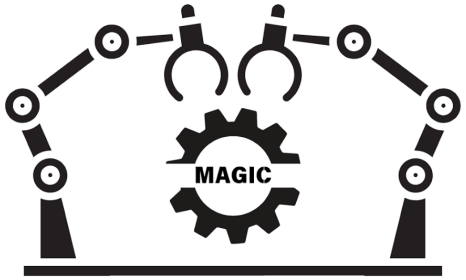


MAGIC

Multi-Agent Geometric Inspection and Classification

Team H: Final Report

Kartik Agrawal
Manyung Emma Hon
Kailash Jagadeesh
Megan Lee
Shreya Ragi

Sponsor: Jiaoyang Li

December 1, 2025

Carnegie
Mellon
University

Abstract

Industrial inspection systems increasingly rely on automated solutions to improve consistency, reduce downtime, and handle complex components. However, most existing robotic inspection platforms are limited by fixed sensor configurations or single-arm setups, which restrict coverage, flexibility, and adaptability, especially when dealing with large, occluded, or geometrically intricate objects. These constraints lead to incomplete reconstructions and missed defects, compromising inspection reliability in dynamic or unstructured environments. To address these limitations, we present MAGIC (Manipulation for Automated Geometric Inspection and Construction), a dual-arm robotic system designed to perform active inspection through coordinated motion and 3D reconstruction. By leveraging bimanual manipulation, MAGIC enables dynamic viewpoint adjustment and collaborative handling of inspection targets, ensuring higher-fidelity reconstruction and thorough surface coverage. The system integrates perception, real-time motion planning, and control within a modular ROS-2 based architecture, facilitating adaptive, task-driven behaviors. Our approach demonstrates improved reconstruction completeness and inspection accuracy over traditional static or single-arm setups, with potential applications in manufacturing, construction, and field robotics.

Contents

1 Project Description

In industrial, construction, and maintenance environments where precision inspection [2] is critical, current automated systems fall short due to their reliance on turntable reconstruction or single-arm manipulation strategies. These systems often fail to achieve comprehensive 3D reconstruction due to occlusions, limited viewpoints, and inadequate object repositioning. As a result, detailed inspection of complex or large-scale components remains a challenge, often requiring manual intervention that slows down workflows and increases the risk of human error.

MAGIC addresses these limitations by introducing a dual-arm robotic manipulation system designed to enable dynamic, object-centered inspection. Instead of repositioning cameras around a fixed object, MAGIC utilizes two Kinova Gen3 robotic arms to collaboratively manipulate and reorient objects in front of a static 3D sensor setup. This approach ensures rich viewpoint diversity without the need for moving camera systems, leading to higher-fidelity 3D reconstructions while simplifying calibration and system integration.

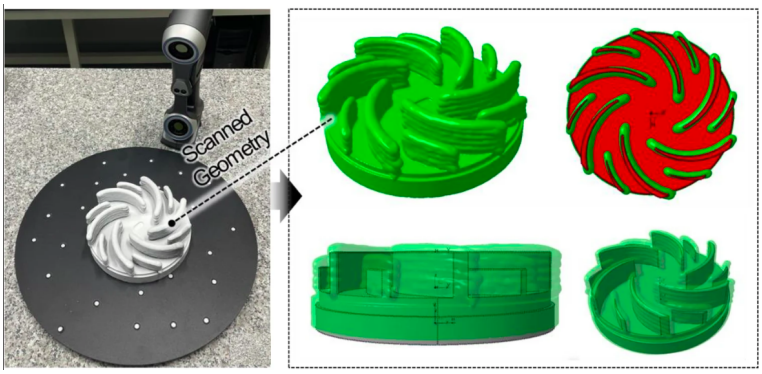


Figure 1: Project Use Case [4]

The robotic arms are controlled through a modular ROS-2 and MoveIt2 framework, supporting real-time Cartesian and joint-space motion planning for synchronized bimanual control. The system is designed to handle objects of varying geometries and sizes, allowing stable and flexible grasping strategies that expose occluded surfaces to the static camera array. Captured images are fused into a dense 3D model, which can then be used for tasks such as surface defect detection,as shown in fig. 1 structural measurement, and digital twin comparison.

MAGIC's approach combines precise manipulation with passive perception, offering a robust and scalable solution for environments where full object accessibility and inspection integrity are paramount. With applications spanning quality assurance, infrastructure assessment, and autonomous manufacturing, MAGIC sets the stage for more intelligent and adaptable robotic inspection systems.

2 Use Case

To address the challenges of post-processing inspection for large, additively manufactured components, the MAGIC system is deployed near the Wire Arc Additive Manufacturing (WAAM) [3] production area. The following use case illustrates a typical workflow, as shown in fig 2, highlighting the interaction between the operator and the robotic inspection system.

Ben, a manufacturing engineer at a high-mix production facility, initiates the inspection process for a recently fabricated WAAM component [1]. The part is placed within the shared workspace of MAGIC, a dual-arm robotic system paired with a fixed 3D sensing setup. Upon activation, the system utilizes high-precision sensors to estimate the object's pose, allowing it to autonomously plan collision-free trajectories for both manipulators.

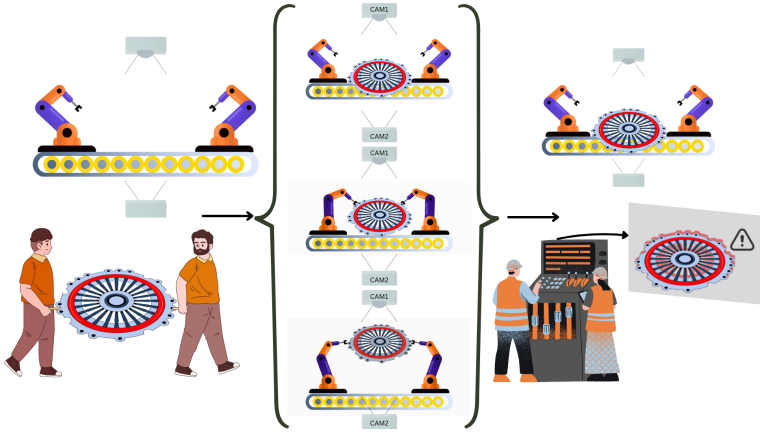


Figure 2: MAGIC Workflow

With the pose estimation complete, the system executes a synchronized bimanual grasp, securely lifting the component without disturbing its structural integrity. The arms then collaboratively reorient the part through a series of predefined poses, each chosen to maximize visibility of complex surface regions under the fixed 3D sensor array. As the component is manipulated, the system performs continuous scanning and data acquisition.

