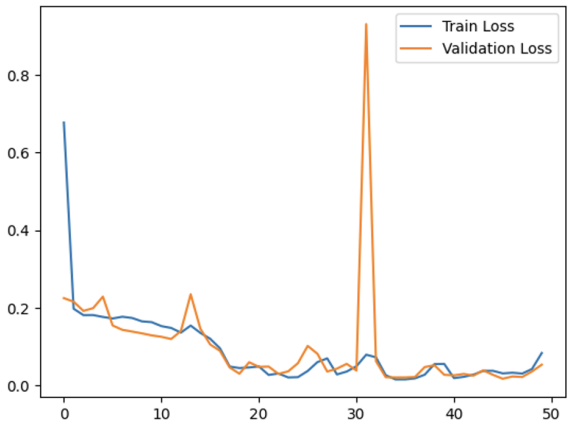


Figure 8-7: First policy training results

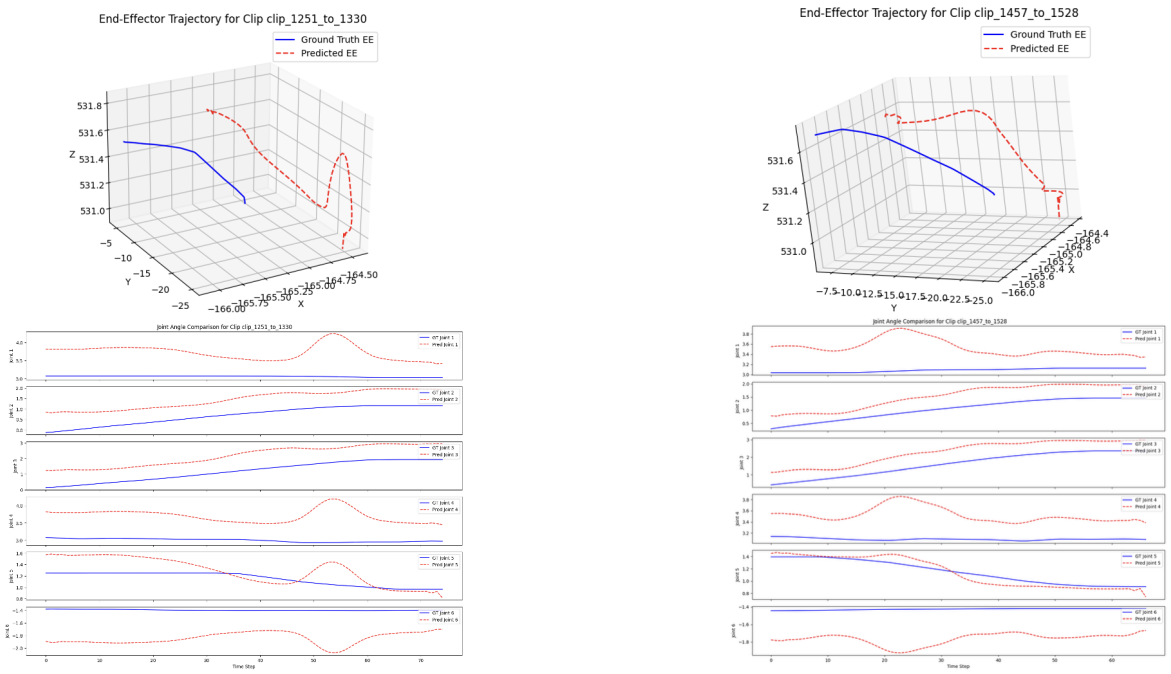


Figure 8-8: First policy tests

truth. This discrepancy likely arises from insufficient ground-truth data during training, causing the network to lack accurate guidance for moving these joints. To address this issue, additional data was collected with explicit movements for all six joints of the cinema robot arm. This new dataset, termed "policy 2," includes synchronized movements for all joints at each timestep, ensuring correct joint correspondences. Such comprehensive reference data is critical for imitation learning in path planning, as there are theoretically infinite joint configurations to transition between points A and B. Providing expert demonstrations for all six joints helps the network converge toward a consistent and accurate path-planning strategy. The training and testing results trained from dataset policy 2 can be seen in Figure 8-9 and Figure 8-10.

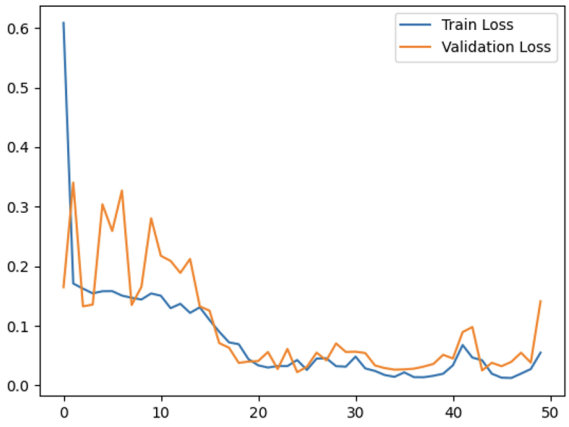


Figure 8-9: Second policy training results

Finally, to overcome the derivation of the autonomous path-planning overtime from policy 2, the data set had been scaled to over 17897 samples for training, 1224 samples for validation, and 950 samples for testings. After adding

