Augmented Reality and Robot assisted Total Knee Arthroplasty

Team D: Final Report

Daksh Adhar
Wentao Cao
Sreeharsha Paruchuri
Parth Singh
Zihan Qu

Sponsors: Smith and Nephew

December 10th, 2025

Abstract

Total Knee Arthroplasty (TKA) demands high precision in bone preparation and implant positioning to ensure long-term surgical success. Traditional approaches often suffer from limited intraoperative visibility, imprecise manual pin placement, and a lack of real-time adaptability to anatomical variations. Our system, **Bone.P.A.R.T.E. (Bone Precision Augmented Reality Tracking Equipment)**, addresses these limitations by integrating augmented reality (AR) and robot-assisted manipulation to enhance surgical planning and execution.

The system overlays pre-operative surgical plans directly onto a patient's anatomy using the Apple Vision Pro AR headset and delegates the screwing of surgical pins to a KUKA LBR Med 7 robotic manipulator. Real-time perception information is provided by an Intel RealSense D405 depth camera, which is used to identify drill sites for AR visualization and guides the robot during pin insertion and auto-reposition. A custom-designed end-effector enables precise pin insertion and supports quick manual pin reloading.

The system was validated using synthetic models of the femur and tibia, achieving an average **pin placement accuracy of** 2 **mm** and a **pin orientation accuracy of** 2°. Key functionalities, such as AR-guided surgical visualizations, auto-repositioning of the arm for improved registration accuracy, autonomous screwing and detaching of pins, and complete control of the system through Apple Vision Pro, were successfully demonstrated in the final demonstration.

Contents

1 Project Description

Total Knee Arthroplasty (TKA) is an orthopedic procedure performed in cases of severe arthritis, where accuracy in bone preparation and implant positioning is critical for long-term patient outcomes. Traditionally, TKA is done manually, with surgeons relying heavily on intuition for bone cuts, often resulting in reduced accuracy and poorer long-term results for the patient. Existing TKA automation systems although can improve accuracy, face other limitations, including restricted surgeon visibility due to external camera setups, reliance on infrared trackers that increase procedural invasiveness, and limited visual detail for the surgeons since the primary display remains a flat screen.

Our solution, **Bone.P.A.R.T.E (Bone Precision Augmented Reality Tracking Equipment)**, addresses these challenges by integrating Augmented Reality (AR) with robot-assisted manipulation. AR provides spatial visualizations that restore the surgeon's depth perception and situational awareness directly at the operative field, while the robotic arm automates accuracy-critical steps of the procedure. Together, they reduce dependency on external trackers, improve intraoperative precision, and enhance long-term patient outcomes.

A key point to note is that our system is semi-autonomous, executing only the accuracy-critical steps of the procedure while leaving clinical decision-making to the surgeon. Figure 1 outlines the standard workflow of a Total Knee Arthroplasty and highlights the stages that are automated or augmented within our system.

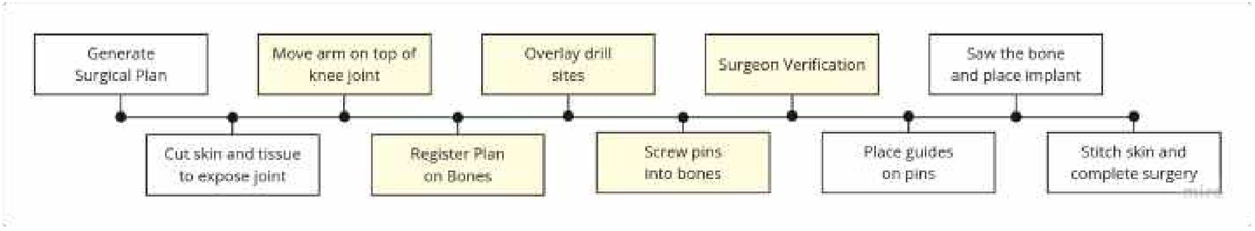

Generate Surgical Plan
Cut skin and tissue to expose joint
Move arm on top of knee joint
Register Plan on Bones
Overlay drill sites
Screw pins into bones
Surgeon Verification
Place guides on pins
Saw the bone and place implant
Stitch skin and complete surgery

Figure 1: Total Knee arthroplasty timeline; yellow highlighting system roles; white highlighting surgeon roles

To achieve these, our platform integrates five tightly coupled subsystems: Perception, Augmented Reality, Planning, Hardware, and Control, for which the system architecture is detailed in Sections 4 and 5, with specific implementation status outlined in Section 6.

This project is sponsored by Smith+Nephew's Research, Technology, and Innovation Centre and serves as a proof of concept for evaluating the viability of integrating AR technology into the broader surgical pipeline.

2 Use Case

Dr. Napoleon (he/him), an experienced orthopedic surgeon, is preparing to perform a total knee replacement surgery for a patient with severe arthritis. Before the operation, using the patient-specific bone model derived from Computational Tomograph (CT) , he generates a pre-operative surgical plan that details the desired location and pose of the implant, as depicted in Figure 2. THis plan is then used to back-calculate the position and orientation of the surgical pins on the bone. (Surgical pins are temporary fixation hardware used to locate and stabilize the surgical guide on the bone.)

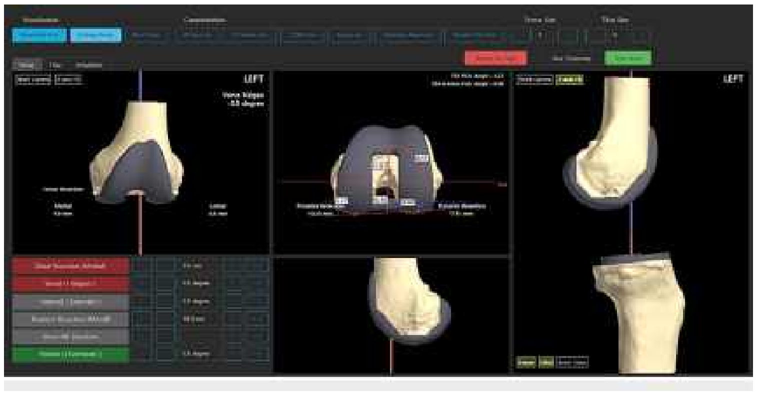


Figure 2: Preoperative surgical plan for total knee arthroplasty surgery, source: [8]

The procedure begins with the patient lying supine, the affected knee fully flexed to provide optimal access, and the robot in its ’away’ position as depicted in figure 3. He then prepares the operative field by carefully dissecting through the skin and subcutaneous tissue to expose the knee joint. With the joint partially exposed, Dr. Napoleon puts on the AR headset and launches the `BONEParte` app. The `robot home` command then moves the robot from its away position to its home position, after which the `proceed mission` command (a) aligns the robot so it is centered over the exposed joint (using the auto-reposition feature) and (b) opens the annotation window in the UI for bone annotations. Once annotation is complete, the system generates a 3D point cloud of the surgical workspace, which is used to register the pre-operative surgical plan with the patient’s anatomy.

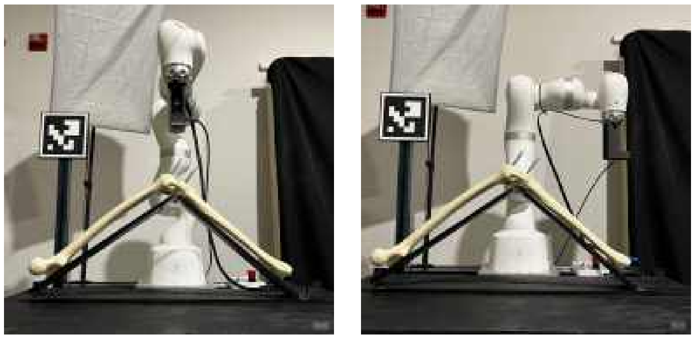


Figure 3: Left: Robot in its ‘home‘ position; Right: Robot in its ‘away‘ position

Dr. Napoleon then switches the app to overlay mode, which displays the 3D visualization of the surgical plan - specifically, the target drilling points where the robot is expected to insert pins along with their drilling axes. If satisfied, he proceeds with the mission and selects one of the four drill poses. The robot then moves precisely to the selected pose, inserts a surgical pin by screwing it into the bone, and returns to its home position, leaving the pin seated in the bone.

After each pin is drilled, Dr. Napoleon can move the robot to its away position and freely reposition the patient's leg to inspect the insertion. Once satisfied, he reloads the end-effector with another surgical pin, brings the robot back to its home position through the AR interface, and continues the procedure in a similar fashion until all pins are drilled.

Once done, the doctor can slide the surgical guides onto the pins, make cuts in the bone, and carry out the remaining steps of the surgery. Throughout the procedure, Dr. Napoleon can immediately stop all system activity using any of the following: (a) the physical emergency stop button on the control pendant, (b) the physical emergency stop button on the electrical subsystem, or (c) the virtual emergency stop button in the AR interface.

3 System Level Requirements

3.1 Functional & Performance Requirements

Our needs analysis led to a set of System-Level Requirements developed through multiple discussions with key stakeholders, including our sponsor Smith+Nephew, Prof. John, former Knee Surgery patients and clinicians we engaged via email and LinkedIn. This initial set was refined by considering available resources, personal learning objectives, and project timelines. The resulting functional and performance requirements are summarized in Table 1.

Functional Requirement	Performance Requirement	Justification
M.F.1: Sense and Segment Bone through manipulator camera	**M.P.1:** Capture the bone's point cloud with a density of 0.5 point/mm^3	Derived based on Paradocs' performance in FVD/encore
M.F.2: Register surgical plan onto bone surface	**M.P.2:** Perform registration with manipulator camera with an error of less than 2.0$\pm$0.5mm	State of the art surgical systems have $\leq$ 2mm registration accuracy
M.F.3: Auto-reposition arm to maintain a centered top-down view of knee joint	**M.P.3:** Compensate for knee joint misalignment up to 8 cm horizontally and 12 cm vertically	Values are based on the camera's field-of-view constraints.
M.F.4: Visualize Drill sites as AR overlay on patient's bone	**M.P.4:** Display the drill sites with a positional error of less than 1cm	Derived by the ability of AR to anchor objects in world space
M.F.5: Provide Surgeon UI for control inputs and visual aids	**M.P.5:** Updates at 4fps to update the surgeon with "real-time" patient info	Based on general understanding of surgeon requirements
M.F.6: Screw Surgical pins at the 4 bone drill sites	**M.P.6:** Screw with a centre-to-centre error $\leq$ 2mm	Knee replacement surgeries have a required accuracy of 2-4mm
M.F.7: Update obstacle map to include newly placed pin after every screw procedure	**M.P.7:** Add newly placed pin to the obstacle map within 0.5 s of screw completion	Required to prevent robot–pin collisions
M.F.8: Allow manual loading/unloading of surgical pins in the end effector	**M.P.8:** The doctor should be able to swap out the pin in 5 seconds	Reduce time of operation

M.F.9: Allow pin to detach from the end-effector after being screwed	**M.P.9:** The end-effector shall release the pin reliably with a failure rate of less than 1%.	Necessary to avoid repeated manipulation of the implant site
M.F.10: Enable control full system contrl through the Apple Vision Pro.	**M.P.10:** Be able to communicate over TCP < 500ms	Reduce cognitive and operational burden of managing multiple devices
M.F.11: Provide both a physical and virtual Emergency Stop	**M.P.11:** Halt all motions within 100ms (physical) and 250ms (AR) in the event of an emergency	Competitor systems have similar quantification

Table 1: Functional and Performance Requirements

3.2 Non-Functional Requirements

Our non-functional requirements were defined with careful consideration of the constraints and dynamics of a surgical environment, ensuring harmony in robotic operation with a human in the loop.