The collected point clouds are fused into a high-resolution 3D reconstruction of the component, capturing detailed geometry and surface features. This reconstructed model is automatically compared to a digital ground truth, enabling the detection of deviations such as surface roughness, dimensional inaccuracies, or incomplete deposition regions.

The results are visualized on Ben's operator interface, where he can inspect the deviation map and identify areas requiring further finishing. Using this feedback, Ben generates CNC-compatible M-codes through a CAD/CAM interface, precisely targeting the non-conforming regions. After verifying the code, he forwards it to the CNC station for post-processing.

This closed-loop workflow eliminates manual measurement tasks, reduces the likelihood of CNC rework, and ensures that each WAAM-produced component meets stringent geometric tolerances before entering the finishing pipeline. Once the inspection cycle is complete, MAGIC resets to a standby state, ready for the next part.

3 System Level Requirements

While the scope of the project extends beyond the MRSD deliverable, we have determined our mandatory functional requirements from a sub-part of our primary objectives. These requirements can be seen in Table 1. Keeping in mind the long-term vision of our project, desired requirements are elucidated in Table 2 (functional) and Table 3 (nonfunctional).

Table 1: Functional Requirements (Mandatory)

Functional Requirements	Performance Requirements
M.F.1 Sense and estimate part pose	**M.P.1** Detect given grasp points 90% of the time
M.F.2 Plan arm trajectory and move to desired point	**M.P.2.1** Plan movement inside valid zones 100% of the time **M.P.2.2** Plan desired paths for both arms within 4.4 +/- 3.4 seconds
M.F.3 Pick Up and Manipulate Part	**M.P.3** Pick up and manipulate the part successfully 4 out of 5 times
M.F.4 Perform 3D Reconstruction	**M.P.4.1** 90% completeness with accuracy < 3cm for all points **M.P.4.2** Construct within 10 minutes **M.P.4.3** Outliers: Hausdorff distance < 5cm, RMSE < 5cm
M.F.5 Compare to ground truth and calculate differences in surface contours	**M.P.5** Robust to outliers 95% of the time
M.F.6 Avoid causing surface damage to samples	**M.P.6** No surface change to part after manipulation 80% of the time

Table 2: Functional Requirements (Desired)

Functional Requirements	Performance Requirements
D.F.1 Render 3D reconstruction in real time	**D.P.1** Display 30 frames per second
D.F.2 Handle irregular shape objects	**D.P.2** Successfully grasp object 80% of the time
D.F.3 Optimize 3D Reconstruction	**D.P.3** Target millimeter level accuracy

Table 3: Non-Functional Requirements

Requirements
Be robust to environmental changes
Incorporate safety measures
Be reliable

4 Functional Architecture

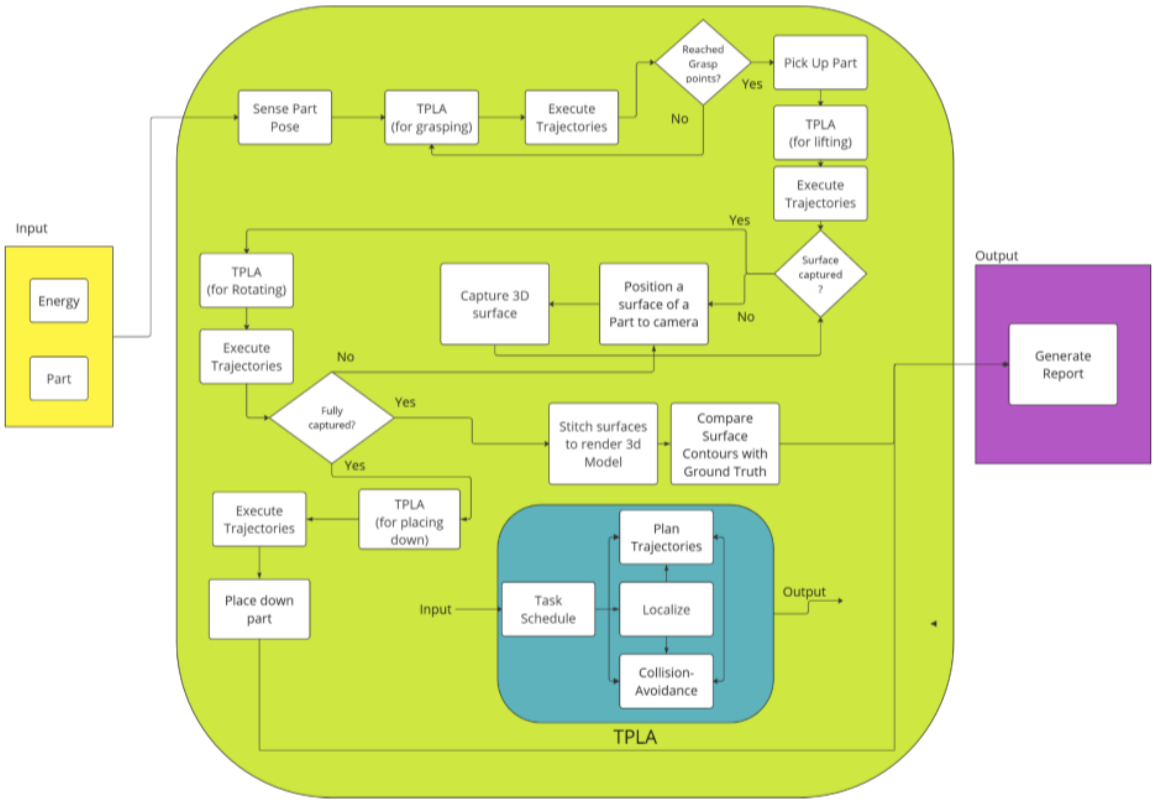


Figure 3: Functional Architecture

The functional architecture represents the high-level functions that our system will perform. It also represents the flow of information from the inputs, through various functional blocks that lead to the outputs of the system. The functional architecture is depicted in Figure 3.