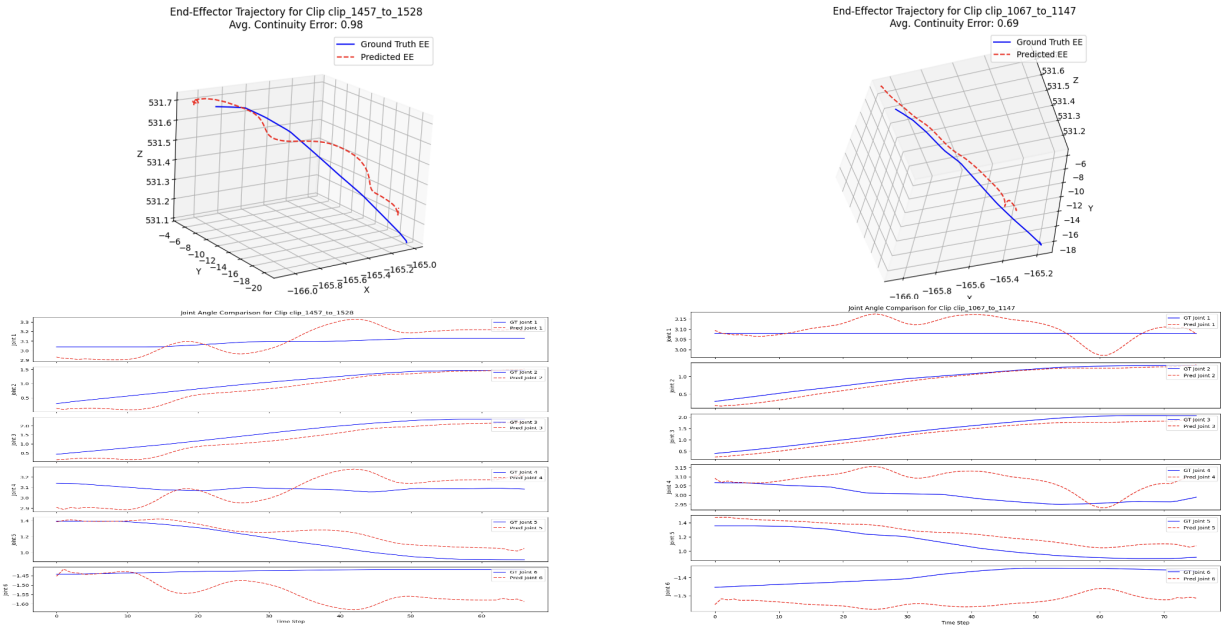


Figure 8-10: Second policy tests

the additional dataset, the training results can be seen from Figure 8-11, where the dataset has been trained over 100 epoch, doubling the number of iterations than before as well. The figure also suggests a clear convergence in the validation data that the policy can generalize well in different path-planning scenarios. Figure 8-12 shows five different testing results of the zero-shot path planning data. Overall, the generated path plans closely match the expert trajectories, with only slight deviations. Importantly, when encountering obstacles, the policy demonstrates strong convergence toward collision-free paths for the cinema robot. The notable improvement observed in policy 2 highlights the effectiveness of the data scaling method in imitation learning, suggesting the potential for even better generalization and obstacle-avoidance capabilities with larger datasets. Furthermore, the enhanced performance of policy 3 can be partially attributed to its scaled-up and diverse dataset, which includes varied environmental backgrounds and distinct path-planning objectives, further improving the model's generalization ability.

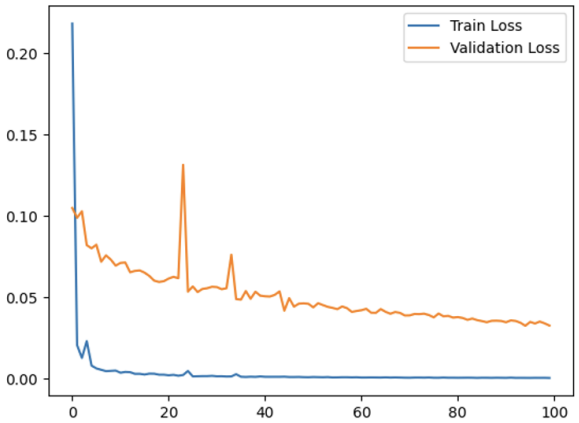


Figure 8-11: Final policy training results

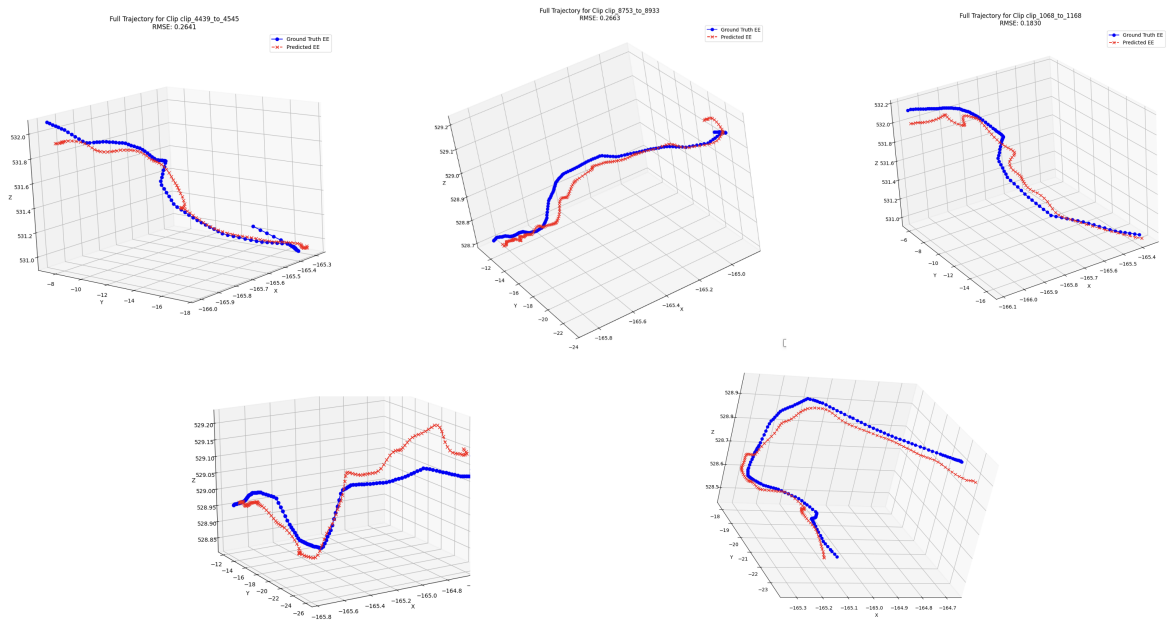


Figure 8-12: Final policy testing results

8.6 Deployment

Once the policy was trained in simulation and validated offline, additional integration and safety measures were implemented to enable deployment onto *IRIS*. The overview of the full stack of *IRIS* can be seen in Figure 7-1. *IRIS* was developed on Ubuntu 20.04 with ROS Noetic, using Python for perception, data pre-processing, and policy inference; and the Unitree SDK for low-level joint control via field-oriented control (FOC) commands.

8.6.1 System Initialization

At startup, *IRIS* is driven to a pre-defined "Home" configuration for each joint at a steady low speed. The "Home" position starts with *IRIS* resting with the shoulder joint pointing downwards, and the elbow joint positioned at 180°. This initialization minimizes the chance of initial collision. Hardware-level safety limits are configured through the FOC interface, including:

- Torque, velocity, and position limits for each joint.
- Joint temperature thresholds to prevent overheating.
- Self-collision avoidance parameters.

These constraints are enforced in parallel to policy execution at the low-level in the SDK controller level, ensuring that any violation triggers an immediate halt.

8.6.2 Policy Inference

During runtime, synchronized RGB frames from the Intel RealSense camera and proprioceptive joint states are fed into the perception pipeline. The RGB images undergo light pre-processing before being passed to the trained imitation-learning policy, which outputs the next joint position or velocity commands. These commands are streamed to the Unitree SDK at the desired control rate at 30Hz.

8.6.3 Execution Monitoring and Logging

Given the trained policy, additional integration steps are required to deploy it in the real world. The runtime system is implemented on Ubuntu 20.04 with ROS Noetic, using Python for perception and policy inference and the Unitree SDK for low-level control. At startup, the robot is initialized in a safe home configuration, and actuator limits for torque, velocity, position, and temperature are set via the FOC interface. A goal-directed start routine moves the arm to a seed configuration near the task goal to avoid zero-action attractors. During execution, the policy receives synchronized RGB frames from the end-effector camera and current joint states, predicts the next joint position or velocity commands, and sends them through the Unitree SDK at the control rate. Commands are filtered and rate-limited to ensure cinematic smoothness and prevent mechanical stress. Safety monitors - including joint limit checks, torque/temperature thresholds, self-collision detection, and depth-based obstacle proximity - are enforced at every cycle, with any violation triggering an immediate halt. Progress toward the goal is continuously tracked, and if stagnation is detected, an IK-based recovery motion blended with policy output is applied. All telemetry, including states, actions, safety events, and progress metrics, is logged for post-deployment analysis. A more detailed pseudocode for the deployment of the IL pipeline can be seen in algorithm 2.

8.6.4 Challenges and Strategies

Zero-Action Attractor

When *IRIS* was initialized in the beginning, the policy produced near-zero joint updates for several consecutive cycles, resulting in stagnation. This *zero-action attractor* phenomenon arises from the policyâĂŹs bias toward previously observed static windows during training. In addition, the goal point is never specified as an input in the training loop, resulting the model confused in the direction it should be moving. Two complementary strategies were implemented:

- A **goal-directed start routine**, computed via resolved-rate inverse kinematics (IK), moves the arm toward an intermediate seed configuration close to the task goal. This ensures the policyâĂŹs input sequence begins with non-zero motion.
- A **stagnation detector** monitors progress toward the goal; if displacement remains below a threshold for N consecutive steps, a short IK-generated motion is blended with the policyâĂŹs output before control reverts to the learned policy.