N.R.1: The system will provide a simple, easy-to-understand interface.
N.R.2: The system will minimize cognitive load by displaying only essential information during surgery.
N.R.3: The system will have a low latency AR sub-system to allow for real-time visualization.
N.R.4: The system will allow the doctor to place the robot arm at a designated initial position.
N.R.5: The system will be designed to enable quick setup in the operating room.
N.R.6: The system will require minimal training for surgeons to operate effectively.
N.R.7: The system will be ergonomic, ensuring comfortable use during surgery.
N.R.8: The system will ensure all its components are easy to sterilize.
N.R.9: The system will ensure the AR components have sufficient battery life for uninterrupted use during surgery.
N.R.10: The system will follow all relevant ISO standards for medical robotic systems.

4 Functional Architecture - NOT UPDATED

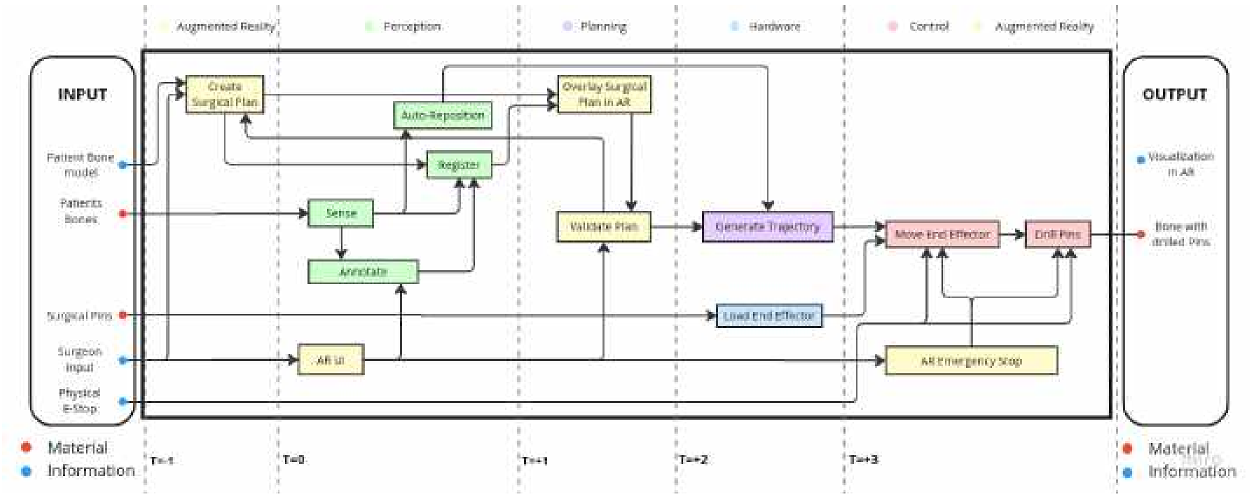


Figure 4: Functional Architecture

The functional architecture outlines the high-level functions that our system performs, as well as the flow of information from inputs through various functional blocks to the system outputs. This architecture is illustrated in Figure 4. The inputs to our system can be broadly categorized into two types: **Material** and **Information**.

- **Material inputs** include the patient's bone and surgical pins. Since we are working in a simulated setting, we substitute the patient's bone with medical-grade synthetic bones sourced from Sawbones [1]. The surgical pins used in the procedure were provided by our industry sponsor, Smith & Nephew.

- **Information inputs** include:

  - A **pre-operative surgical plan**, which we approximate based on our study of standard Total Knee Arthroplasty (TKA) procedures,
  - **3D scans of the bone**, captured using photogrammetry-based mobile applications, and
  - **Surgeon input**, which is simulated by a team member during the demonstration to reflect realistic intraoperative interactions.

The system also produces outputs in both material and informational domains.

- The **material output** includes the placement of surgical pins screwed into the synthetic bone.

- The **informational output** is the core focus of our system: providing accurate and intuitive 3D visualizations to assist the surgeon in intra-operative decision making.

The functional blocks within the system are organized temporally, corresponding to specific time stances from $T = 0$ (system initiation) to $T = 4$ (post-drilling phase). Each function is triggered and executed within a defined time window, ensuring synchronization between material handling and data processing components.

4.1 T=0 $\rightarrow$ T=1

The system begins by using the manipulator camera to sense the environment and generate a point cloud. It then uses the auto-reposition feature to move the arm so that the camera can provide a centered top-down view of the exposed joint. It then uses annotations provdied by the surgeon to segment the bone from the sensed data, and register the pre-operative surgical plan onto the live point cloud.

4.2 T=1 $\rightarrow$ T=2

In this phase, the augmented reality (AR) headset enables the surgeon to overlay the surgical plan directly onto the patient's actual bone, offering a more intuitive and immersive visualization. This is achieved from the transformation obtained from manipulator camera to AR headset, thus providing the same resolution as the arm camera. The surgeon can use this to validate the plan and go back to creating a new one if anything feels off.

4.3 T=2 $\rightarrow$ T=3

Once the plan is finalized, it is sent to the off-board computer, which computes a safe and feasible trajectory for the robotic manipulator to drill the surgical pins at the specified locations. The surgeon or assistant is responsible to identify and remove any potential obstacles that might be in the path of the arm, thereby ensuring procedural safety within the environment.

4.4 T=3 $\rightarrow$ T=4

In the final phase, the trajectory is executed by the manipulator, which guides the end effector to drill the pins into the bone. During the homing process, the surgeon or assistant loads surgical pins into the end effector as required. Prior to each drilling procedure, re-registration needs to be done, allowing for physiological movements after each drill, while preserving accuracy.

5 Cyberphysical Architecture

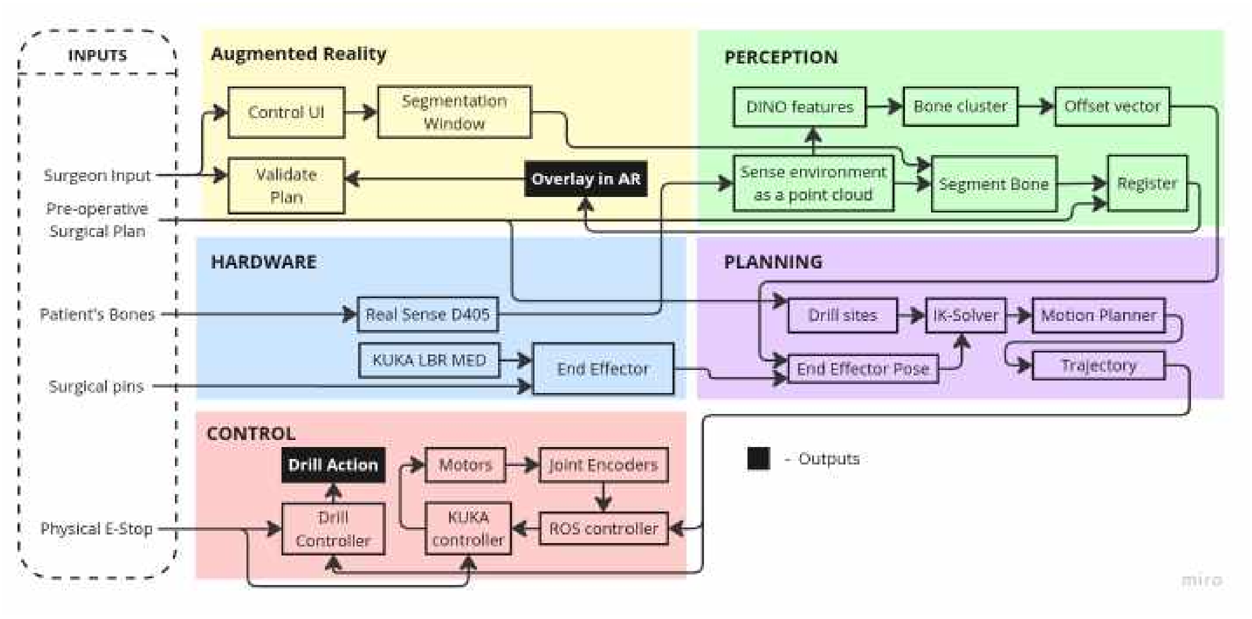


Figure 5: Cyberpyhsical Architecture

The cyber-physical architecture of our system is designed to reflect the sequential execution of functions outlined in the functional architecture. The architecture is composed of five core subsystems: **Augmented Reality**, **Perception**, **Hardware**, **Planning**, and **Control**. These subsystems correspond to the main stages of the surgical workflow and are supported by a structured organization of five main modules, along with an initial input block. Each module elaborates on a specific functional block from the functional architecture, breaking it down into the computational and physical components required for execution. The complete cyber-physical architecture is shown in Figure 5.

5.1 Augmented Reality

The Augmented Reality (AR) subsystem acts as the central command center for the surgeon, bridging the gap between the digital surgical plan and the physical patient. Following a comprehensive trade study, we selected the **Apple Vision Pro (AVP)** over the HoloLens 2 and Magic Leap 2. The AVP was chosen for its superior onboard compute power, mature development ecosystem (RealityKit/Swift), and ability to run low-latency machine learning models for object tracking.

The AR subsystem is responsible for the following functions:

1. **Calibration & Overlay:** To align the virtual surgical plan with the headset, the AR system performs a one-time calibration routine using ArUco markers. This establishes a stable transformation matrix from the Robot End Effector frame to the AVP World frame. Utilizing the Data Node (Port 5001), the headset then receives the surgical plan and overlays 3D drill sites directly onto the patient's bone. Allowing the surgeon to verify the calculated sites and prevent mishaps.

2. **Safety & Control:** The AR interface provides full control over the system's state machine. Crucially, it implements a virtual Emergency Stop connected to the high-priority Command Node (Port 5000), providing a redundant safety layer alongside the physical hardware stop.

5.2 Perception

The perception module is responsible for acquiring and processing spatial information about the surgical environment. The primary sensing device used is the Intel Realsense D405 [2], selected for its compact dimensions (42 mm $\times$ 42 mm $\times$ 23 mm) and lightweight form factor (60 g). Team Paradocs has built the perception stack around this camera using the `realsense-ros` package, which interfaces with the underlying `librealsense2` library. The camera provides a high-resolution point cloud of the surgical workspace, which is subsequently processed to segment the bone, reconstruct its 3D mesh, and register it to the pre-operative surgical model. All processing, including neural feature extraction, is done on images saved to the PC or communicated through ROS messages.

Unlike systems deployed within controlled or static environments, our platform must operate under non-deterministic conditions where lighting, occlusions, and workspace geometry may vary unpredictably. Consequently, the perception stack must not only identify when the environment is conducive to safe robotic operation but also recognize when conditions become inimical to execution. In such cases, it must trigger an appropriate error-recovery protocol to restore the system to a deterministic or stable state before continuing operation.

The two information inputs given to our perception subsystem are:

- **Pre-Operative Bone Model**: This is an STL file representing the femur or tibia bone that will be used for registration. The model may either be a scan of the actual patient's bone, closely matching the physical setup, or a generalized representation based on an average human anatomy.
- **Pre-Operative Surgical Plan**: This plan defines the target position, orientation, and depth for each hole to be drilled. It is provided as a text file containing these coordinates relative to the bone model's frame. Once the bone model is successfully registered, the drill targets can be determined in global coordinates. Both the bone model and the surgical plan are pre-computed and remain fixed during the procedure.