4.1 System Overview

The system receives two primary inputs: material/part input and energy input. The material input is a part produced via the Wire Arc Additive Manufacturing (WAAM) process. The system performs a sequence of operations to capture and validate the 3D surface model of the part.

4.1.1 Material Input: WAAM Part

The part is manufactured using the Wire Arc Additive Manufacturing (WAAM) process. This process involves additive deposition of material, typically metal, layer by layer. The output is a near-net-shape part that requires precise surface characterization.

4.1.2 Pose Estimation and Localization

Upon receiving the part, the system perceives its pose and localizes it accurately on the workbench. This involves using a combination of vision-based sensors and model-based pose estimation algorithms to identify the part's orientation and position.

4.1.3 Manipulator Control and Part Grasping

A bimanual manipulator system is employed for handling the part. The system:

- Localizes the positions of both manipulator arms.
- Plans collision-free trajectories to safely reach and grasp the part.
- Executes a grasping strategy designed to securely hold parts of varying geometry and weight.

4.1.4 3D Surface Capture

Once the part is successfully grasped, the arms lift and position the part in front of a dedicated depth sensing unit. The sensors then perform a 3D surface scan to capture detailed point clouds of the exposed surfaces.

4.1.5 Surface Coverage Checkpoint

After an initial scan, the system enters a checkpoint phase where it verifies if the entire surface of the part has been adequately captured:

- The system evaluates the completeness of the point cloud.
- If any regions are found to be missing, a deterministic policy is applied to reorient the part.
- Additional scans are performed by manipulating the part into new orientations until full surface coverage is achieved.

4.1.6 Placement and 3D Model Reconstruction

Upon completion of surface capture, the system:

- Returns the part back onto the workbench.
- Processes the multiple point clouds to stitch them together into a unified 3D model using surface reconstruction techniques.

4.1.7 Surface Validation Report Generation

Finally, the reconstructed 3D model is compared to a ground truth model:

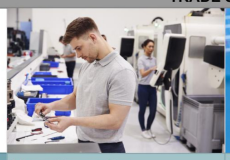
- Deviations in surface contours are analyzed.
- A report is generated detailing the comparison, including quantitative metrics (e.g., surface deviation, missing regions).

5 Trade Studies

5.1 System-Level Trade Study

A comprehensive trade study was conducted to evaluate the most suitable inspection system for our needs. The study compared our proposed system with the most commonly used systems in the industry, including Manual Inspection, Fixed Sensor Systems, Conveyor-Based Systems, and Multi-Manipulator Systems. The assessment utilized a weighted scoring framework based on key functional and non-functional requirements, both mandatory and desirable, alongside some performance evaluation criteria. This methodology ensured the evaluation aligned with the system's operational goals and application needs. The criteria used for evaluation included the ability of the system to operate autonomously, the capacity to handle larger, heavy items efficiently (i.e., Pick up and Move Large Objects), and the provision of detailed and accurate data on the inspection process, supporting downstream analysis and decision-making. Other criteria captured broader performance metrics and practical considerations to evaluate the system. Manual Inspection demonstrated low initial setup cost and high adaptability, safety, and dexterity. However, it scored poorly on automation, speed, and reliability, leading to high operational and maintenance costs, making it less viable for long-term use. The Fixed Sensor System offered moderate initial and maintenance costs, with strong performance in precision and data collection due to static visual inspection. Its limitations included poor adaptability and scalability, and an inability to handle objects autonomously, resulting in decreased throughput.

Table 4: System Trade Study

TRADE STUDY - SYSTEM LEVEL					
					
	Specs	Manual Inspection	Fixed Sensor System	Multiple Manipulator system	Conveyor-based System
	Initial Cost	1400 (training cost)	50000	100000	75000
	cost of operation	50000	10000	7000	20000
	Maintenance Cost	10000 (benefits & insurance)	5000	7000	15000
CRITERIA	**WEIGHT(100)**	**VALUES (1-5)**			
Functional Requirements					
Operate Autonomously	9.68	1	3	5	3
Pick up and Move Large Object	9.68	5	1	4	1
Data Collection and Reporting	9.68	3	4	5	4
Performance Requirements and Non Functional Requirements					
Speed and Throughput	9.68	2	1	5	5
Accuracy and Precision	9.68	3	4	4	4
Reliabiltiy	6.45	5	2	3	2
Repeatability (Objectivity)	9.68	3	5	5	5
Adaptability	3.23	5	1	3	1
Cost of Operation	6.45	1	3	4	2
Maintenance Cost	1.61	3	5	4	2
Initial Setup Cost	1.61	5	3	1	2
Safety	6.45	5	4	3	3
Scalability	6.45	2	3	5	4
Error (Missed defects)	9.68	2	3	4	2
Weighted Average		2.97	2.97	4.24	3.13

The Multiple Manipulator System excelled in autonomy, object handling, throughput, and precision. It was highly reliable and repeatable, with scalable operations, although it came with high

initial setup costs and moderate maintenance expenses, and posed challenges in integrating adaptability. The Conveyor-Based System provided high speed, throughput, scalability, and good data collection capabilities, with moderate setup and operational costs. However, its inability to handle objects autonomously and moderate scores in reliability and safety limited its adaptability. In conclusion, the weighted average scores indicated that the Multiple Manipulator System (4.24) outperformed other systems due to its strong functional capabilities and superior performance metrics. Despite its higher initial cost, the advantages in automation, precision, and scalability justified the investment. The Conveyor-Based System (3.13) ranked second, excelling in speed and scalability but lacking in adaptability and safety.

5.2 Sub-System Trade Studies

5.2.1 Sensing System Trade Studies

A comprehensive trade study was conducted to determine the most appropriate sensing subsystem for environment sensing and 3D reconstruction. The study evaluated four sensor modalities: Passive Stereo, Time of Flight (ToF), Active Stereo (Infrared Projection), and LiDAR, using a weighted scoring framework aligned with key functional and non-functional requirements.

The evaluation considered three primary criteria: Object Detection, assessing the sensor's reliability in identifying objects under varying conditions; Localization, measuring the accuracy of positional data critical for environment sensing and precise placement; and 3D Scanning, evaluating the richness and completeness of reconstructed geometry for detailed modeling tasks. Broader performance factors such as cost, compatibility, computational load, and operational range were also included.