Algorithm 2: IL Policy Execution with Goal-Directed Start (ROS Noetic + Unitree SDK/FOC)

Data: Camera frames I_t, joint state $(\mathbf{q}_t, \dot{\mathbf{q}}_t)$, goal pose $\mathcal{G}$ (EE pose or waypoint set)
Result: Safe real-time control commands to Unitree FOC actuators
Initialize:
Start ROS core (Ubuntu 20.04);
Launch Python node (policy + perception);
Connect Unitree SDK; Load trained policy π_θ;
Home robot $\rightarrow \mathbf{q}_{\text{home}}$; set filters (low-pass & rate limiters);
Define stuck detector: window W, thresholds $\epsilon_{\text{act}}, \epsilon_{\text{prog}}$;

Goal-Directed Start (anti-zero-action):
Compute short IK path to a *goal-biased seed* $\mathbf{q}_{\text{seed}}$ near $\mathcal{G}$ (within joint limits);
Execute a time-limited (T_seed) minimum-jerk jog: $\mathbf{q}_{t+1} \leftarrow \text{Interp}(\mathbf{q}_t, \mathbf{q}_{\text{seed}})$ with saturation;
If safety triggers (collision, over-torque, over-temp) $\Rightarrow$ halt and alert;

Main Control Loop (at f_c Hz):
Acquire synchronized $(I_t, \mathbf{q}_t, \dot{\mathbf{q}}_t)$;
Form input window $\mathcal{X}_t = \{(I_{t-k}, \mathbf{q}_{t-k})\}_{k=0}^{T_{\text{past}}-1}$;
$\hat{\Delta\mathbf{q}}_{t:t+H} \leftarrow \pi_\theta(\mathcal{X}_t)$; // Predict future joint deltas
Select first-step command: $\Delta\mathbf{q}_t \leftarrow \text{LPF}(\hat{\Delta\mathbf{q}}_t)$; apply rate/accel limits;
(Optional) add tiny exploration nudge near home: $\Delta\mathbf{q}_t \leftarrow \Delta\mathbf{q}_t + \mathcal{N}(0, \sigma^2)$ for $t <$ T_nudge;
Send command to Unitree SDK (position or velocity setpoint); enable FOC tracking;

Safety, Progress, and Recovery:
Monitor: joint limits, torque/temp, depth-based proximity, self-collision;
Update progress metric P_t (e.g., EE distance to $\mathcal{G}$ or waypoint index);
if $\|\Delta\mathbf{q}_{t-k:t}\|_\infty < \epsilon_{act}$ ***or*** $\Delta P_{t-k:t} < \epsilon_{prog}$ *for* $k \in W$ **then**
  // Policy stall / zero-action attractor
  Execute **Goal Nudge**: short IK push toward next waypoint with PD assist;
  Blend command: $\mathbf{u}_t \leftarrow \alpha\, \mathbf{u}_t^{\text{IK}} + (1-\alpha)\, \mathbf{u}_t^{\pi}$, $\alpha \in [0.2, 0.5]$;
end
if *hard safety trip* **then**
  Stop and hold (gravity-safe posture) $\rightarrow$ wait for operator;
end

Termination:
If $\mathcal{G}$ reached (within tolerance) or user stop: decelerate with minimum-jerk to hold pose;
Log telemetry for analysis (states, actions, safety flags, progress);

Depth Sensor Noise

Raw depth data exhibited temporal flicker and spatial holes, particularly around reflective or thin objects. Although the policy could tolerate minor noise, larger artifacts occasionally caused spurious obstacle detections. To mitigate this:

- Intel's built-in hole-filling and temporal smoothing filters were applied online.
- A spatial median filter over a fixed region of interest was added to reject isolated outliers.

Note: The final policy omits depth input because the depth sensor on *IRIS* has a near-field blind zone (no reliable returns below ~ 30 cm), which is common in close-range shots. Instead, RGB input with temporal context (e.g., stacked frames or optical flow) allows the model to recover depth cues from parallax, motion, occlusions, and size priors, and proved more reliable for this application.

Path-planning without obstacle Path-planning with obstacle

Figure 8-13: Two testings done in the real world: one with the obstacle, another without obstacles

8.7 Discussion and Future Work

IL was initially selected because it enables online obstacle avoidance of both static and dynamic obstacles and exhibits strong generalization to novel scenes. This is essential for a consumer-oriented system. *IRIS* should operate safely and autonomously without requiring users to understand robot operation. IL also leverages expert demonstrations to reproduce cinematographic moves, allowing *IRIS* to execute shots that align with professional practice.

Empirically, while IL generalized well to zero-shot environments and across platforms, the trade-off in safety and predictability can be non-trivial for cinema use. During testing on *IRIS*, learned policies exhibited higher jerk and stepwise actuation compared with continuous manual teleoperationâĂŤan effect that becomes noticeable when pixel-level framing is required. These observations motivate a controller that retains IL's perception-driven responsiveness while imposing tighter smoothness and safety structure.

A practical direction is a hybrid stack in which an IL policy proposes perception-informed intent, a fast trajectory layer enforces cinematic smoothness and actuator constraints (39), and a thin safety supervisor guarantees constraint satisfaction at runtime (40). Perception-aware control, where the controller directly optimizes objectives tied to visibility, feature tracking, and obstacle margins, has shown robustness gains in related domains (41; 42) and can be adapted to maintain subject framing and line-of-sight for cinematography.

On the learning side, scaling and diversifying data remains a priority. Varied backgrounds and lighting, richer mixtures of static/dynamic obstacles, and multiple task objectives help reduce covariate shift on set. Conditioning policies on user-specified goal poses (or short waypoint sequences) and incorporating standard planning terms (*clearance*, *smoothness/jerk*, *time*) directly into the loss can turn the current 'move-forward' bias into general goal-directed behavior. To further improve motion quality, diffusion-based visuomotor policies are promising drop-in replacements or finetuning targets, often yielding smoother, more stable actions than standard behavior cloning

(21).

Taken together, a perception-informed IL module, a trajectory layer with TOPP-RA-style retiming (39), and a predictive safety filter (40), all shaped by perception-aware objectives (41; 42), could be a potential path to achieve the responsiveness of learning while preserving the predictability, safety, and temporal consistency required for repeatable, pixel-level shots on *IRIS*.