5.3 Hardware

The hardware subsystem comprises several off-the-shelf components and one customized element. The primary components include the sensors onboard the Apple Vision Pro, a camera mounted on the end-effector, the KUKA robotic arm, and a custom-designed end effector. While the camera, robotic arm, and sensors are used as-is, significant development and iterative prototyping were required for the end effector.

The custom end effector was necessary due to two critical functional requirements. First, it must allow the surgeon or assistant to reload surgical pins, ensuring seamless transition between drilling operations. Second, it must incorporate a mechanism to securely release and detach the pin once the drilling is completed. These requirements were addressed through the design of a custom gear system and the integration of a housing unit for an off-the-shelf motor. The resulting design supports both reloading and automated detachment, enabling precise and efficient operation during surgery.

5.4 Planning

The perception subsystem identifies the surgical drill site poses in the camera coordinate frame. The planning subsystem is responsible for transforming these poses into the robot base frame and generating corresponding motion plans for the manipulator to accurately reach and drill at these locations. This transformation relies on inputs from both the hardware (specifically the TF tree describing relative transforms) and the perception subsystem (which provides the detected drill site coordinates). The output of this is a trajectory which is sent to the control stack for manipulator motion.

5.5 Control

The control subsystem is responsible for executing the planned trajectories on the robotic arm with high precision. It interfaces directly with the KUKA controller to send joint-level or Cartesian commands, depending on the operation mode. During execution, the control stack ensures smooth motion, enforces velocity and torque limits, and monitors for deviations from the intended path. Most of this is handled by open source repositories and the control box that comes with the manipulator.

6 Current System Status

6.1 Targeted Requirements

For the Fall Validation Demonstration, our goal was to meet the following objectives and verification criteria which we came up with in the first semester while planning for the project. We were able to meet, if not exceed, all the ambitious targets outlined by our performance requirements.

Objective	Verification
Sense and segment the bones as a point cloud through a camera	Successful registration in under **3 seconds**
Automatically compensate for bone movement during the procedure	Within a radius of **12cm** from the ideal position
Visualize drill sites as overlay on patients bone in the Apple Vision Pro	Within a **±1cm** positional drift
Control the entire pipeline through the Apple Vision Pro	Be able to communicate over **TCP < 500ms**
Screw 4 pins at designated surgical sites	Center-to-center error **< 4mm** and Angular error **< 2°**
Pin detaches from end-effector	The pins are screwed into the bone at a depth equal to the length of the threads in **all 4 attempts**
Update environment map after screw procedure	Manipulator rejects repeated waypoints

Provide **emergency-stop** capability through both hardware and Apple Vision Pro	All processes halted in **<1000ms**

Table 2: System Objectives and Corresponding Verification Criteria

6.2 Overall System Depiction

Figure 6 shows the working system in its final state in a simulated surgical workspace setup.

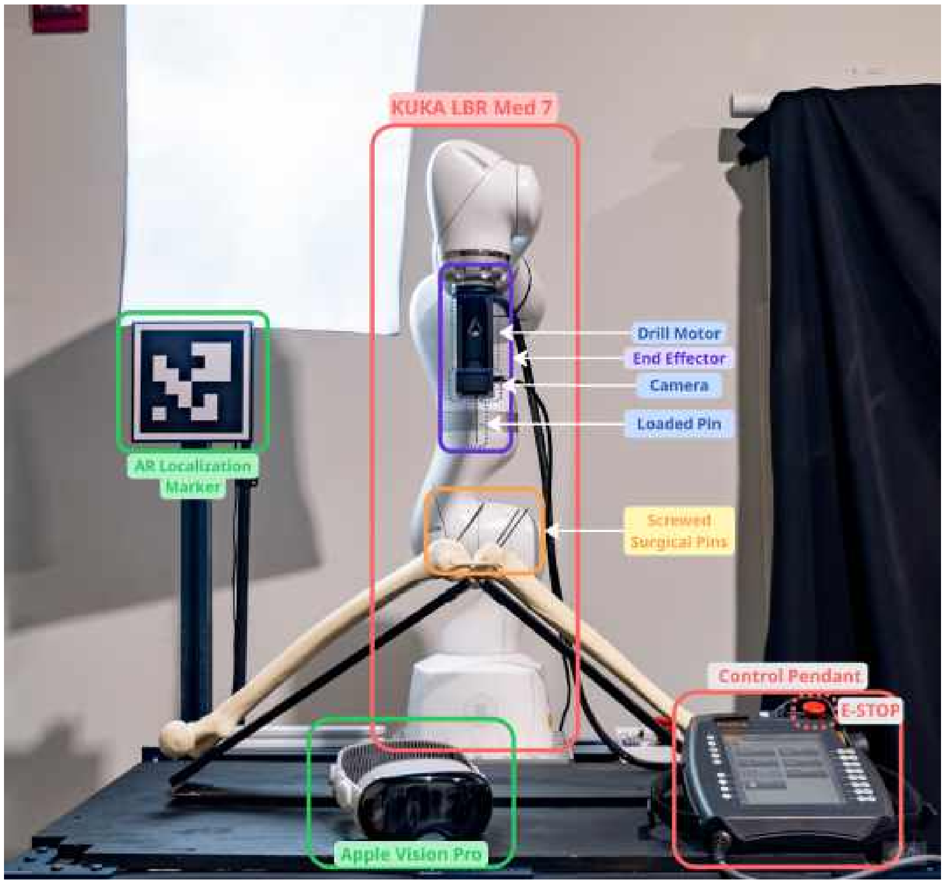


Figure 6: Overall System Depiction

6.3 Subsystem Descriptions

6.3.1 Perception

In our medical robotics application, the visual processing pipeline must provide rich perceptual information to downstream modules while operating in unstructured environments where the robot must detect unsuitable conditions and initiate error recovery. This challenge is compounded by the human-in-the-loop paradigm: the surgeon relies on the same visual feedback as the robot, yet a viewpoint that enables autonomous functionality may not provide the surgeon with sufficient clarity to assess execution safety. To address these requirements, we designed this subsystem to achieve two key objectives: **sub-2mm root mean squared error in 3D-to-3D point cloud registration** and

automatic calibration recovery from erroneous states within 250 milliseconds.

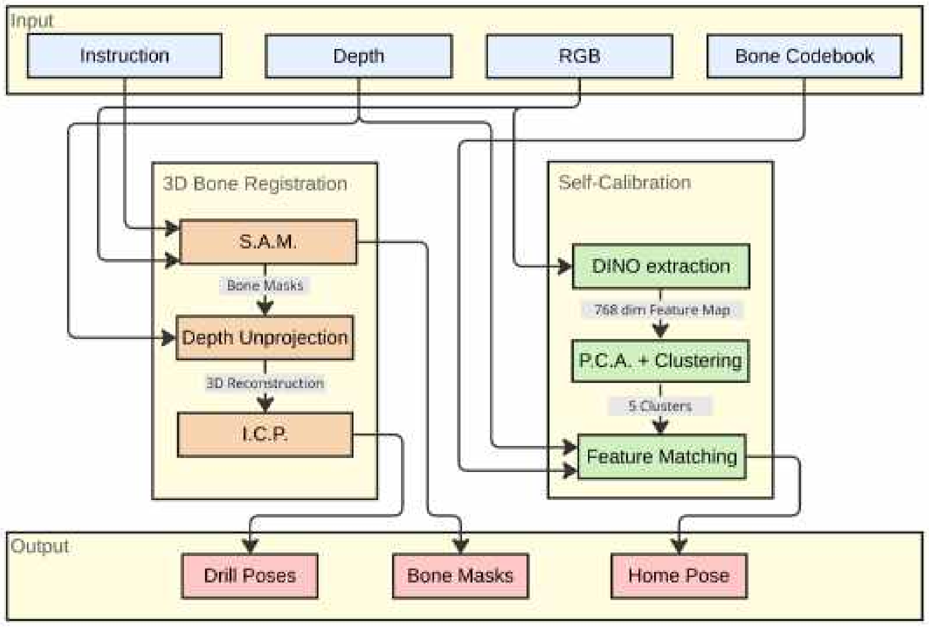


Figure 7: Perception subsystem functional architecture

3D - 3D surgical plan registration

From the pipeline highlighted in Figure 7, the 3D Bone Registration workflow takes as input the surgeon input and a snapshot captured from the frame of the RGB camera placed on the manipulator end-effector and output a resultant set of poses in the robot base (which we set to be our 'world') coordinate frame. This process involves taking an RGB and Depth image input, performing segmentation in the RGB colourspace using Segment Anything v2 [15], performing registration in the 3D pointcloud space [12], [19], [10], camera calibration using the Tsai-Lenz method [9] and a series of coordinate transformations.

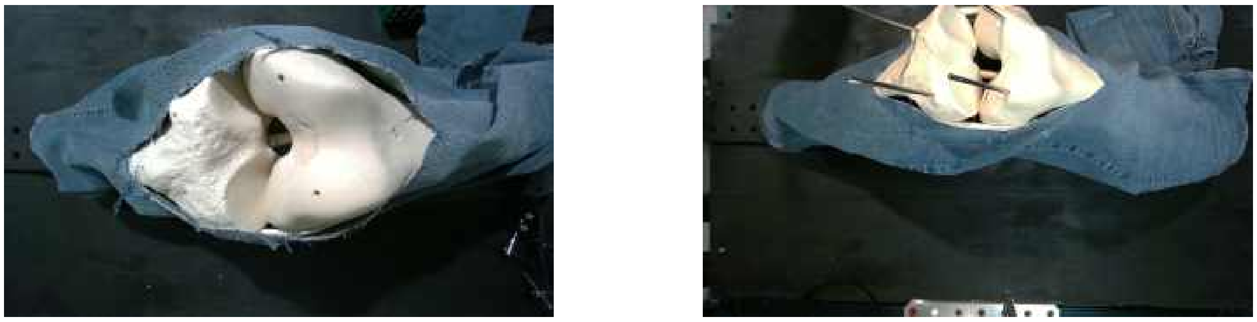


Figure 8: Comparison of viewpoints: optimal bone centering for surgeon annotation (left) versus significantly displaced surgical site (right)

Self-Calibration of Robotic Arm under suboptimal viewpoints

The human-in-the-loop interaction manifests as point prompts to the Segment Anything Model, wherein the surgeon selects points on the femur and tibia via GUI. Optimal annotation requires bone centering in the camera frame; however, residual degrees of freedom in the patient's hip and torso, coupled with intraoperative bone repositioning following pin insertion, induce lateral displacements of several centimeters (Figure 8). Such suboptimal viewpoints impede annotation efficiency, potentially necessitating multiple segmentation attempts that increase procedure duration and cognitive load. We therefore introduce an autonomous repositioning pipeline that moves the robot arm to maintain optimal bone-centered vantage points; the complete perception subsystem architecture is presented in Figure 7.

To achieve the desired functionality, we had to solve 3 key problems: localizing the bone in the RGB space without a user label, generalizing this method to multiple bone geometries and with minimal added latency. To solve the bone localization problem, we had a choice between two popular and strong foundation models, DINOv3 [17] and CLIP [14]. To decide between the two, we evaluated feature maps generated from both models as shown in Figure 9 and Figure 10a.