Table 5: Sensor Subsystem Trade Study

TRADE STUDY - SENSING SUBSYSTEM						
						
		Sensor Types	Stereo (Passive Stereo)	ToF	IR Projection (Active Stereo)	LiDAR
		Specs				
		Reference Model	Stereolabs ZED Mini	Azure Kinect	Intel RealSense D435i	Velodyne
		Output	RGB + Depth Map	RGB + Depth Map + IMU	RGB + Depth Map + IMU	Point Cloud
Value Ratings		Computation Cost for Depth	very high	low to medium	medium	low
1: inadequate		Range	0.1m to 15m	WFOV unbinned 0.25-2.21m WFOV binned 0.25-2.88m	0.3-3m	200m
2: Tolerable		Resolution	2x (2208x1242) @15fps 2x(672x376) @100fps	WFOV unbinned 1024x1024 WFOV binned 512x512	Depth output: 1280 x 720 RGB output: 1920 x 1080	16 - 32 channels
3: Adequte		Field of View	102°(H) x 57°(V) x 118°(D)	90°x59° 90°x74.3°	Depth: 87° × 58° RGB: 69° × 42°	360° × 40°
4: Good		FPS/ Scan rate	100 FPS	30 FPS	Depth: 90FPS RGB: 30 FPS	20 FPS
5: Excellent		Cost	$399	$399	$334	$5,000
Environment	**Criteria**	**Weight (100%)**	**Values (1-10)**			
	Functional Requirements					
	Object detection	10%	8	8	8	9
	Localization	20%	6	6	6	9
	3D scanning	5%	8	6	8	10
	Performance Requirements and Non Functional Requirements					
	Resolution	22.5	9	6	8	9
	Field of View	12%	8	8	7	8
	Computational requirement	5%	4	7	6	8
	Reflectance issues	5%	5	8	8	8
	Cost	10%	9	9	10	1
	Compatibility	10%	8	7	9	3
	Weighted Avg		8.939	6.043	7.986	8.933
Object	**Criteria**	**Weight (100%)**	**Values (1-10)**			
	Functional Requirements					
	Object detection	10%	7	8	8	9
	Localization	3%	6	6	6	9
	3D scanning	25%	6	6	8	10
	Performance Requirements and Non Functional Requirements					
	Resolution	25	9	6	8	9
	Field of View	16%	8	8	7	8
	Computational requirement	5%	4	7	6	8
	Reflectance issues	5%	5	8	8	8
	Cost	1%	9	9	10	1
	Compatibility	10%	8	7	9	3
	Weighted Avg		8.932	6.031	7.992	8.973

Passive Stereo provided good resolution and a wide field of view at a relatively low cost, but required significantly higher computational effort for depth estimation and offered limited effective range. Because of these practical constraints, especially for consistent depth generation, we transitioned from Active Stereo to Passive Stereo for our system. Time of Flight (ToF) showed moderate performance across criteria with lower computational burden but lacked the resolution and range necessary for high-fidelity reconstruction tasks. Active Stereo, while offering strong object detection and scanning capabilities, carried limitations in field of view and range that made it less attractive once we reassessed compatibility and cost. LiDAR consistently outperformed all other modalities in localization accuracy, 3D scanning quality, and long-range sensing, generating dense and precise point clouds ideal for reconstruction.

While LiDAR (8.973) was objectively the best-performing sensor, particularly for 3D reconstruction, its cost placed it outside the project's budget constraints. The updated analysis therefore identified Passive Stereo (replacing our earlier choice of Active Stereo) as the most balanced and feasible option within available resources, despite LiDAR remaining the preferred choice if budget were not a limitation.

6 Cyber-Physical Architecture

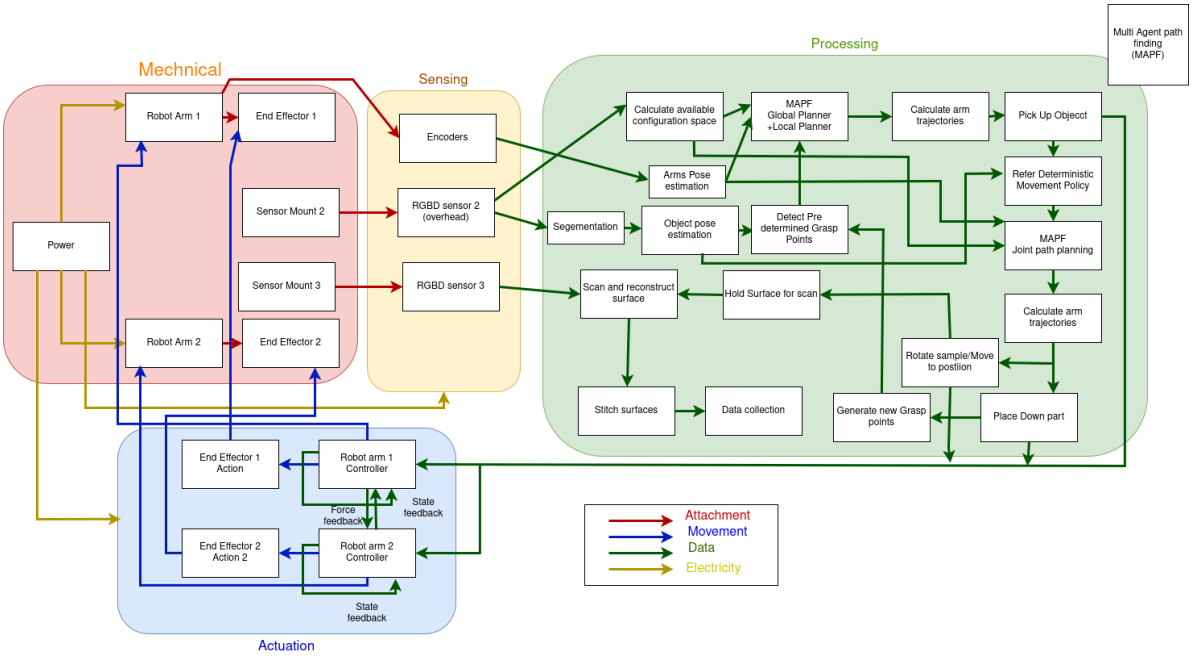


Figure 4: Cyber-Physical Architecture

The cyber-physical architecture organizes the system into four primary units: the mechanical unit, the sensing unit, the actuation unit, and the processing unit. Each unit plays a critical role in enabling the perception, manipulation, and validation tasks required for handling WAAM-produced parts. The architecture is depicted in Figure 4.