Chapter 9

Validations

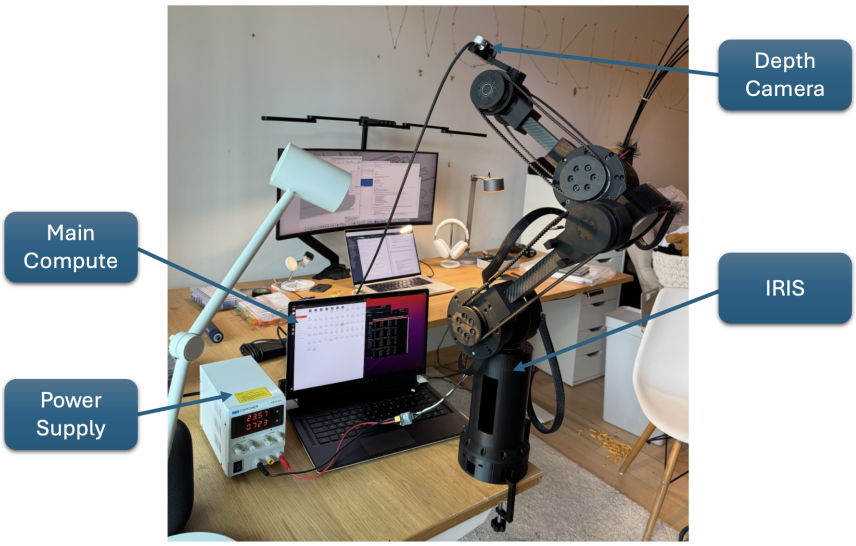


Figure 9-1: Breakdown of the evaluation testing setup.

This chapter presents a set of evaluations conducted to validate both the low-level actuator performance and the high-level cinematic shot quality of *IRIS*. In evaluation, to justify the objectives listed in Chapter 2, we focus on three aspects: actuator tracking and repeatability, trajectory fidelity under dynamic motion, and teach-and-repeat path following. The setup for these validations is shown in Figure 9-1.

9.1 Actuator Performance Evaluation

Cinema applications demand smooth, precise joint tracking. This section verifies that the FOC-driven actuators in *IRIS* can achieve stable and accurate performance.

In this evaluation, joint-level tracking accuracy was evaluated by commanding sinusoidal profiles under impedance control with gravity feedforward. Commanded and measured positions were logged at 1 kHz.

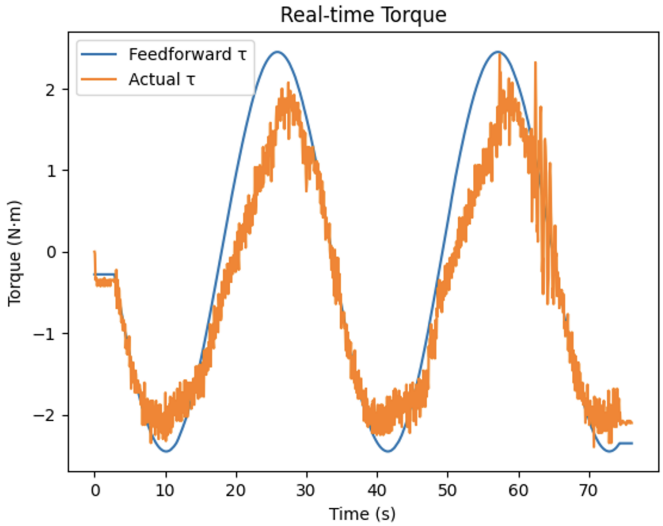


Figure 9-2: Actuator tracking performance: commanded vs. actual position over time for one representative actuator.

Results Figure 9-2 shows close alignment between commanded and measured trajectories, with small transient deviations and near-zero steady-state bias across the tested range. Tracking error increases modestly with payload, attributable to unmodeled link flexibility and joint friction. Overall, the behavior supports the high-accuracy and smooth-motion objectives for *IRIS*. Looking ahead, periodic parameter identification and modest structural stiffening should further reduce these errors; additionally, lightweight strain/flex sensing on critical links could enable closed-loop compensation for compliance during loaded motions.

9.2 Tracking Accuracy

For repeatable cinematic moves, the end-effector of *IRIS* must adhere closely to commanded paths in Cartesian space while remaining dynamically well-behaved in joint space. In this evaluation, *IRIS* executed sinusoidal and trapezoidal trajectories; commanded and measured states were logged at 1 kHz and compared in both frames.

As shown in Figure 9-3 and Figure 9-4, the executed motion closely overlays the references with smooth, low-amplitude residuals. Small phase offsets appear at higher speeds and around curvature changes, yet the deviations remain bounded and visually imperceptible for the tested move profiles. The combined Cartesian and joint-space views indicate that the kinematic mapping and low-level control preserve trajectory fidelity expected for professional camera moves, and that jerk-limited references further promote the desired cinematic smoothness.

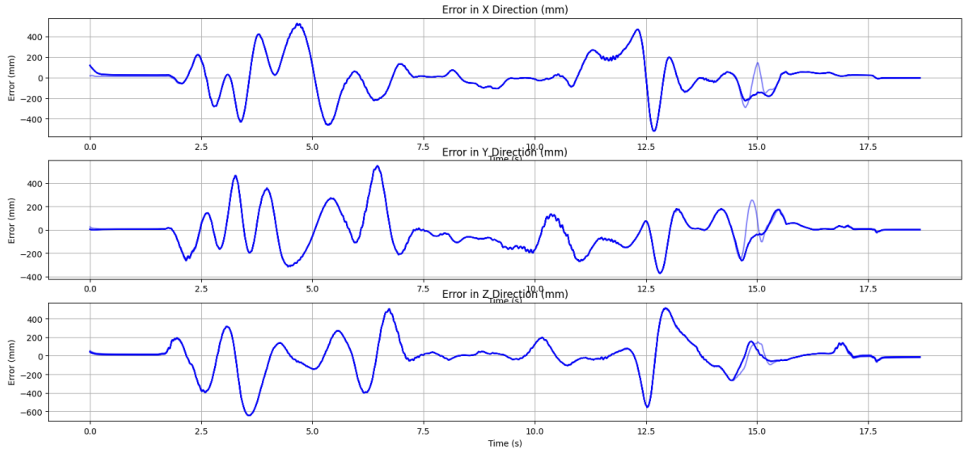


Figure 9-3: End-effector tracking performance for *IRIS*: commanded vs. actual trajectory in Cartesian space.

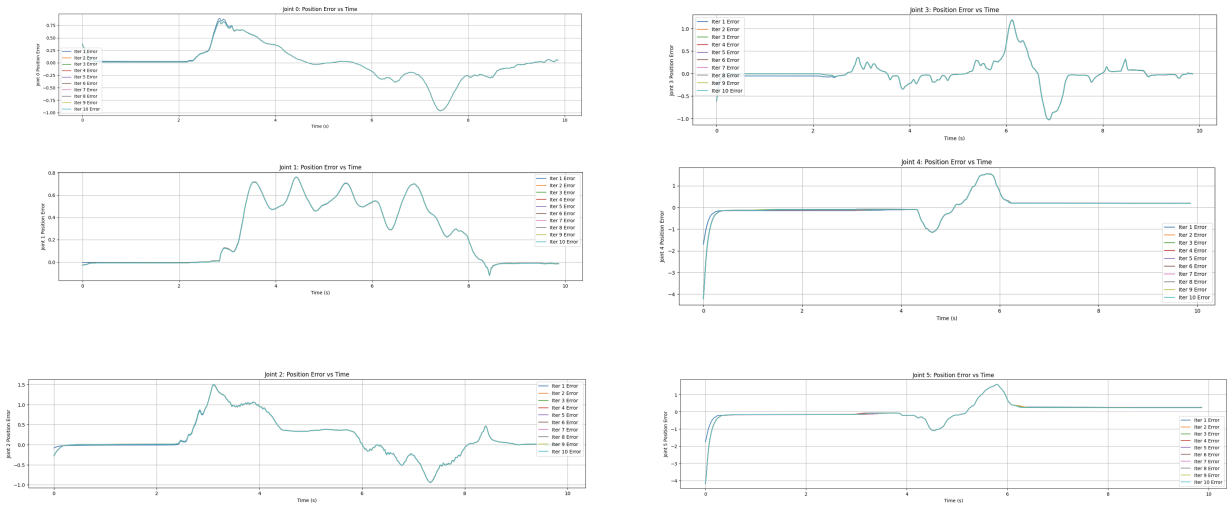


Figure 9-4: Joint-space tracking performance for *IRIS*: commanded vs. actual joint positions.