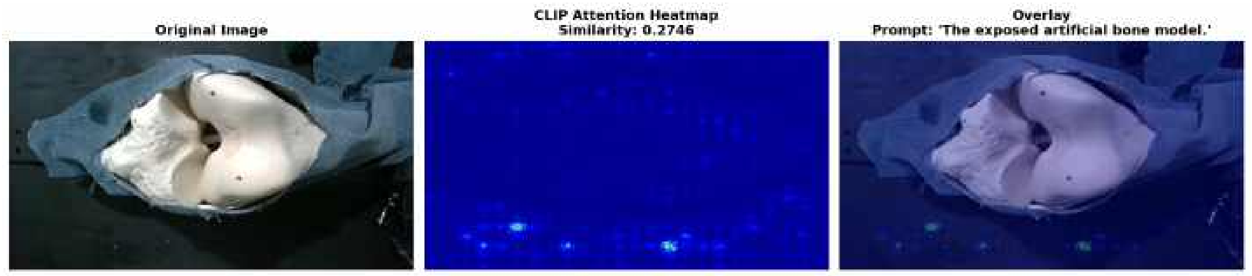
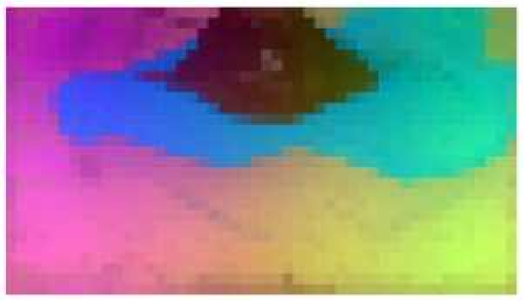
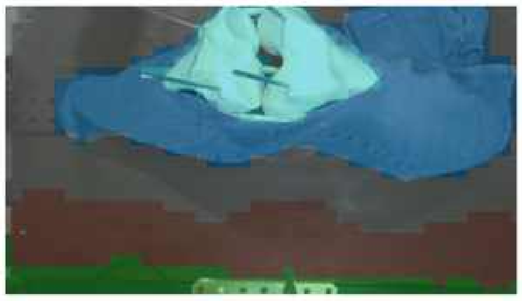


Figure 9: CLIP features on a RealSense image



(a) DINO features on a RealSense image



(b) Clustering on 32-dimensional PCA-reduced DINO feature space

Figure 10: Comparison of DINO feature visualization and clustering results

The comparison revealed that DINOv2 produces significantly more spatially coherent feature representations, which prove essential for unsupervised segmentation via techniques such as `K-Means clustering` could be applied to extract a cluster containing the bone Figure 10. This spatial coherence stems from DINO's self-supervised learning objective, which enforces consistency between different augmented views of the same image through teacher-student distillation. Critically, DINO's loss function encourages local feature smoothness by operating on dense patch-level representations that implicitly encode spatial relationships within the image. In contrast, CLIP optimizes for global image-text alignment through contrastive learning, which prioritizes semantic discriminability over local spatial structure.

Having validated the clustering pipeline, we constructed a codebook to match extracted clusters at test-time and localize the kneecap bone. We curated a dataset of 44 manually labeled images capturing diverse bone orientations and lighting conditions, a subset of which is shown in Figure 11. From these samples, we extracted bone-specific feature clusters to form our codebook. At inference, we compute cosine similarities between all detected clusters and the codebook, selecting the highest-scoring cluster as the bone location. The resulting offset vector is then passed to the planning subsystem for trajectory generation.

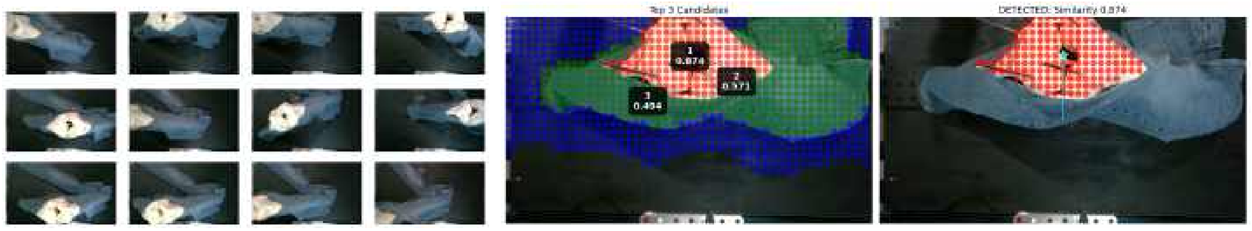


Figure 11: Subset of diverse data samples used to construct the bone feature codebook (left) and Detected bone cluster and corresponding offset vector (right)

6.3.2 AR Subsystem

The Augmented Reality (AR) subsystem provides real-time surgical guidance through the Apple Vision Pro headset, enabling surgeons to visualize drill trajectories and interact with the robotic system through an intuitive mixed-reality interface. The subsystem bridges the perception and manipulation components with immersive 3D visualization overlayed on the physical surgical environment.

The AR subsystem is structured in four distinct layers (Figure 12):

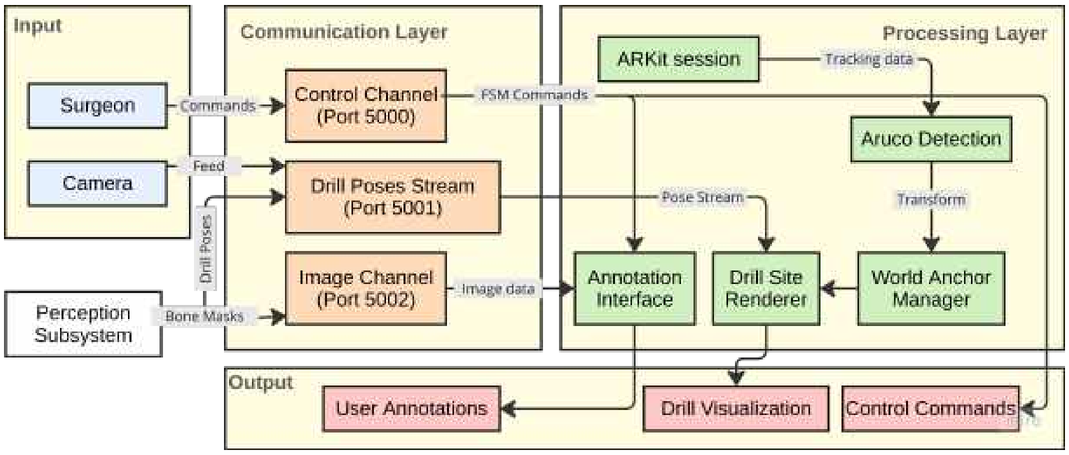


Figure 12: Workflow of the AVP Subsystem

Input Layer: The system receives three primary data streams from the ROS2 backend: (1) RGB camera images from the RealSense D405 for surgeon annotation, (2) segmented bone images from the ParaSight perception pipeline, and (3) real-time drill pose arrays computed by the surgical planner.

TCP Communication Layer: A unified TCP server node manages bidirectional communication between the AVP and the ROS system across three dedicated channels:

- **Port 5000** (Control Channel): Transmits finite state machine commands and control signals between the Vision Pro and ROS2 system from UI shown in figure 13
- **Port 5001** (Drill Poses Stream): Streams drill pose arrays at 10 Hz for real-time visualization
- **Port 5002** (Images Channel): Handles image transfers for annotation and segmentation review

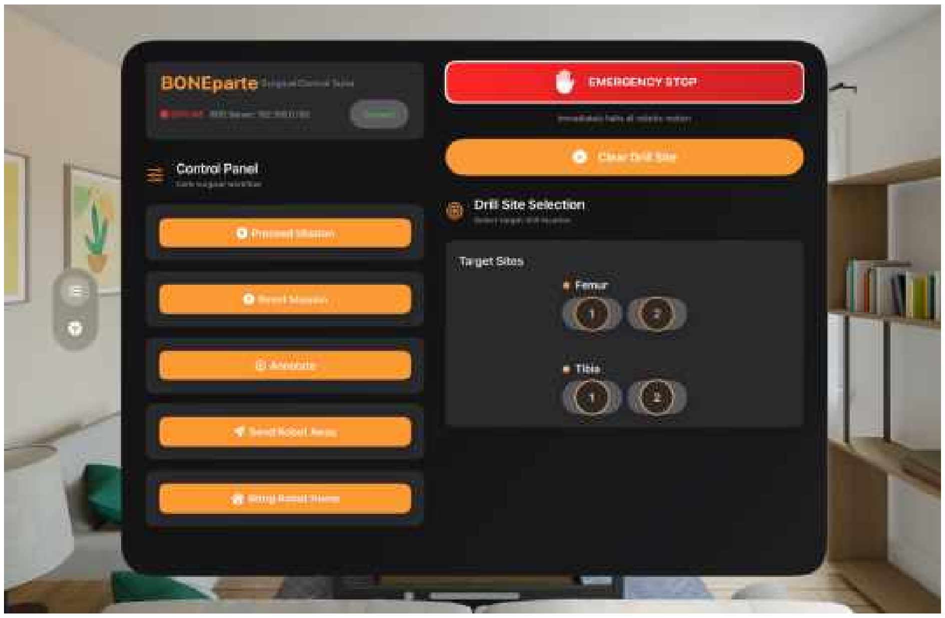


Figure 13: Unified AVP App UI

AR Processing Layer: The core AR functionality is implemented in Swift using the ARKit and RealityKit frameworks, consisting of five key components:

1. **ARKit Session Manager:** Initializes and maintains the AR session with world tracking and image tracking providers, enabling 6-DOF pose estimation and spatial understanding.
2. **ArUco Detection Module:** Performs one-time spatial localization by detecting a 17cm ArUco marker (DICT_6X6_250, ID=0) placed in the surgical environment. The marker serves as a fixed reference point linking the robot base frame to the Vision Pro world coordinate system.
3. **World Anchor Manager:** Establishes and maintains the coordinate transformation chain from the ArUco marker to Vision Pro world space. This persistent anchor enables stable visualization as the surgeon moves around the operating room.

4. **Drill Site Renderer:** Transforms drill poses from the ArUco marker frame to Vision Pro world space and renders them as RealityKit entities. Each drill site is visualized as a red sphere (5mm radius) with three coordinate axes indicating the drilling direction.

5. **Annotation Interface:** Provides an interactive UI for surgeon input during the segmentation workflow. Surgeons tap on displayed RGB images to annotate anatomical landmarks, which are converted to normalized coordinates and transmitted back to the perception system.

Output Layer: The subsystem generates three primary outputs: (1) immersive 3D drill site visualizations rendered in mixed reality as shown in figure 14, (2) user annotations with normalized image coordinates sent to the perception pipeline, and (3) control commands triggering state transitions in the surgical workflow from the UI

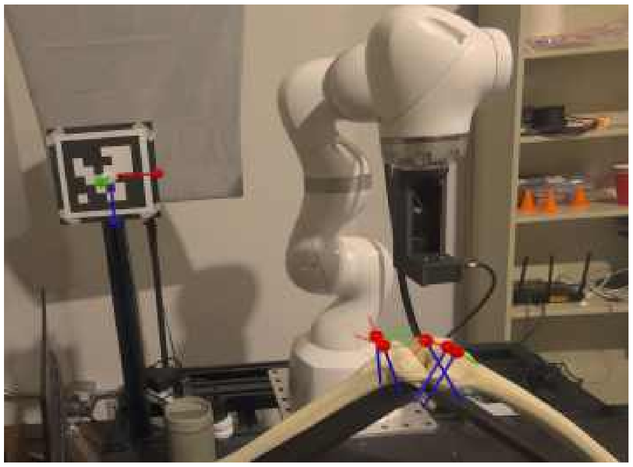


Figure 14: The drill sites being visualized in the AR view.