6.1 Mechanical Unit

The mechanical unit forms the physical foundation of the system. It consists of two 7-degree-of-freedom (DOF) Kinova robotic arms, each equipped with an end effector designed for grasping and manipulating the WAAM part. Additionally, two sensor mounts are installed within the workspace to securely hold the sensing devices required for perception tasks.

This unit is responsible for all physical interactions with the part, including picking it up from the workbench, holding it in front of sensors for 3D scanning, rotating it for complete surface coverage, and finally placing it back after inspection.

6.2 Sensing Unit

The sensing unit enables perception of the part and the workspace environment. It includes:

- An overhead RGB-D sensor mounted above the workspace for object segmentation, pose estimation, and grasp point detection.
- A secondary RGB-D sensor mounted on a fixed sensor mount within the workspace for high-resolution 3D surface scanning.

- Encoders within the robotic arms to provide precise joint state and pose feedback for manipulation tasks.

The sensing unit allows the system to segment the WAAM part from its surroundings, determine its 6-DOF pose, identify suitable grasp points, and reconstruct the 3D geometry necessary for validation.

6.3 Actuation Unit

The actuation unit governs the control and execution of arm movements. It includes:

- Dedicated state feedback controllers for each robotic arm.
- Force feedback systems to enable compliant and adaptive manipulation.
- Shared feedback mechanisms to coordinate bimanual tasks between both arms.

The actuation unit ensures precise, collision-free, and synchronized motions, enabling safe and efficient part handling and positioning throughout the scanning and validation process.

6.4 Processing Unit

The processing unit acts as the computational core of the system. Its responsibilities include:

- Segmenting the WAAM part from sensor data and estimating its pose relative to the workspace.
- Calculating the available configuration space for manipulation and detecting pre-determined grasp points.
- Planning collision-free arm trajectories using coordinated dual-arm Cartesian planning, treating both arms as a unified kinematic chain, together with standard MoveIt local planners.
- Referring to a deterministic movement policy to orient the part optimally for scanning, leveraging known object geometries.
- Capturing 3D surface scans, evaluating surface coverage, and generating new grasp points and trajectories if additional scans are required.
- Stitching collected point clouds into a unified 3D model.
- Comparing the reconstructed model to a ground truth CAD model and generating a surface validation report.

The processing unit integrates perception, motion planning, control, and data analysis to fully automate the surface validation workflow.

7 Current System Status

7.1 Overall System Depiction

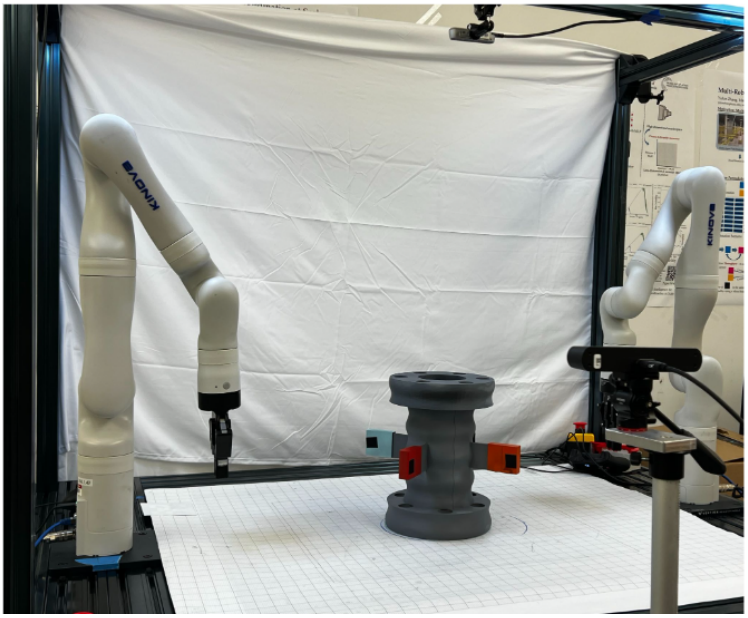


Figure 5: Overall System Depiction

The overall system comprises of two Kinova Gen3 7-Dof robot arms with a Robotiq 2f-85 gripper attached on each arm. Each arm comes with a safety E-stop and a joystick for teleoperation. These arms are mounted on a custom vention table. On the table, there is an Intel RealSense camera mounted on the top bar right above the object being manipulated. This is used for the pose estimation of the object which will feed into the manipulation subsystem. Once the arms locate the grasp position and orientation from the pose estimation pipeline, they will go to the grasp, lift, and manipulate the object to show all sides of the object to the Zed stereo camera.

The Zed camera will be positioned on the side of the system facing the object being manipulated. As the arms maneuver the object, it will collect points cloud data to perform a reconstruction. Finally, the GUI that displays the 3D reconstruction will also visualize a 3D comparison from the desired CAD model to the reconstructed model.

7.2 Subsystem Descriptions

7.2.1 Perception: Pose Estimation

The perception subsystem is responsible for providing accurate, real-time 3D pose estimates of the object for downstream manipulation and 3D reconstruction tasks. This cycle focused on redesigning the handle geometry, improving detection reliability through a custom YOLOv11 model, refining the projection and yaw estimation logic, and validating the system through extensive integration with the manipulation pipeline.

Handle Redesign and Visual Encoding: To minimize class ambiguity and reduce false detections, the object's handle assembly was redesigned with four distinct, high-contrast colors. Each color corresponds to a unique semantic class: *handle_o*, *handle_r*, *handle_c*, and *handles*. This visual encoding

provides clear separation between handle elements and introduces strong geometric constraints that directly improve stability in orientation estimation.