9.3 Repeatability Test

Consistent framing across takes relies on the ability of *IRIS* to return to the same target configuration from a standardized start. Here, each joint was driven from home to an identical terminal pose across $N = 30$ trials under unloaded and 1.5 kg payload conditions; final encoder readings were recorded and summarized.

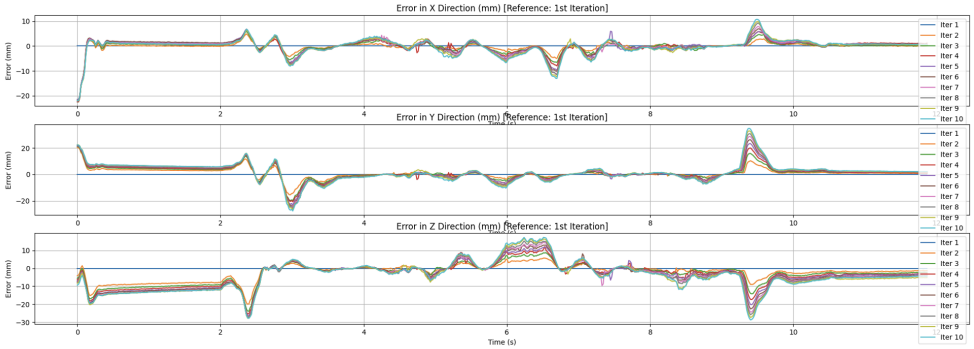


Figure 9-5: Repeatability results for *IRIS*: endpoint dispersion across repeated trials.

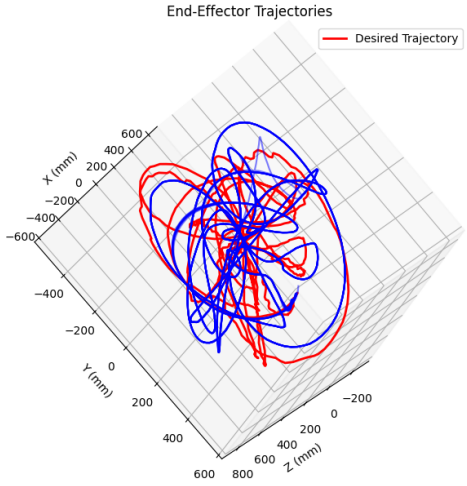


Figure 9-6: Trajectory repeatability for *IRIS*: representative overlays of repeated executions.

The clusters in Figure 9-5-Figure 9-6 remain tight across joints and payload conditions, with dispersion concentrated within a narrow band around the setpoint. Payload introduces a slight broadening of the cluster, consistent with small compliance and friction effects, yet the aggregate behavior supports frame-accurate re-takes and alignment-sensitive workflows such as VFX plate acquisition and motion-matched inserts.

9.4 Path-following Accuracy (Teach & Repeat)

Teach-and-repeat allows an operator to demonstrate a camera move once and have *IRIS* reproduce it reliably. During the *teach* phase, joint states and timestamps were recorded while guiding *IRIS* along the desired path; the *repeat*

phase executed the captured trajectory with time-synchronized playback. Deviations were computed against the taught path in both joint and Cartesian spaces.

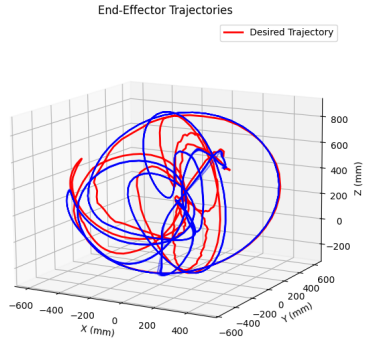


Figure 9-7: Path-following test for *IRIS*: overlay of taught (reference) and repeated trajectories.

The overlay in Figure 9-7 shows strong agreement between the demonstrated and reproduced paths. Errors are smooth and low-frequency, with small phase shifts near sharper turns but no discontinuities, indicating stable timing and consistent execution. In practice during a film production, this level of agreement is sufficient for matching beats to on-set cues and repeating complex blocking; synchronized clocks and minimum-jerk re-timing further improve temporal alignment on longer paths while preserving the cinematic character of the motion.

Chapter 10

Conclusion

10.1 Achievement and Problems Addressed

This project's goal has always been bringing cinema robot arm to the general consumers who love cinema and filming. It meant to build a tool that enable filmmakers and hobbyists to be creative, bold and carry their ideas into fruition.

The design of *IRIS* demonstrated the feasibility of democratizing robotic cinematography through open-source, 3D-printed, and learning-enabled robotic systems. It offers an accessible alternative to high-end rigs and opens up new creative possibilities for independent filmmakers and studios.

On the hardware side, by leveraging off-the-shelf actuator modules and timing-belt-driven transmission, we engineered a 3D-printed robot arm that is low-budget with intuitive control methods. On the software side, instead of using external controller or complex professional software, an iOS application is designed to be carried with the user everywhere since everyone already have a mobile device in their pocket. Any extra devices or complex software can appear redundant and unnecessary. Finally, the project integrated all the mechanical design, firmware, and the high-level control algorithm to make the project as vertically-integrated as possible. This further helps the project to bring down the cost, and accessibility to the masses.

10.2 Reflections and Limitations

In the beginning, the project aim to address the gap between the expensive hardware cinema robot arm faces, the bridge that gap to general consumers by reducing the cost. However, this is not without the challenges and limitations. In a nutshell, to bring any product to the consumer market, we realized that there are a lot more to be considered and polished to be truly consumer ready. One of the biggest drawback is the industrial design of the robot. Consumers care about the aesthetics of the product besides their functionalities, which can often be overlooked in the industry. This objective of focusing on the design of the robot was not fully considered during the design phase,

thus making the final robot design too industrial instead of looking friendly. Moreover, despite the addition of the mobile application, there are still a lot of functions required to be added before the product is ready to be shipped to the consumers. Current applications only addresses simple teleoperations and the teach and repeat using the built-in IMU on the mobile device. More complex controls for the robot arm, including object tracking, key-frame, speed controls, and face tracking would still need to be added to be a more complete product. Thirdly, although using addictive manufacturing can drastically reduce the cost of the product, this greatly sacrifices the accuracy of the cinema arm. Finally, to achieve full vertical integration, actuators still need to be designed in-house. In addition, the current design uses the same actuator in all joint, which is not optimal as each joint's torque requirement differs from each other. For example, the actuators used at the wrist joint can be reduced to smaller-torqued actuators to further minimize the cost.

10.3 Future Outlooks

The current design sets the ground work of what a budget cinema robot could be - a low-cost design that is tailored for cinema purposes, with an intuitive user-interface. However, more work and iterations are required for the future and making it consumer ready.