Spatial Localization and Coordinate Transformation : The AR subsystem employs a multi-stage coordinate transformation pipeline to accurately align virtual drill trajectories with physical anatomy:

$$T_{\text{world}}^{\text{site}} = T_{\text{world}}^{\text{aruco}} \cdot T_{\text{aruco}}^{\text{base}} \cdot T_{\text{base}}^{\text{site}} \tag{1}$$

where $T_{\text{world}}^{\text{site}}$ represents the drill pose in Vision Pro world coordinates, $T_{\text{world}}^{\text{aruco}}$ is the ArUco-to-world transform from ARKit image tracking, $T_{\text{aruco}}^{\text{base}}$ is the static calibrated transform from the ArUco marker to robot base, and $T_{\text{base}}^{\text{site}}$ is the drill pose computed by the perception and planning subsystems.

During surgery, the Vision Pro performs one-time ArUco marker detection using ARKit's `ImageTrackingProvider`. Once detected, the system stores $T_{\text{world}}^{\text{aruco}}$ and relies on ARKit's world tracking to maintain spatial stability, eliminating the need for continuous marker tracking and allowing the marker to be occluded during the procedure.

The AR subsystem is implemented as a native visionOS application written in Swift, utilizing: ARKit, RealityKit and SwiftUI. The ROS2 integration layer consists of Python nodes in the `tcp_server_pkg` package, managing the three-port TCP server architecture and interfacing with the perception and manipulation subsystems through standard ROS2 topics and services.

Beyond drill visualization, the AR subsystem includes experimental bone tracking functionality using ARKit's `ObjectTrackingProvider`. Separate applications (`FemurTracking` and `TibiaTracking`) can track 3D bone models in real-time, providing continuous pose estimation as the patient or surgical site moves. This capability was demonstrated in the Spring Semester but dropped in the Fall semester.

6.3.3 Hardware

The Hardware Subsystem is responsible for physically realizing the surgical plan generated by the surgeon and visualized via AR by executing accurate, stable, and safe drilling operations on a bone model. It interfaces directly with the robotic arm, the perception sensors, and the manual control layer to enable autonomous pin insertion in a clinical-like setting.

At the core is a custom-designed end-effector (see Figure 16) mounted on a KUKA LBR Med 7 manipulator. The end-effector integrates a drill motor, a pinion cum collet mechanism for the surgical pin, and an Intel RealSense D405 depth camera. The drill motor is electrically actuated through a custom PCB and an Arduino R3 microcontroller which regulates power and handles switching logic based on ROS inputs. The pinion cum collet (see Figure 15) was designed for securely holding the surgical pin during screwing while simultaneously having a compliant hold (achieved through a neodymium magnet inside the collet) on the pin to allow for manual loading and automatic detachment of the pin. More details on the subsystem components are mentioned in table 3.

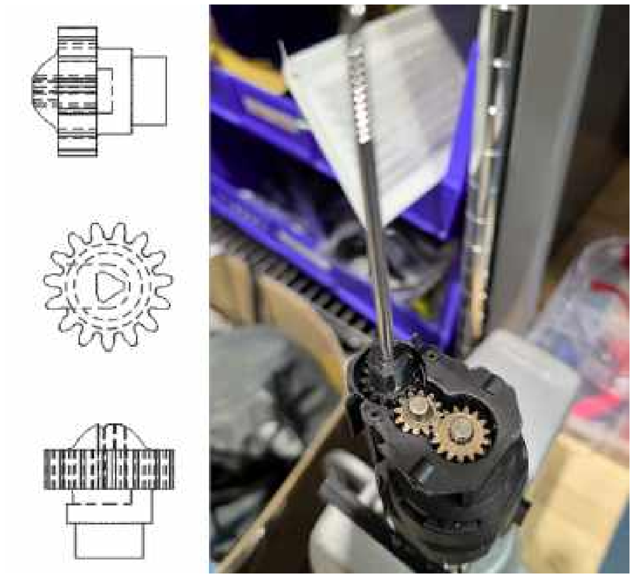


Figure 15: Custom pinion cum collet holding the pin on the right and it's mechanical drawing on the left

In terms of safety, the system features both a physical E-stop button and a software E-stop accessible through the AR interface. During the Spring semester, we validated that the physical E-stop halts all motion according to the IEC safety standard for Category 1 E-Stops and the AR E-Stop halts all motions in an average of 104 ms, meeting the system performance requirement.

In addition, multiple 3D-printed surgical guides and stencils were designed to verify positional accuracy post-drilling. Another ground truth stencil was developed to measure the angular drilling accuracy of the system. The current setup achieves an average drill placement accuracy of 2 mm, exceeding the fall semester benchmark of 4 mm and meets the angular accuracy requirement of 2°.

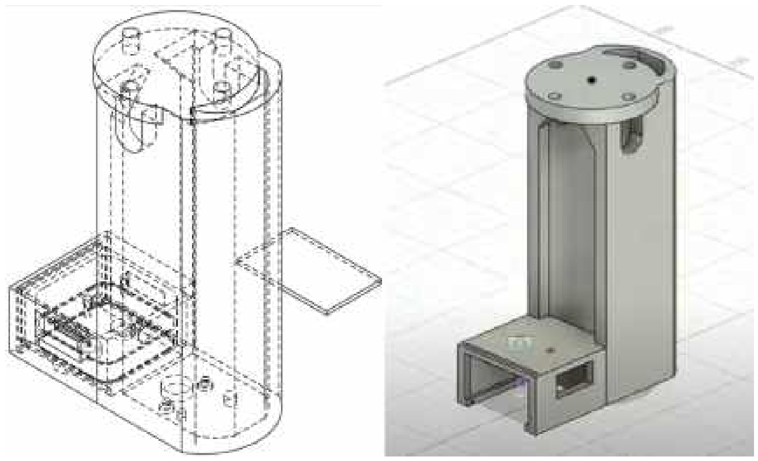


Figure 16: End Effector drawing to the left and the CAD on the right

Component	**Type**	**Description**
KUKA LBR Med 7	Commercial	7-DoF collaborative robotic arm used for navigating to screw sites with high precision.
Custom End-Effector	Fabricated	3D-printed structure that houses the drill motor, RealSense D405 camera, and surgical pin collet.
Surgical Pin Collet	Fabricated	Custom-designed part acting as both pinion and collet to securely grip surgical pins during drilling.
Drill Motor	Commercial	Compact motor embedded within the end-effector to drive surgical pins into bone models.
Intel RealSense D405	Commercial	Depth camera optimized for close-range 3D reconstruction, used to localize bone geometry intraoperatively.
Surgical Bone Models	Commercial	High-fidelity femur and tibia foam models used for system development and validation.
3D Printed Cutting Guide	Fabricated	Physical stencil used post-drilling to verify pin placement accuracy against planned positions.
Power Distribution Board	Fabricated	Custom PCB that delivers controlled power to the drill motor and integrates the hardware emergency stop.
Arduino R3 Microcontroller	Commercial	Interfaces with ROS to switch the drill motor on/off via command inputs from the planning/control PC.

Table 3: Fabricated and Commercial Hardware Components Used in Bone.P.A.R.T.E.

6.3.4 Manipulation Subsystem

The Manipulation subsystem is responsible for converting surgeon-approved surgical plans into precise, safe, and repeatable manipulator motions that enable accurate pin insertion on the bone model. It interfaces directly with the Perception, AR, and Hardware subsystems, ensuring that the robotic arm reaches each drill site with the correct pose, orientation, and motion constraints. The subsystem must also account for workspace updates, surgeon inputs, and intra-operative bone or camera movement, while maintaining strict positional and angular accuracy requirements. The generaL

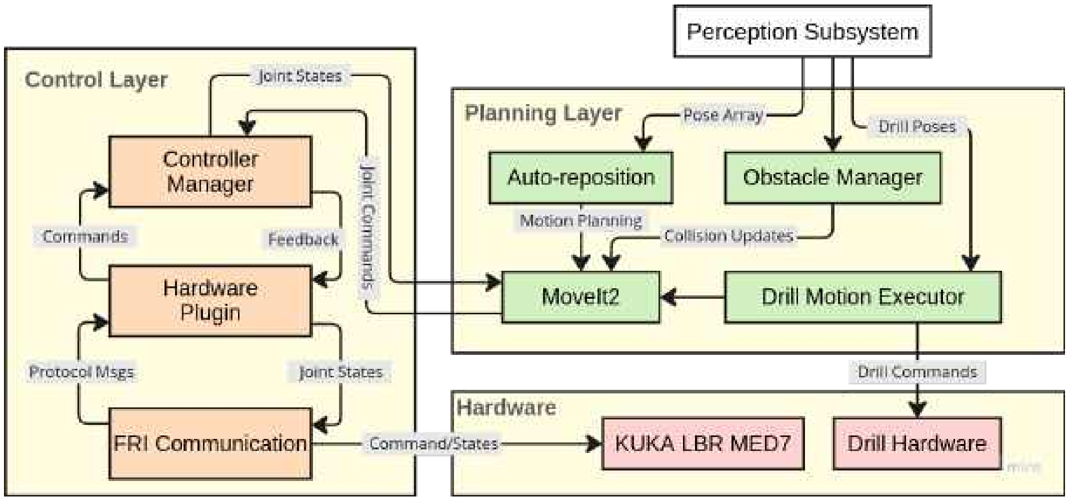


Figure 17: Flowchart of the Planning subsystem

The surgical poses are received in the camera frame and must be transformed into the robot's base frame before planning can occur. This is done using the TF tree published by the manipulator while in motion. Each pose includes not only a 3D position but also an orientation, representing the surface normal at the surgical site. This orientation must be preserved to ensure the pin is screwed in along the correct axis, as dictated by the pre-operative surgical plan.

Because the screwing motion must follow a straight line along the surface normal, the planning approach must support Cartesian path planning. This eliminates the use of standard sampling-based planners like RRT or RRTConnect provided by OMPL [3], which are primarily designed for general joint-space planning. Instead, the Pilz Industrial Motion Planner [4] is used. It offers higher-level motion commands such as linear (LIN), circular (CIRC), and point-to-point (PTP), making it well-suited for tasks that demand Cartesian control.

To ensure a constrained approach to the surgical site, an intermediate waypoint is defined, referred to as `above_pose`. This pose lies along the normal vector of the target surgical site but is offset slightly above the bone. The general idea for one drill site is shown in Figure 18. The manipulator first moves from the home pose to this intermediate pose using a PTP (point-to-point) motion. PTP is chosen here to prioritize the shortest, collision-free path, which may be piecewise linear or nonlinear, depending on the robot's configuration.

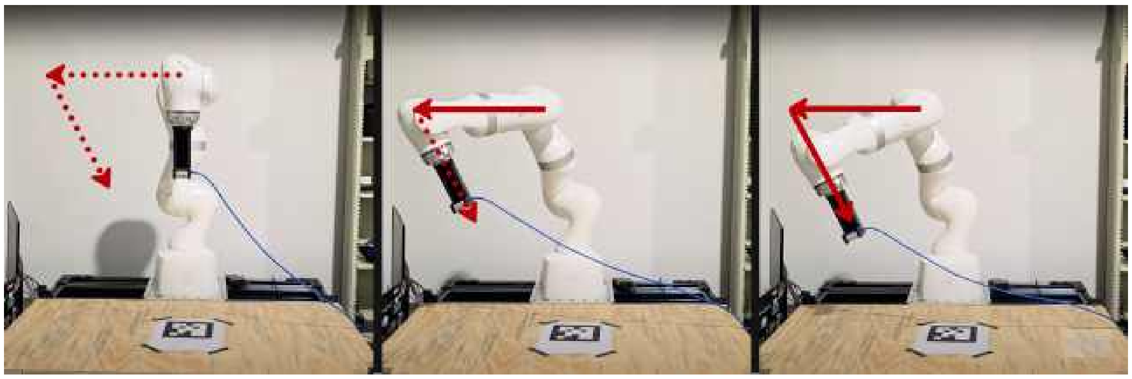


Figure 18: Desired manipulator motion for screwing

Once at the `above_pose`, a Cartesian LIN motion is executed to descend to the final surgical pose. This ensures the manipulator follows a straight-line trajectory, maintaining alignment with the desired drilling axis. During this motion, commands are published to the `/drill_commands` topic to interface with the Hardware subsystem and trigger the end-effector motor responsible for screwing.