Multi-Handle Detection Using YOLOv11: A custom YOLOv11 model was trained on an expanded dataset containing diverse lighting conditions, object poses, and workspace configurations. All four handle classes were annotated and used in training Fig6. The resulting detector provides high-confidence, high-frequency predictions and significantly reduces false positives, label swapping, and frame-to-frame jitter. This serves as the foundation for reliable 2D-to-3D mapping.

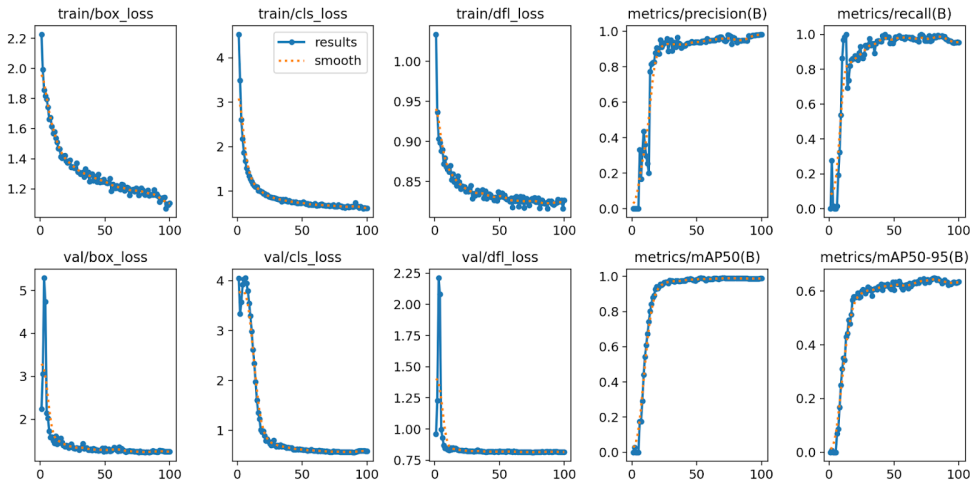


Figure 6: YOLOv11 loss curves

World-Frame Projection and Pose Computation: Using the calibrated camera parameters, 2D detections are projected into the world frame as seen in Fig7. The relative geometry among the four detected handles enables robust computation of:

- The 3D center position of the object,
- A stable yaw orientation estimate,
- Redundant pose cues resilient to transient detection noise.

The multi-handle configuration eliminates prior orientation ambiguities and produces a smooth, low-variance pose suitable for closed-loop control.

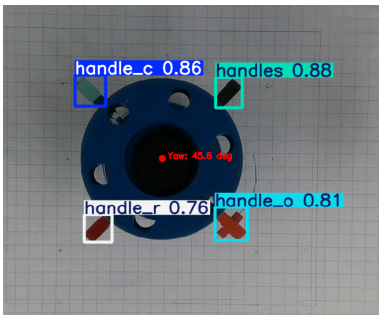


Figure 7: Detected Center and Yaw

Real-Time Pose Publishing The perception subsystem publishes the final pose (x, y, z, θ) as a `PoseStamped` message at real-time rates. This continuous stream of pose updates feeds directly into the manipulation stack and ensures that grasping, alignment, and reorientation behaviors remain synchronized with the current object state as seen in Fig8

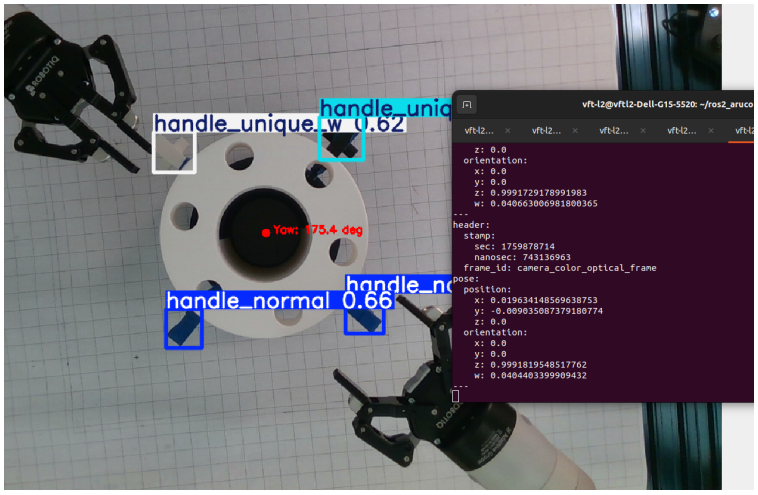


Figure 8: real-time message publishing

Integration and Stress Testing: The updated perception pipeline was validated through repeated hardware trials involving full grasp–lift–place cycles. Metrics such as detection stability, yaw consistency, and inference throughput were recorded. The pose estimates remained stable across long-duration runs, enabling smooth arm trajectories and consistent reorientation behavior. This reliability is essential for downstream tasks such as the final 3D reconstruction stage.

7.2.2 Perception: 3D Reconstruction

The 3D reconstruction subsystem was developed to enable scene understanding from multiple viewpoints using a single, static camera observing a moving object. The core idea was validated through a **turntable reconstruction** proof of concept for SVD, establishing that accurate 3D reconstructions are feasible even when the object, not the camera, is in motion. This approach relies on capturing a sufficient number of diverse views to resolve occlusions and ensure geometric completeness. Finally for FVD, this system was fully integrated with the manipulation subsystem with added

refinements.

A reliable and validated pipeline was developed, as seen in Figure 9, composed of:

- **World Frame Calibration** using Aruco markers for transformations between camera and world frame.
- **Object Motion Transformations** expressed in the world frame and calculated from the inverse kinematics of the dual arm chain
- **Frame-wise Segmentation** using the SAM2 model, propagated across video frames to reduce manual effort.
- **Depth-to-3D Projection** using camera intrinsics, depth maps, and object transformations.
- **Camera-to-World Transformation** using pre-calibrated extrinsic matrices.
- **Final Aggregation** of world-space 3D points from both rotations into a unified object-space point cloud.
- **Ground Truth Overlay** of reconstructed point cloud and ground truth point cloud for visual comparison
- **Metrics** evaluation and requirements verification