- **Housing design**: current cinema arm do not have a housing to hide away the timing-belt and the electronics, making it daunting to an average consumer. In addition, without a casing, the design look more like a student's capstone project than a matured product that is ready for the market. A new industrial design for the cinmea robot that fits the tastes for filmmakers and photographers could be essential to the product's success and appeals.
- **More mobile App functions**: As mentioned in the prior dissicion, additional functions like object tracking, pre-planned paths, gesture controls, and face recoginition controls could be handy when it comes to user controls. Like the DJI consumer drone technology, where they have the functions embedded to the drone itself and one single polished application on the mobile device, the average consumer may find it a lot less intimidating to use than a complex control software for the professionals.
- **Actuator design**: The current actuators are using off-the-shelf Unitree motors. This is suitable for the initial prototype, but not for long-term mass production. Designing the in-house motor, gearbox, and motor control is critical for long-term success. On top of that, an electrically controlled stopping mechanism should be added for the safety concerns when the robot arm is off power.
- **Integration**: Further integrations are still required. The robot arm is small and ideally be portable, the base still lacks a powerful enough battery pack to power all six actuators. In addition, a suitable microcontroller is needed to process the on-board controls of the cinema robot, while keeping the cost down.
- **Materials**: Finally, instead of using PLA for the printed material, some carbon-fiber enforced materials should be explored to further incease the stiffiness of the parts and increase accuracy.

Chapter 11

Future Work

11.1 Bimanual *IRIS* (Under Development)

Figure 11-1: Real-life build of the bimanual *IRIS*

To demonstrate the scalability of the design, a bimanual cinema system was prototyped by mounting two identical *IRIS* units on a common base, each controlled independently but time-synchronized; an overview is shown in Figure 11-2. This configuration opens creative workflows that a single arm cannot realize. Here are some ways I have imagined a bimanual cinema system could achieve:

- **Dual-camera choreography.** One *IRIS* carries the *A*-camera for the hero shot while the second runs a syn-

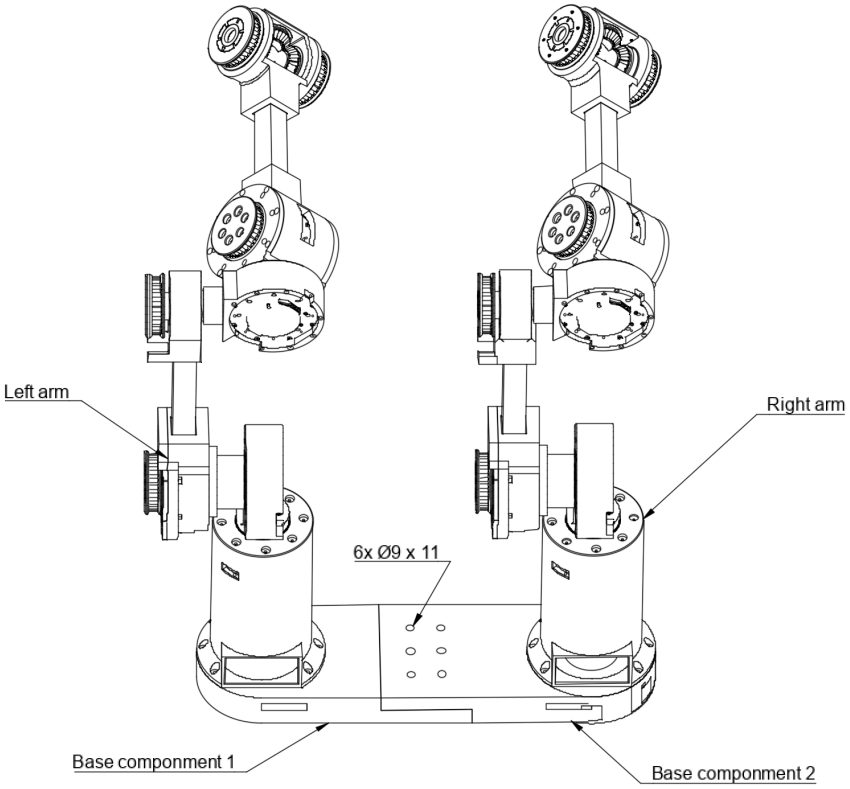


Figure 11-2: Concept sketches of a bimanual *IRIS* rig. Independent controllers synchronize via a shared world frame and timeline.

chronized *B*-camera at an offset angle or focal length, enabling simultaneous wide/close coverage with matched motion.

- **Active lighting partner.** One *IRIS* flies a key/fill source (e.g., LED panel or reflector) while the other carries the camera; light position and orientation are co-animated with framing for consistent catchlights and motivated moves.
- **Dynamic background screen.** One *IRIS* holds a high-brightness display/LED tile as a live background for product shots, while the second *IRIS* carries the camera; the on-screen content updates in real time based on the camera *IRIS*'s pose and focal length to maintain parallax- and perspective-correct imagery.

11.2 iOS App for Teleoperation (Under Development)

To improve on-set usability, an iOS application was built to teleoperate *IRIS* without a tethered gamepad or laptop. The app aims to communicate over Bluetooth for local testing (and Wi-Fi for longer range) with a lightweight server on the *IRIS* control computer (Ubuntu 20.04), bridging to ROS Noetic.

Three input modes are supported:

1. **Individual Joint Control.** Sliders or jog buttons command joint angles directly for setup and calibration. A live 3D model mirrors *IRIS* in real time.

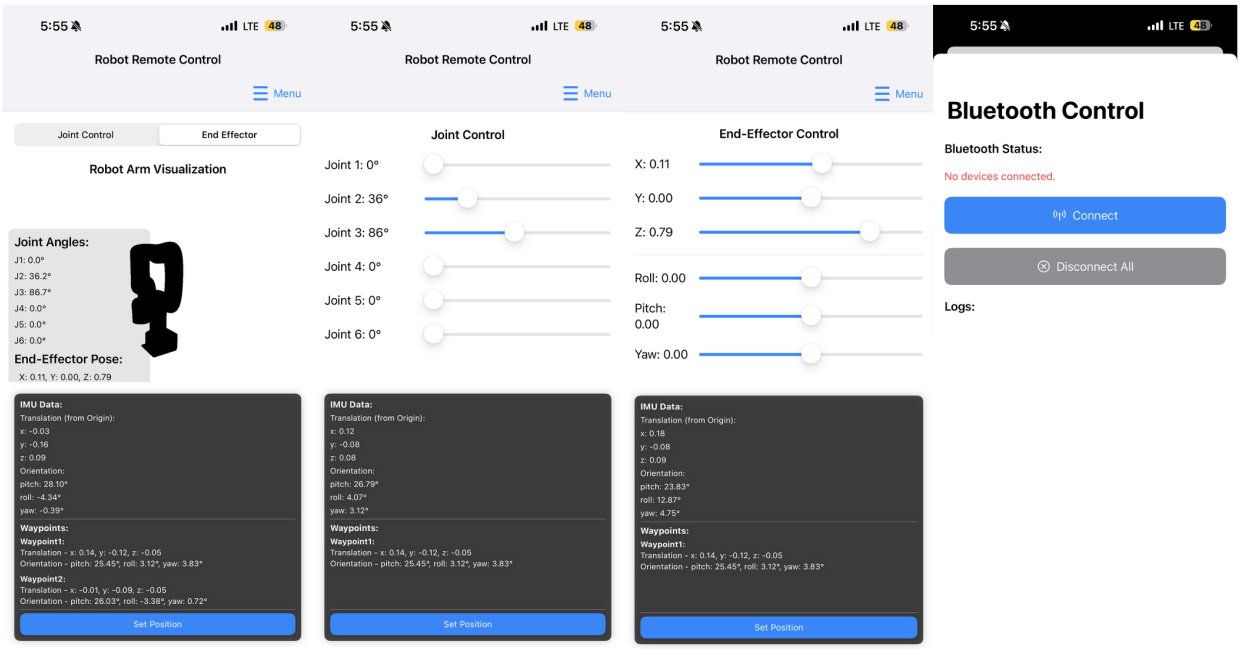


Figure 11-3: iOS teleoperation app for *IRIS*: joint sliders, Cartesian pose input, device-IMU control, and live model preview.