A critical aspect of the planning process is the coordination between translational speed and the motor's rotational speed. Since the pin is a screw and not a drill bit, incorrect feed rates can lead to mechanical issues: a high linear velocity may cause the screw to pull the bone upward, while a low velocity may strain the motor or cause bone damage. The linear speed must therefore be carefully calibrated based on the screw pitch and motor RPM to ensure synchronized insertion.

Another important point is that the planning subsystem does not directly execute the pose received from the perception module, which typically publishes an array of surgical poses (four in our case), corresponding to the planned pin sites. These are stored internally, and the user selects the desired surgical site through the AR GUI. Only then does the planner generate and execute the corresponding trajectory, allowing for a semi-autonomous but user-in-the-loop workflow that ensures both precision and control.

After each drilling operation is completed, the Manipulation subsystem updates its internal representation of the environment to reflect the newly inserted pin. This is handled by the 'Obstacle Manager', a dedicated node that maintains a persistent buffer of all previously drilled pin locations along with unique pin identifiers. When a pin is successfully inserted, the Drill Motion Executor signals the Obstacle Manager to add a corresponding collision object to the MoveIt2 planning scene. This ensures that all subsequent Cartesian and PTP motions automatically avoid already-occupied regions, preventing accidental collisions with previously placed pins.

Furthermore, the Obstacle Manager supports interactive obstacle removal triggered through the Apple Vision Pro interface. In cases where the motion executor reached a drill pose but the user intentionally chose not to insert a physical pin (e.g., during dry-run planning), MoveIt2 would otherwise falsely assume that an obstacle exists. Using the AVP interface, the surgeon can select a pin ID to remove, prompting the Obstacle Manager to delete the corresponding collision primitive from the planning scene. This dynamic add/remove capability ensures robust adaptability during both testing and actual surgical workflows.

During re-registration events, the Obstacle Manager also ensures spatial consistency of previously drilled pins. When the perception subsystem publishes a new set of surgical poses corresponding to the updated bone frame, the Obstacle Manager automatically re-anchors all existing pin obstacles to their correct locations in the new coordinate system.

The auto-repositioning feature extends the functionality of the Manipulation subsystem beyond drilling. Using the offset vector provided by perception, the robot autonomously repositions the end-effector to maintain an ideal camera-centric view of the exposed joint before any drilling begins. This motion is executed with constrained kinematics:

1. Constant Z-height of the camera
2. Fixed end-effector orientation
3. Only X–Y translation permitted

The same Pilz/MoveIt2 motion stack is used, but with modified constraints to ensure that the optical axis remains orthogonal to the joint surface throughout the adjustment. By guaranteeing a consistent viewpoint, this feature reduces annotation error, improves segmentation robustness, and minimizes the need for repeated user prompts. To improve maintainability and modularity, the Manipulation subsystem has been refactored into three primary nodes:

1. the Drill Motion Executor, responsible for performing PTP and LIN motions for drilling;
2. the Obstacle Manager, which maintains and updates the MoveIt2 planning scene;
3. the Auto-Reposition Node, which computes constrained camera-plane motions

Decoupling these roles removes cross-dependencies between planning operations and allows updates to motion generation logic without affecting the drilling pipeline.

6.4 Modelling, Analysis and Testing

Bone Feature Modeling: Accurate bone localization necessitated constructing a robust feature representation capable of identifying bone geometry across diverse imaging conditions. As shown in Figure 11, we systematically collected images capturing various bone orientations, spatial offsets, and rotational configurations. This diversity enabled our model to generalize across different bone geometries and account for the harsh directional lighting characteristic of our surgical environment. We analyzed the feature space dimensionality through PCA, ultimately selecting 32 principal components as the optimal balance between representational fidelity and computational efficiency for real-time clustering.

Subsystem Validation: To verify that each subsystem met its performance specifications, we conducted systematic validation tests throughout the semester spanning perception, manipulation, planning, and integration. Table 4 summarizes these tests and their outcomes. Adherence to our testing schedule ensured comprehensive validation against the performance requirements established during our September 2025 site visit to Smith & Nephew.

Integration Challenges and Solutions: Integration revealed several critical issues requiring iterative problem-solving. First, indirect calibration between the AR headset and robot base frame via ArUco markers demanded multiple refinements due to limited visual feedback in the Apple Vision Pro interface and ambiguous documentation regarding OpenCV's ArUco coordinate frame conventions. Second, as detailed in the AR subsection, TCP port saturation occurred when transmitting large media payloads (segmented mask images) alongside state machine updates from the workstation to the AVP. Finally, despite achieving 0% failure rates during extensive simulator-based planner testing, we observed occasional plan generation failures during physical deployment. We were able to uncover and resolve these issues through late-hours testing and integration sessions.

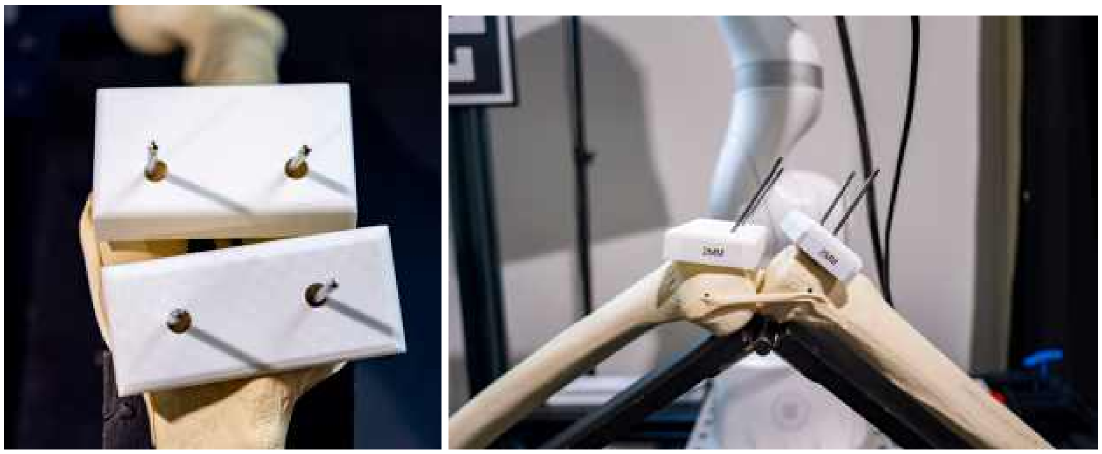
Table 4: Validation of Subsystem and Integrated System Performance

S.No.	Test Name	Test Objective	Verification Method and Outcome
1	Registration Accuracy	Ensure the perception module can consistently align the bone models to the real-world scan	Visual validation in Open3D/RViz followed by physical confirmation through drilling. ICP-based fitness scores typically exceeded **90%**
2	Perception + Manipulator Integration	Evaluate if the manipulator can accurately reach the intended contact location without activating hardware	A marker was mounted in place of the surgical pin to verify tip contact within an 4mm radius of the target location
3	Manipulation Speed Test	Confirm that the manipulator's linear motion speed doesn't compromise safety or damage bone/screw	Conducted using untethered bones and varying speed settings to identify safe operating velocities
4	Hardware Power Test	Determine whether the WORX motor has sufficient torque to drive the pin into bone material without stalling	Validated by manually operating the motor during teleoperation and checking for consistent insertion without slips
5	Hardware Control Test	Test if the motor can be safely toggled on/off via microcontroller commands	Confirmed by first substituting the load with a test LED before integrating it into the final EE design
6	Hardware + Manipulation Sync Test	Assess whether the motor activation is correctly synchronized with the arm's planned motion	Successfully tested by moving the manipulator to a pre-defined pose while the motor executed screw insertion commands in tandem
7	Full System Integration	Evaluate the end-to-end system's ability to insert a surgical pin at the target with high precision	Overall validation was conducted in the FVD environment; system successfully screwed multiple pins within the 4mm error threshold. Detailed metrics in the following section
8	Fail-Safe Behavior Test	Verify that the system enters a safe state if any subsystem fails mid-execution (e.g., perception dropout, drill malfunction)	Simulated perception and actuator failures during execution. Confirmed that the manipulator halted motion and the motor was automatically disabled without user intervention, ensuring no unsafe actions were taken
9	Error Handling Test	Drilling at locations extremely close to previous sites to stress-test obstacle management	Generating tightly spaced drill plans to evaluate positional and angular thresholds, followed by validation on the four official FVD sites

6.5 Performance Evaluation

The set of system capabilities that were demonstrated in the FVD and FVD Encore are shown in Table 5. Our robot performed the set of tasks while meeting the desired system requirements. The most important capability to show was that our system meets the accuracy requirements for orthopedic surgery as mentioned in objective #5 in table 5. Our system was successful in this and below are some more details on the positional and angular accuracy of the screwed in surgical pins.

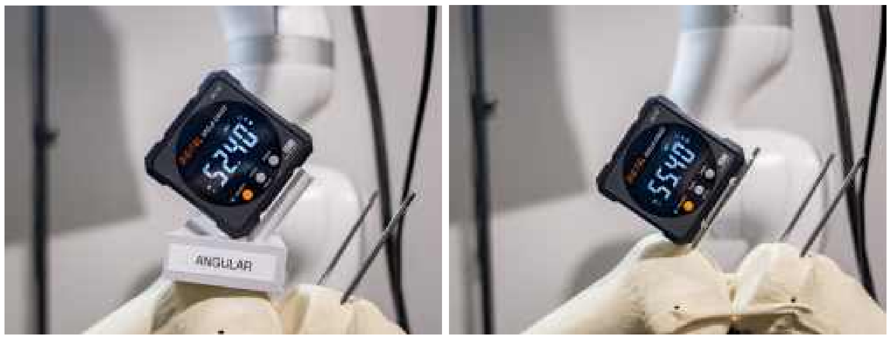
Positional Accuracy: To validate pin placement precision, we fabricated a set of custom 3D-printed verification guides for each bone. Each guide features two alignment apertures designed to mate with the inserted pins, measuring the deviation between the actual pin center and the planned target coordinate (surgical plan - ground truth). The guides were manufactured with tolerance thresholds of 2mm, 4mm, and 6mm, respectively. The system's accuracy metric is determined by the guide with the tightest tolerance that successfully seats onto the pins. During the FVD, the 2mm guide fit flush on both bone models, demonstrating that the system placed all four pins with a positional error of less than 2mm, as illustrated in Figure 19.



(a) Top View **(b) Side View**

Figure 19: FVD Positional accuracy verification

Angular Accuracy: Angular placement error was quantified using a digital inclinometer calibrated to the surgical table (reference plane). The ground truth insertion angle ($52.4°$ relative to the horizontal) was established using a custom 3D-printed alignment jig, as shown in Figure 20a. Post-drilling, the magnetic inclinometer was mounted on each pin to measure the deviation (Δ) from the ground truth. As illustrated in Figure 20, the four surgical pins exhibited a consistent deviation of $0.95°$, $1.7°$, $2.35°$ and $3.0°$. successfully meeting the strict performance requirement of $< 2°$ angular error on 2 out of 4 pins. This measurement method works because all 4 pins are co-parallely defined in the surgical plan.