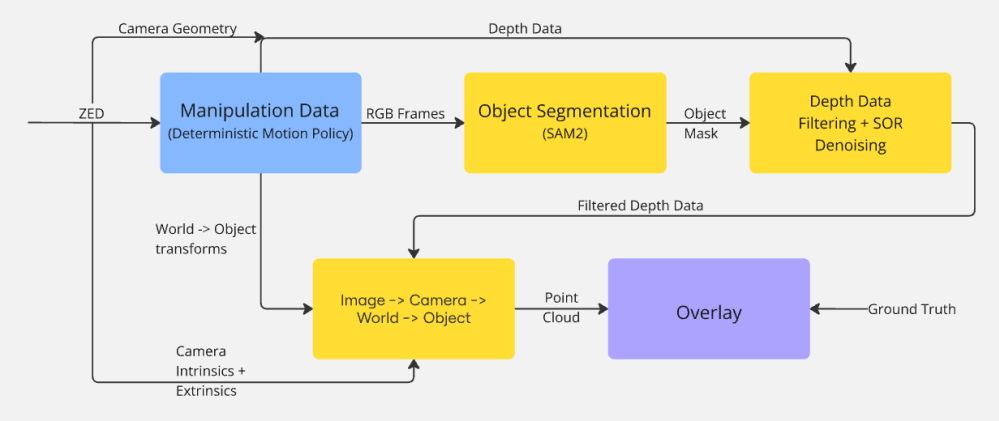


Figure 9: 3D Reconstruction Pipeline

Implementation:

- **Data Acquisition:** During the object manipulation policy, synchronized RGB-D data are collected for two complete 360-degree rotations of the object. The relative transformation between these two motions is assumed to be a 90° rotation about the object's yaw axis, which later assists in merging complementary viewpoints. Depth maps are precomputed and stored in a single `depth.npy` file aligned with the RGB frame sequence, containing per-pixel depth values in the 100–1000 mm range. The object's motion is estimated from the inverse kinematic (IK) chains of the two robot arms, assuming the object center lies midway between the end effectors.

- **Segmentation**: Segmentation is performed using the SAM 2 model with both point-based and box-based prompting. Temporal consistency is maintained using the custom video propagator, ensuring stable object isolation across the entire sequence.
- **Projection to 3D**: For each mask, a partial point cloud is generated by projecting pixel coordinates into the camera frame using the intrinsic matrix K. The `mask2cam` function attaches depth values and lifts each pixel into 3D. These points are then transformed into the world frame via the calibrated extrinsic parameters obtained from Aruco markers.
- **Denoising**: Each world-frame partial point cloud undergoes voxel downsampling to stabilize downstream alignment. An initial ICP pass is applied between temporally adjacent partial clouds to refine the raw IK-derived motion, reducing drift and correcting local inconsistencies. After pairwise refinement, Spatial Outlier Removal (SOR) and statistical filtering remove noise and isolated points, yielding clean partial clouds ready for coordinated fusion.
- **Motion Transformation**: Using synchronized timestamps, the refined motion estimates are applied to express every denoised partial point cloud in a unified object coordinate frame. The first recorded object pose in the sequence serves as this canonical frame, since manipulation demonstrations always begin from the same initial alignment.
- **Reconstruction**: All object-frame partial clouds are fused to produce the final 3D reconstruction. A final multi-view ICP alignment is performed to ensure global consistency across all viewpoints. The consolidated reconstruction is exported as a PLY file.
- **Visualizations**: Visualizations are produced throughout the pipeline for verification. Early-stage tools such as `show_box` and `show_points` allow inspection of segmentation quality, while the fused point cloud is rendered for final assessment. Precision statistics are computed to quantify reconstruction performance.

Metrics: The reconstructed point cloud is aligned to the ground-truth mesh using CloudCompare for qualitative inspection. Quantitative evaluation uses the following metrics:

- **Completeness**: The percentage of ground-truth points that have a corresponding reconstructed point within 1 cm.
- **Accuracy**: The percentage of reconstructed points that lie within 3 cm of the nearest ground-truth point.
- **Hausdorff Distance**: The maximum bidirectional nearest-neighbor distance between the reconstruction and the ground truth, kept below 5 cm.
- **RMSE**: The root-mean-square error across corresponding points, with values below 5 cm indicating a smooth and consistent reconstruction.

This modular and automated reconstruction subsystem enables efficient and accurate generation of object-level 3D point clouds from RGB-D video, supporting downstream tasks such as manipulation planning and inspection. The final reconstruction of the test objects is shown in Fig. 10.

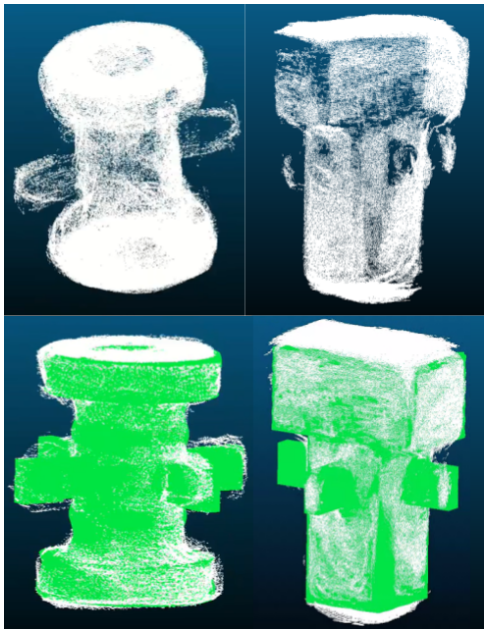


Figure 10: 3D Reconstruction of Test Objects

7.2.3 Manipulation

The manipulation subsystem handles the smooth movement of the object so that it can be reconstructed. The implementation centers around a finite-state machine (FSM) architecture integrated with the MoveIt motion planning framework, enabling coordinated planning and execution of a dual arm manipulation sequence. We used Rviz to visualize the planned trajectories to ensure that it is dynamically feasible before executing the plans on real hardware. Some of the key concepts of implementation are the following:

Motion Planning: The subsystem uses the MoveIt planning interface by defining 3 planning groups: left arm, right arm, and both arms. The KDL kinematics solver plugin is used for performing inverse kinematics on the left and right arm based on the goal end effector pose. Then, the goal joint positions are set as the target pose for the dual arm planning group. We treat the dual arms as a unified kinematic chain to plan a trajectory to the goal in Cartesian space. From the current end-effector to the goal-pose, we calculate linear waypoints of 5 mm with Spherical Linear Interpolation (SLERP) for maintaining a viable position and orientation of the end-effectors.