2. **End-effector Pose Control.** Cartesian targets (position & orientation) are sent to the backend, which solves IK (chapter 5) and streams joint setpoints to *IRIS*. This mode is suited for precise framing.

3. **Device-IMU Pose Control.** The phone acts as a 'virtual handle': fused pose (e.g., VIO/IMU) defines the desired end-effector pose. A gain and workspace scale (e.g., 2:1) map hand motion to safe, precise robot motion.

For handheld control, the operator simply moves the mobile device in space; the phone's fused pose (VIO/IMU) is mapped to the end-effector pose of *IRIS*. To keep motion cinematic, the input stream is low-pass filtered and constrained by velocity clamps, which remove high-frequency jitter and prevent abrupt jumps. A press-and-hold 'clutch' enables motion only while depressed; releasing it freezes the target so the operator can reposition without affecting the robot. The app also provides shot utilities, record/stop for teach-and-repeat, save/load of shot files, and a 'safety bubble' slider to temporarily expand keep-out zones during rehearsals. A representative screen is shown in Figure 11-3.

Overall, the vision of the mobile app composes as follows:

- **Core functions:** log current pose; command individual joints; IK-based Cartesian control; live 3D visualization; teach-and-repeat record/playback.
- **Teleoperation options:** Bluetooth (local), Wi-Fi/UDP (stage distance); device-IMU mode; optional CV modules (e.g., person/prop tracking) that bias Cartesian targets.
- **Future directions:** on-device path sketching (draw-to-trajectory); AR overlay for safety bubbles and reachable set; shot lists with takes/versioning; multi-*IRIS* synchronization controls; operator presets (speed/accel/jerk caps); and a calibration wizard to align phone and end-effector frames.

Chapter 12

Acknowledgement

Finally, I would like to use this chapter to express my sincere gratitude to my two outstanding supervisors, Professor Matthew Mackay and Professor Ali Bereyhi, for their unwavering support and guidance throughout this personal project. Their encouragement, insights, and mentorship have been instrumental in bringing this vision to life. I hope this project serves not only as a testament to the potential of cinema robotics, but also as an inspiration for future Master's projects in the field of robotics.

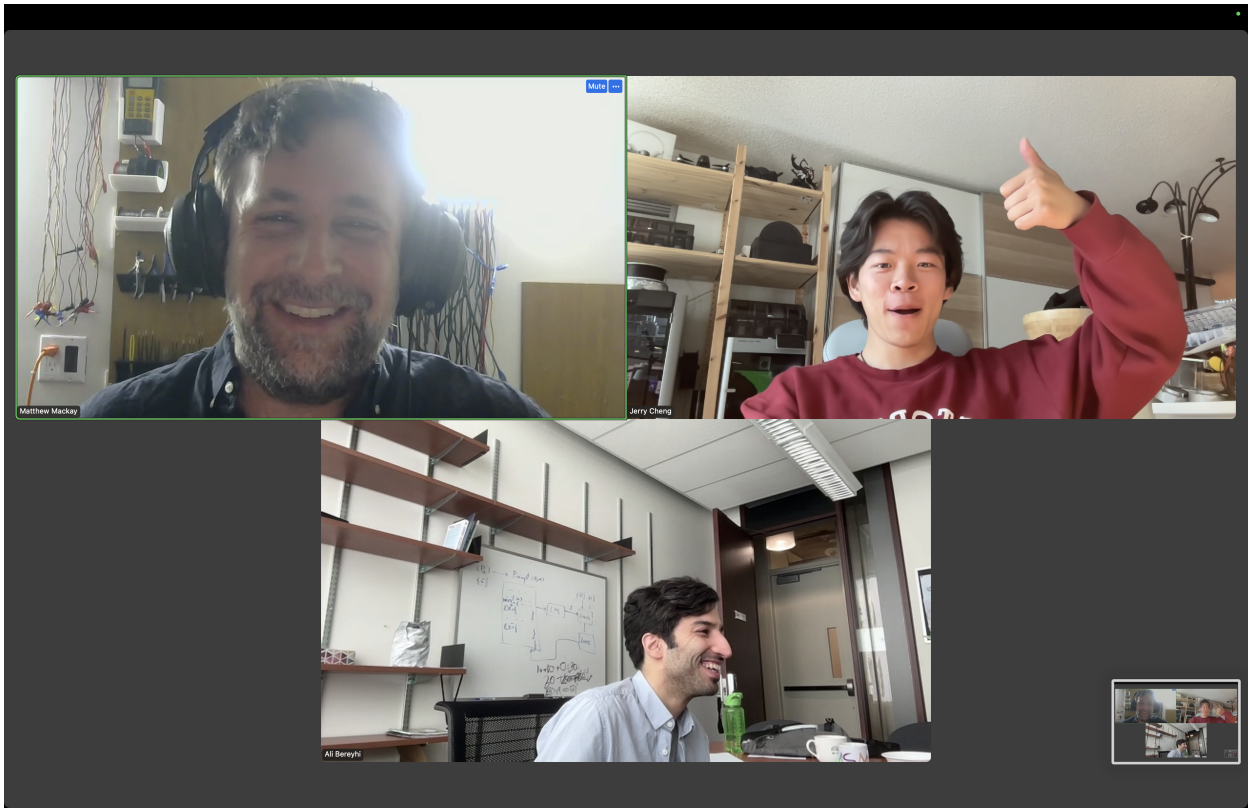


Figure 12-1: Group photo with both supervisors

Bibliography

[1] J. Smith and L. Wang, "Industrial robots in cinema: Technology and applications," *International Journal of Robotics and Automation*, vol. 15, no. 2, pp. 105–113, 2022.

[2] E. Johnson and D. Lee, "The rise of cinema robots: Driving forces and market trends," *Robotics Today*, vol. 28, no. 5, pp. 305–315, 2023.

[3] ROBOTFACE, "What is Camera Motion Control?" https://www.youtube.com/watch?v=Blotbqv7D9U, 2019, accessed: Jul. 4, 2025.

[4] MRMC. (2024) Bolt high-speed cinebot. Accessed: 2024-09-12. [Online]. Available: https://www.mrmoco.com/bolt/

[5] ——. (2024) Milo motion control rig. Accessed: 2024-09-12. [Online]. Available: https://www.mrmoco.com/milo/

[6] M. Precision. (2024) Kira cinema robot. Accessed: 2024-09-12. [Online]. Available: https://www.motorizedprecision.com/kira

[7] P. Anderson and W. Zhao, "Training and skill requirements for operating cinema robots," *International Journal of Robotics in Media Production*, vol. 8, no. 4, pp. 125–135, 2022.

[8] M. Garcia and A. Tan, "Teach pendants in modern robotics: Industrial and media applications," *IEEE Robotics and Automation Magazine*, vol. 10, no. 1, pp. 85–90, 2023.