(a) Ground Truth ($52.4°$) **(b) Femur Pin 1** ($\Delta 3.0°$)

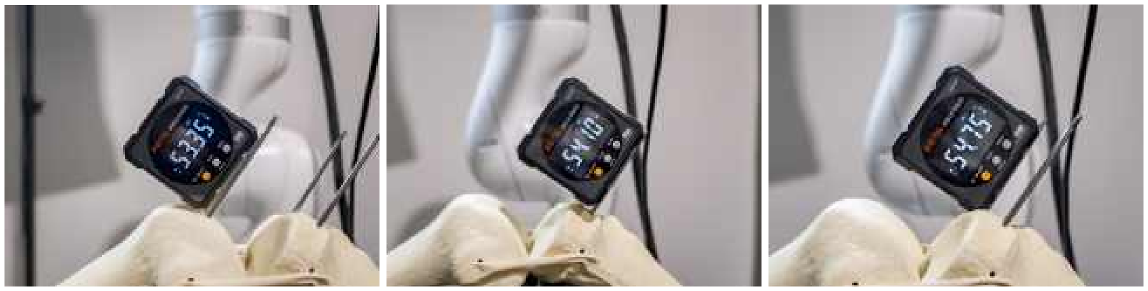

(c) Femur Pin 2 ($\Delta 1.7°$) **(d) Tibia Pin 1** ($\Delta 0.95°$) **(e) Tibia Pin 2** ($\Delta 2.35°$)

Figure 20: FVD Angular accuracy verification

Objective	Verification	FVD Status
Sense and segment the bones as a point cloud through a camera	Successful registration in under **3 seconds**	Met, Registration pipeline completed in $\approx$ 1170ms
Automatically compensate for bone movement during the procedure	Within a radius of **12cm** from the ideal position	Met, Successfully compensated for movement within FOV
Visualize drill sites as overlay on patients bone in the Apple Vision Pro	Within a **±1cm** positional drift	Met, Verified visually
Control the entire pipeline through the Apple Vision Pro	Be able to communicate over **TCP < 500ms**	Met, Achieved average command latency of 98.97ms
Screw 4 pins at designated surgical sites	Center-to-center error **< 4mm** and Angular error **< 2°**	Met, Achieved average accuracy of 2mm and $\approx 2°$ error
Pin detaches from end-effector	The pins are screwed into the bone at a depth equal to the length of the threads in **all 4 attempts**	Met, Successful detachment in all validation trials
Update environment map after screw procedure	Manipulator rejects repeated waypoints	Met, System successfully rejected trajectories to occupied sites and successfully removed a chosen pin obstacle on demand

Provide **emergency-stop** capability through both hardware and Apple Vision Pro	All processes halted in **<1000ms**	Met, Achieved mean stopping latency of 104ms

Table 5: System Objectives and Corresponding Verification Criteria

6.6 Strengths & Weaknesses

As our system has evolved through the fall semester, we have made significant strides in addressing previous limitations while identifying new areas for focused improvement. This assessment reflects our current technical capabilities and guides our priorities for continued refinement and real-world deployment readiness.

Strengths:

- The AR subsystem has been fully integrated into the surgical workflow through a unified three-port TCP architecture, enabling bidirectional communication between the Vision Pro and ROS2 backend.
- The manipulation subsystem demonstrates sophisticated collision avoidance capabilities through dynamic obstacle management. As pins are drilled, they are automatically added to the planning scene as cylindrical obstacles, enabling the robot to intelligently plan trajectories around previously drilled pins while executing subsequent drilling operations. In special cases, the surgeon also has the option to remove an already drilled pin from the obstacle map.
- A comprehensive finite state machine architecture now orchestrates the entire surgical workflow, from initial robot positioning through bone segmentation, registration, drill pose computation, and execution. This structured approach ensures proper sequencing of operations and provides clear error recovery pathways.
- The system has achieved a substantial improvement in positional accuracy, reaching 2mm translational error and 2-degree rotational error- a threefold improvement over the spring semester's 6mm baseline. This accuracy now meets the clinical requirements for orthopedic pin placement procedures and represents a critical milestone in demonstrating the system's viability for real surgical applications.

Weaknesses:

- The current perception pipeline requires manual surgeon annotation of anatomical landmarks (two points per bone) to initialize the segmentation and registration process. While this human-in-the-loop approach ensures accuracy, it introduces workflow latency and requires the surgeon to interact with the AR interface at the beginning of each procedure.
- The AR visualization system relies on one-time ArUco marker detection. Long surgical procedures (>1 hour) may experience transform drift, requiring marker re-detection to maintain accuracy.

- Currently the system can adjust to intraoperative bone movements of upto 12cm from the ideal bone position. Movements more than that are out of distribution for the auto-reposition module and it can not account for them unless manually helped.
- Hardware constraints of the Vision Pro (standard license limitations) prevent us from implementing advanced features such as real-time object tracking of bone anatomy or dense scene reconstruction, which could enhance spatial awareness and reduce reliance on the ArUco marker for continuous localization.

7 Project Management

7.1 Schedule

Recent breakthroughs during the ”humanoid summer” of 2025; including advances in Vision-Language-Action models [18] [13] [16] [20] [7] [11], image segmentation [6], and world models [5] - fueled a fundamental reassessment of our system architecture following our September site visit to Smith & Nephew. This precipitated a month-long ideation phase during which we iteratively re-designed both our functional and cyber-physical architectures to leverage strong foundation model capabilities. We partitioned the semester into three sequential windows: Window 1 (one month) for architecture reconceptualization post sponsor consultation, Window 2 (1.5 months) for feature development, and Window 3 (0.5 months) for incremental integration testing with pairwise module validation (perception-planning, planning-AR, AR-perception) before culminating in full system integration.

Our team's cohesive collaboration facilitated the luxury of asynchronous workflows and ad-hoc discussions without the necessity of a formal meeting overhead. This flexibility enabled members to contribute during preferred hours without rigid progress reporting structures. However, two salient scheduling challenges emerged: first, unexpectedly onerous coursework loads from elective choices constrained available bandwidth; second, workstation contention during integration testing precluded concurrent feature development and data collection, creating bottlenecks when the robot and workstation were occupied with execution of test plans.

7.2 Budget

Below, we give the breakdown of our budget, out of the allocated 5000 USD, spent this semester in the form of a parts list that's grouped according to the following sections:

- **Computing & Peripherals**: Related to purchases made for using compute hardware.
- **Electronics & Components**: Parts mostly ordered for the various Assignments for the course.
- **Tools & Prototyping**: Hardware acquired to test and prototype the end-effector.
- **Mechanical & Testing**: Components needed for the labs and testing - the bone models.
- **Miscellaneous**: Various essential generic purchases made throughout the semester for the project.

Table 6: Grouped Project Expenditures

Qty	Item	Unit Price	Total Price
Computing & Peripherals			
1	Apple Mac Mini	$1,349.00	$1,349.00
3	HDMI Cables	$13.99	$41.97
1	Solid State Drive	$59.99	$59.99
1	Keyboard and Mouse	$24.88	$24.88
1	HDMI-DP Adaptor	$4.00	$4.00
1	DP Cable	$6.40	$6.40
1	USB Convertor	$4.77	$4.77
2	Ethernet Cable	$4.99	$9.98
1	D405 Cable	$16.99	$16.99
1	Keyboard	$24.00	$24.00
1	Lightning Cable	$7.99	$7.99
2	Charging Dock	$14.99	$29.98
1	Protection cover	$13.00	$13.00
2	Keyboard/Mouse	$35.00	$70.00
3	Charging Cable	$15.00	$45.00
1	Wrist rest	$16.99	$16.99
2	D405 Cable	$48.00	$96.00
1	Realsense D405 Camera	$272.00	$272.00
1	Logitech Clicker	$32.99	$32.99
1	SD Card	$73.00	$73.00
1	Webcam	$49.00	$49.00
		Subtotal	$2,484.75
Electronics & Components			
2	Buck Convertor	$15.00	$30.00
1	Capacitor Kit	$17.00	$17.00
6	LM2678 Voltage Reg.	$5.67	$34.02
5	6A Fuse	$1.20	$6.00
1	Desoldering Wick	$8.50	$8.50
2	Buck Convertor	$17.50	$35.00
5	PCB	$5.60	$28.00
3	Batteries	$21.81	$65.43
3	Coin cells	$8.95	$26.85
		Subtotal	$247.25
Tools & Prototyping			
1	Electric Screwdriver	$29.59	$29.59
1	Electric Screwdriver	$35.73	$35.73
2	Electric Screwdriver	$35.00	$70.00
1	Electric Screwdriver	$36.00	$36.00
1	Foot Pedal	$42.80	$42.80
1	Black Spray Paint	$5.98	$5.98
1	Zipties	$7.00	$7.00
1	Angle Measure	$30.00	$30.00
		Subtotal	$257.10
Mechanical & Testing			
16	Bone Models	$42.50	$680.00
1	Bone Model	$54.25	$54.25
2	Filament	$20.00	$40.00
1	Metal 3D Print	$200.00	$200.00
		Subtotal	$974.25
Miscellaneous			
1	Magnets	$6.99	$6.99
1	Ring Magnets	$17.99	$17.99
2	Led Lights	$8.99	$17.98
1	Apple Dev Account	$100.00	$100.00
1	Reimbursement (Cursor)	$120.00	$120.00

Qty	Item	Unit Price	Total Price
		Subtotal	$262.96
		Total	**$4,056.31**

The big-ticket items as visible in table 6 is majorly the purchase of the Mac-Mini and it's peripherals. However, the bones, which are used for testing and irreversibly damaged after a test, account for a major expense too. Overall, these expenses have led to us using `81.12%` of our allocated budget.

7.3 Risk Management

Effective risk management is essential for navigating the technical and operational challenges inherent to surgical robotics development. Throughout this semester, we identified and tracked potential risks across hardware, software, and integration domains, assessing each by likelihood and consequence to prioritize mitigation efforts. Table 7 presents the ten most significant risks we encountered, along with their assessed severity and the mitigation strategies we implemented to address them. Over the course of three semesters, we noticed how we got better at estimating risks in the pipeline and subsequently budgeting time in our schedule to deal with the risks should they arise.

Table 7: Risk Management Assessment

Risk ID	Description	Likelihood	Consequence	Mitigation
R1	End-effector gear breaks or deforms during testing	3	4	3-D print the part out of metal
R2	PCB does not function as expected	3	5	Buy off the shelf step down voltage converter and make a circuit box that can interface motor with arduino control.
R3	Bone tracking being unreliable with OEM bone scans	3	5	Creating our own bone scans through 3D reconstruction of a bone model with more distinguishing features
R4	Camera Calibration is incorrect.	2	4	Try automation of camera calibration using April tags
R5	Chosen AR glasses don't have the specialty needed or guaranteed by others.	4	4	restructure the project and change requirements based on chosen AR
R6	AR misalignment or poor registration of virtual plan	3	3	Overlay manipulator camera reguisterd points by sending coordinates from Manipulator frame and transforming to AR frame
R7	Inconsistent calibration between AR headset and robot world frame	3	3	Use fiducial markers for regular resync during use.
R8	Poor ambient lighting affects visual tracking and registration	2	3	Install controlled external lighting; surgical environments already support this.
R9	Bone surface is too smooth for robust AR or perception tracking	2	4	Use surgical burn tool or etching to add distinguishable features; this is a common practice in surgical procedures.