[9] Camerabotics. (2024) Crx1100 cinema robot. Accessed: 2024-09-12. [Online]. Available: https://www.camerabotics.com

[10] M. Precision. (2024) Kira cinema robot. Accessed: 2024-09-12. [Online]. Available: https://www.motorizedprecision.com/kira

[11] S. C. Robotics. (2024) C20 cinema robot. Accessed: 2024-09-12. [Online]. Available: https://www.sisucinemarobotics.com

[12] ——. (2024) Sisu lab interface. Accessed: 2024-09-12. [Online]. Available: https://www.sisucinemarobotics.com

[13] B. Katz, N. Paine, and A. Parness, "Digit: A platform for mobile manipulation and dynamic interaction," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*. IEEE, 2023.

[14] D. Kim and S. Kim, "Highly dynamic quadruped locomotion via whole-body control with centroidal momentum," in *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 718–725.

[15] G. Williams, P. Drews, B. Goldfain, J. M. Rehg, and E. A. Theodorou, "Information theoretic model predictive control: Theory and applications to autonomous driving," *arXiv preprint arXiv:1707.02342*, 2017.

[16] M. Bhardwaj, B. Sundaralingam, A. Mousavian, N. Ratliff, D. Fox, F. Ramos, and B. Boots, "Storm: An integrated framework for fast joint-space model-predictive control for reactive manipulation," in *Proceedings of the Conference on Robot Learning (CoRL)*, 2022. [Online]. Available: https://proceedings.mlr.press/v164/bhardwaj22a/bhardwaj22a.pdf

[17] J. Hwangbo, J. Lee, A. Dosovitskiy, D. Bellicoso, V. Tsounis, V. Koltun, and M. Hutter, "Learning agile and dynamic motor skills for legged robots," *Science Robotics*, vol. 4, no. 26, p. eaau5872, 2019.

[18] N. Rudin, D. Hoeller, P. Reist, and M. Hutter, "Learning to walk in minutes using massively parallel deep reinforcement learning," *arXiv preprint arXiv:2109.11978*, 2022, coRL 2021 version.

[19] D. Hafner *et al.*, "Mastering diverse domains through world models," *arXiv preprint arXiv:2301.04104*, 2023.

[20] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, "Control barrier functions: Theory and applications," *arXiv preprint arXiv:1903.11199*, 2019.

[21] C. Chi, Z. Xu, S. Feng, E. Cousineau, Y. Du, B. Burchfiel, R. Tedrake, and S. Song, "Diffusion policy: Visuomotor policy learning via action diffusion," *arXiv preprint arXiv:2303.04137*, 2023. [Online]. Available: https://arxiv.org/abs/2303.04137

[22] Open X-Embodiment Collaboration, "Open x-embodiment: Robotic learning datasets and rt-x models," *arXiv preprint arXiv:2310.08864*, 2023.

[23] O.-X. E. Team, "Open x-embodiment: Robotic learning datasets and rt-x models," 2024, dataset: 1M+ real robot trajectories, 22 embodiments. [Online]. Available: https://robotics-transformer-x.github.io/

[24] A. Khazatsky, K. Pertsch, S. Nair *et al.*, "Droid: A large-scale in-the-wild robot manipulation dataset," in *Robotics: Science and Systems (RSS)*, 2024. [Online]. Available: https://arxiv.org/abs/2403.12945

[25] H. Walke, K. Black, A. Lee *et al.*, "Bridgedata v2: A dataset for robot learning at scale," *arXiv:2308.12952*, 2024. [Online]. Available: https://rail-berkeley.github.io/bridgedata/

[26] S. Dasari, F. Ebert, S. Tian *et al.*, "Robonet: Large-scale multi-robot learning," *arXiv:1910.11215*, 2019. [Online]. Available: https://www.robonet.wiki/

[27] E. Jang, A. Irpan, M. Khansari *et al.*, "Bc-z: Zero-shot task generalization with robotic imitation learning," in *Proceedings of CoRL*, 2022. [Online]. Available: https://arxiv.org/abs/2202.02005

[28] S. James, Z. Ma, D. Arrojo, and A. J. Davison, "Rlbench: The robot learning benchmark & learning environment," in *IEEE RA-L/ICRA*, 2020. [Online]. Available: https://github.com/stepjam/RLBench

[29] (2025) Lite 6 Collaborative Robot Arm – Specification Sheet. UFactory. Accessed: 1-Jun-2025. [Online]. Available: https://www.ufactory.us/product/lite-6

[30] (2025) Intel® RealSense™ Depth Camera D415 – Datasheet. Intel. Accessed: 1-Jun-2025. [Online]. Available: https://www.intelrealsense.com/depth-camera-d415/

[31] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-end training of deep visuomotor policies," *Journal of Machine Learning Research*, vol. 17, no. 39, pp. 1–40, 2016. [Online]. Available: https://jmlr.org/papers/v17/15-522.html

[32] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar *et al.*, "RT-1: Robotics transformer for real-world control at scale," *arXiv preprint arXiv:2212.06817*, 2022. [Online]. Available: https://arxiv.org/abs/2212.06817

[33] ——, "RT-2: Vision-language-action models transfer web knowledge to robotic control," *arXiv preprint arXiv:2307.15818*, 2023. [Online]. Available: https://arxiv.org/abs/2307.15818

[34] L. Chen, K. Lu, A. Rajeswaran, K. Lee, A. Grover, M. Laskin, P. Abbeel, A. Srinivas, and I. Mordatch, "Decision transformer: Reinforcement learning via sequence modeling," *arXiv preprint arXiv:2106.01345*, 2021. [Online]. Available: https://arxiv.org/abs/2106.01345

[35] M. Shridhar, L. Manuelli, and D. Fox, "Perceiver-actor: A multi-task transformer for robotic manipulation," in *Proceedings of The 6th Conference on Robot Learning (CoRL)*, 2023. [Online]. Available: https://proceedings.mlr.press/v205/shridhar23a.html

[36] A. Zeng, P. Florence, J. Tompson, S. Welker, J. Chien, M. Attarian, T. Armstrong, I. Krasin, D. Duong, A. Wahid, V. Sindhwani, and J. Lee, "Transporter networks: Rearranging the visual world for robotic manipulation," in *Proceedings of the Conference on Robot Learning (CoRL)*, ser. Proceedings of Machine Learning Research, vol. 155, 2021. [Online]. Available: https://proceedings.mlr.press/v155/zeng21a.html

[37] M. Shridhar, L. Manuelli, and D. Fox, "CLIPort: What and where pathways for robotic manipulation," *arXiv preprint arXiv:2109.12098*, 2021. [Online]. Available: https://arxiv.org/abs/2109.12098

[38] D. Ghosh, A. Gupta, J. Fu, A. Reddy, C. Devin, B. Eysenbach, and S. Levine, "Learning to reach goals via iterated supervised learning," in *International Conference on Learning Representations (ICLR)*, 2021. [Online]. Available: https://openreview.net/forum?id=rALA0Xo6yNJ

[39] H. Pham and Q.-C. Pham, "A new approach to time-optimal path parameterization based on reachability analysis," *IEEE Transactions on Robotics*, vol. 34, no. 3, pp. 645–659, 2018.

[40] K. P. Wabersich and M. N. Zeilinger, "A predictive safety filter for learning-based control of constrained nonlinear dynamical systems," *Automatica*, vol. 129, p. 109597, 2021.

[41] D. Falanga, P. Foehn, P. Lu, and D. Scaramuzza, "Pampc: Perception-aware model predictive control for quadrotors," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2018, pp. 1–8.

[42] T. Miki, J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning robust perceptive locomotion for quadrupedal robots in the wild," *Science Robotics*, vol. 7, no. 66, p. eabk2822, 2022.