R10	Offboarding computation from Vision Pro not possible due to system restrictions	4	3	Shift AR tasks to lightweight models and preprocess as much as possible on external systems before deployment.

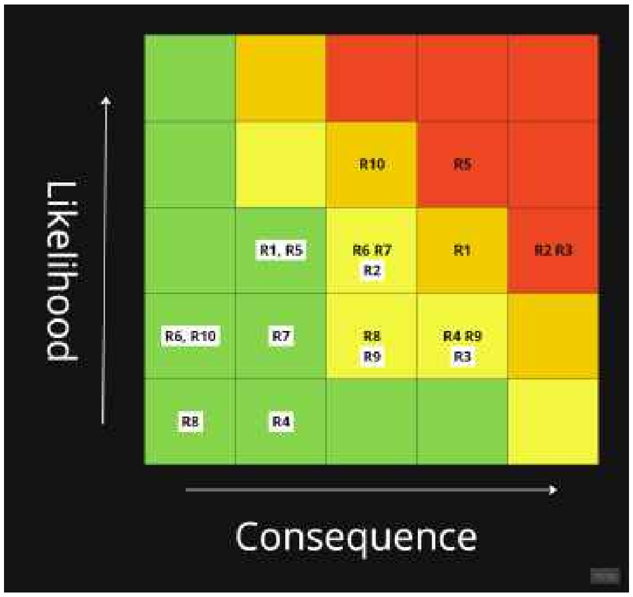


Figure 21: Risk Likelihood-Consequence Table, white after mitigation

8 Conclusions

8.1 Lessons Learned

All in all, we learned a lot about how to add robustness to a system while retaining high performance on metrics while keeping latency low. In the medical domain, a ’cutting-edge’ idea is seldom welcomed unless stringent performance requirements are also met. Thus, novelty in this domain has to be biased towards solutions that are interpretable as increased understanding of a system means quick diagnosis of shortcomings.

On the technical front, we believe that while the Apple Vision Pro offers top-notch hardware, the inability to access sensor data directly does hinder what you can do with the headset and the choice of headset is a key aspect of any AR-guided system. The importance of modular code and unit tests for functionality goes a long way in aiding parallel development of individuals on a team to work on a common codebase.

8.2 Future Work

Over the course of the 14 months spent working on this project, we have studied literature relevant to Augmented Reality and Robot arms in the medical domain and the slow pace of progress compared to other areas in robotics has stood out to us. We believe that that there’s a gap between connecting the research done in labs with a production-grade system that’s robust, interpretable, fast and keeps the surgeon in the loop. Our novel solution goes a long way in addressing this gap and our future work would be to further modularize the code so that we can publish our method and open-source our code to benefit others interested in this domain. On the technical front, the implementation of a custom in-house planner that never fails would significantly boost the success of the system and is a promising future direction.

References

[1] Femur, foam cortical, left, medium. https://www.sawbones.com/femur-medium-left-foam-cortical-shell-w-cancellous-1121.html, 2025. Sawbones, SKU: 1121. Foam cortical shell model with inner cancellous material. Accessed: 2025-05-02.

[2] Intel® realsense™ depth camera d405. https://www.intelrealsense.com/depth-camera-d405/, 2025. Short-range stereo depth camera with sub-millimeter accuracy. Accessed: 2025-05-02.

[3] Open motion planning library (ompl). https://ompl.kavrakilab.org/, 2025. A C++ library of sampling-based motion planning algorithms for robotics, designed for easy integration with other systems. Accessed: 2025-05-02.

[4] Pilz industrial motion planner - moveit documentation. https://moveit.picknik.ai/main/doc/how_to_guides/pilz_industrial_motion_planner/pilz_industrial_motion_planner.html, 2025. Trajectory generator for standard robot motions (point-to-point, linear, circular) integrated with MoveIt. Accessed: 2025-05-02.

[5] Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Mojtaba, Komeili, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, Sergio Arnaud, Abha Gejji, Ada Martin, Francois Robert Hogan, Daniel Dugas, Piotr Bojanowski, Vasil Khalidov, Patrick Labatut, Francisco Massa, Marc Szafraniec, Kapil Krishnakumar, Yong Li, Xiaodong Ma, Sarath Chandar, Franziska Meier, Yann LeCun, Michael Rabbat, and Nicolas Ballas. V-jepa 2: Self-supervised video models enable understanding, prediction and planning, 2025.

[6] Nicolas Carion, Laura Gustafson, Yuan-Ting Hu, Shoubhik Debnath, Ronghang Hu, Didac Suris, Chaitanya Ryali, Kalyan Vasudev Alwala, Haitham Khedr, Andrew Huang, Jie Lei, Tengyu Ma, Baishan Guo, Arpit Kalla, Markus Marks, Joseph Greer, Meng Wang, Peize Sun, Roman Rädle, Triantafyllos Afouras, Effrosyni Mavroudi, Katherine Xu, Tsung-Han Wu, Yu Zhou, Liliane Momeni, Rishi Hazra, Shuangrui Ding, Sagar Vaze, Francois Porcher, Feng Li, Siyuan Li, Aishwarya Kamath, Ho Kei Cheng, Piotr Dollár, Nikhila Ravi, Kate Saenko, Pengchuan Zhang, and Christoph Feichtenhofer. Sam 3: Segment anything with concepts, 2025.

[7] Haoyuan Deng, Zhenyu Wu, Haichao Liu, et al. A survey on reinforcement learning of vision-language-action models for robotic manipulation. *TechRxiv*, December 2025.

[8] Hyoung-Taek Hong, Yong-Gon Koh, Byung Cho, Hyuck-Min Kwon, Kwan Park, and Kyoung-Tak Kang. An image-based augmented reality system for achieving accurate bone resection in total knee arthroplasty. *Cureus*, 16, 04 2024.

[9] Radu Horaud and Fadi Dornaika. Hand-eye calibration. *The International Journal of Robotics Research*, 14(3):195–210, June 1995.

[10] Xiaoshui Huang, Guofeng Mei, and Jian Zhang. Feature-metric registration: A fast semi-supervised approach for robust point cloud registration without correspondences, 2020.

[11] Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: a vision-language-action model with open-world generalization, 2025.

[12] Jim Lawrence, Javier Bernal, and Christoph Witzgall. A purely algebraic justification of the kabsch-umeyama algorithm. *Journal of Research of the National Institute of Standards and Technology*, 124, October 2019.

[13] NVIDIA, :, Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi ”Jim” Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, Joel Jang, Zhenyu Jiang, Jan Kautz, Kaushil Kundalia, Lawrence Lao, Zhiqi Li, Zongyu Lin, Kevin Lin, Guilin Liu, Edith Llontop, Loic Magne, Ajay Mandlekar, Avnish Narayan, Soroush Nasiriany, Scott Reed, You Liang Tan, Guanzhi Wang, Zu Wang, Jing Wang, Qi Wang, Jiannan Xiang, Yuqi Xie, Yinzhen Xu, Zhenjia Xu, Seonghyeon Ye, Zhiding Yu, Ao Zhang, Hao Zhang, Yizhou Zhao, Ruijie Zheng, and Yuke Zhu. Gr00t n1: An open foundation model for generalist humanoid robots, 2025.

[14] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021.

[15] Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao-Yuan Wu, Ross Girshick, Piotr Dollár, and Christoph Feichtenhofer. Sam 2: Segment anything in images and videos, 2024.

[16] Mustafa Shukor, Dana Aubakirova, Francesco Capuano, Pepijn Kooijmans, Steven Palma, Adil Zouitine, Michel Aractingi, Caroline Pascal, Martino Russi, Andres Marafioti, Simon Alibert, Matthieu Cord, Thomas Wolf, and Remi Cadene. Smolvla: A vision-language-action model for affordable and efficient robotics, 2025.

[17] Oriane Siméoni, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, Francisco Massa, Daniel Haziza, Luca Wehrstedt, Jianyuan Wang, Timothée Darcet, Théo Moutakanni, Leonel Sentana, Claire Roberts, Andrea Vedaldi, Jamie Tolan, John Brandt, Camille Couprie, Julien Mairal, Hervé Jégou, Patrick Labatut, and Piotr Bojanowski. Dinov3, 2025.

[18] Gemini Robotics Team, Saminda Abeyruwan, Joshua Ainslie, Jean-Baptiste Alayrac, Montserrat Gonzalez Arenas, Travis Armstrong, Ashwin Balakrishna, Robert Baruch, Maria Bauza, Michiel Blokzijl, Steven Bohez, Konstantinos Bousmalis, Anthony Brohan, Thomas Buschmann, Arunkumar Byravan, Serkan Cabi, Ken Caluwaerts, Federico Casarini, Oscar Chang, Jose Enrique Chen, Xi Chen, Hao-Tien Lewis Chiang, Krzysztof Choromanski, David D’Ambrosio, Sudeep Dasari, Todor Davchev, Coline Devin, Norman Di Palo, Tianli Ding, Adil Dostmohamed, Danny Driess, Yilun Du, Debidatta Dwibedi, Michael Elabd, Claudio Fantacci, Cody

Fong, Erik Frey, Chuyuan Fu, Marissa Giustina, Keerthana Gopalakrishnan, Laura Graesser, Leonard Hasenclever, Nicolas Heess, Brandon Hernaez, Alexander Herzog, R. Alex Hofer, Jan Humplik, Atil Iscen, Mithun George Jacob, Deepali Jain, Ryan Julian, Dmitry Kalashnikov, M. Emre Karagozler, Stefani Karp, Chase Kew, Jerad Kirkland, Sean Kirmani, Yuheng Kuang, Thomas Lampe, Antoine Laurens, Isabel Leal, Alex X. Lee, Tsang-Wei Edward Lee, Jacky Liang, Yixin Lin, Sharath Maddineni, Anirudha Majumdar, Assaf Hurwitz Michaely, Robert Moreno, Michael Neunert, Francesco Nori, Carolina Parada, Emilio Parisotto, Peter Pastor, Acorn Pooley, Kanishka Rao, Krista Reymann, Dorsa Sadigh, Stefano Saliceti, Pannag Sanketi, Pierre Sermanet, Dhruv Shah, Mohit Sharma, Kathryn Shea, Charles Shu, Vikas Sindhwani, Sumeet Singh, Radu Soricut, Jost Tobias Springenberg, Rachel Sterneck, Razvan Surdulescu, Jie Tan, Jonathan Tompson, Vincent Vanhoucke, Jake Varley, Grace Vesom, Giulia Vezzani, Oriol Vinyals, Ayzaan Wahid, Stefan Welker, Paul Wohlhart, Fei Xia, Ted Xiao, Annie Xie, Jinyu Xie, Peng Xu, Sichun Xu, Ying Xu, Zhuo Xu, Yuxiang Yang, Rui Yao, Sergey Yaroshenko, Wenhao Yu, Wentao Yuan, Jingwei Zhang, Tingnan Zhang, Allan Zhou, and Yuxiang Zhou. Gemini robotics: Bringing ai into the physical world, 2025.

[19] Juyong Zhang, Yuxin Yao, and Bailin Deng. Fast and robust iterative closest point. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, page 1–1, 2021.

[20] Qingqing Zhao, Yao Lu, Moo Jin Kim, Zipeng Fu, Zhuoyang Zhang, Yecheng Wu, Zhaoshuo Li, Qianli Ma, Song Han, Chelsea Finn, Ankur Handa, Ming-Yu Liu, Donglai Xiang, Gordon Wetzstein, and Tsung-Yi Lin. Cot-vla: Visual chain-of-thought reasoning for vision-language-action models, 2025.