Collision Management: We implemented collision avoidance by creating detailed collision object representations of the test objects with precise dimensions including walls and bottom surfaces. When grasping occurs, the system uses the AttachedCollisionObject functionality to properly model the physical connection between the robot and the manipulated object. The implementation defines specific allowed touch links to prevent false-positive collision detection between the object and the gripper components during manipulation tasks. We also fully define the Vention table on which the arms are mounted because the overhead bars of the table pose a significant limitation for the arm movements.

Finite State Machine: The FSM framework controls the dual-arm manipulation sequence through a hierarchical set of states. The system supports three object types: bins, cylinders with spokes, and T-bars, each with object-specific grasp strategies and collision models. A visual of this is shown in Figure 11. In each planning state, the operator can manually replan trajectories, providing critical visual feedback in RViz for verification of complex movements before hardware execution. The FSM also performs automatic replanning when goals are not reached within the allocated time frame (initial 5 seconds, with 5-second increments per retry).

The manipulation sequence consists of four major phases:

1. **Initialization:** The system receives object pose data from the perception pipeline and spawns the corresponding collision object in the planning scene. Dual-arm grasp points are calculated based on the object type and detected pose (x, y coordinates and yaw angle). The arms then plan and execute trajectories to approach positions above the object, followed by opening the grippers.

2. **Grasp and Lift:** The end effectors linearly descend to the grasp positions and close the grippers. The object is attached to the planning scene, making both arms collision-aware of the grasped object. The system then lifts the object 35cm vertically from the grasp position.

3. **Centering and 3D Capture:** An adaptive centering algorithm moves the object to position (0, 0) using incremental steps sized based on the current offset distance. The arms are then straightened and aligned for 360-degree rotation. During rotation, the system performs 16 steps of 22.5° each, rotating the left and right wrists in opposite directions while publishing a capture flag at 100Hz to synchronize with external 3D scanning systems.

4. **Placement and Return:** The object is moved to the placement position in two stages: first horizontally to the target XY coordinates at the current height, then vertically down to the placement height. The grippers open to release the object, which is detached from the planning scene. After retracting from the placement position, the system can either return to home or perform a secondary manipulation cycle using alternate grasp points. The return-to-home planning is the only time that search-based motion planners (RRTConnect, RRTstar, PRM) are used with extended timeouts to ensure robust trajectory generation from complex configurations.

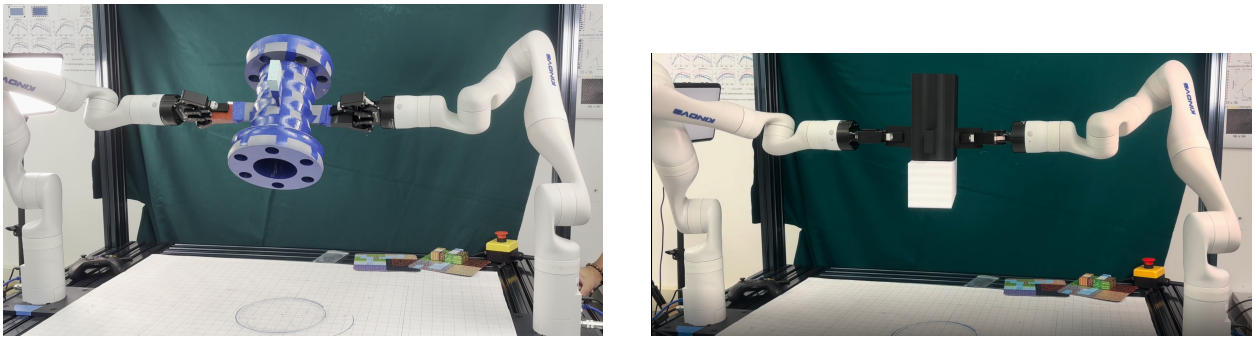


Figure 11: Kinova arms manipulating cylinder test object (left) and t-bar stretch goal object (right)

7.3 Modeling, Analysis and Testing

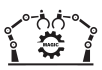
The reconstruction pipeline was subjected to a variety of modeling and parameter studies to optimize both quality and runtime performance. Table 6 summarizes key components, tested parameters, and insights gained.

Table 6: Reconstruction Pipeline Component Testing

Component	Parameter	Variations Tried	Observation
Data Collection	FPS	15, 30, 60	Higher FPS produced smoother motion but resulted in larger file sizes
Data Collection	Resolution	720p, 1080p, 2K	Higher resolution provided better detail but increased processing time
Data Collection	Recording Duration	25s, 60s, 100s	Longer sequences improved reconstruction density but also increased preprocessing time
Data Collection	Depth Estimation	Stereo Depth from ZED 2i, Monocular depth from Depth Anything	Monocular Depth Estimation fails to capture the geometry accurately due to lack of features but the depth maps are clean. Stereo depth performs better but is very noisy.
Post-processing	Denoising Method	Median filter, Region Growing, Statistical Outlier Removal (SOR)	Region Growing removed dense noise; SOR better eliminated isolated outliers; Median filtering used between frames
Post-processing	Alignment Method	Direct Transformations, Transformations + General ICP, Transformations + RGB ICP	Transformations + General ICP performed best since it was easier to align geometric features as compares to RGB features for a solid object. Plain transformations had errors due to jitters and slipping during manipulation

From this analysis, the optimal configuration was found to be:

- **15 FPS at 1080p for 60 seconds** — fastest pipeline with minimal loss in reconstruction quality.
- **Direct Stereo Depth + Denoising** — Since the underlying geometry is captured more accurately, stereo depth is used with denoising.
- **Voxel Downsampling + SOR** — best denoising strategy for our calibrated static camera setup.
- **Motion Transformations + General ICP** — most accurate geometric alignment per frame and for alignment of reconstructions from both rotations.



We ran over 40 total run of the manipulation subsystem to ensure robustness and reliability of our planning strategy. Table 7 shows the results of our tests on the cylinder object for the first 10 runs, and Table 8 shows 10 runs of the t-bar object. For each test, we varied the initial position of the object to validate that we can check pose estimation and grasping calculations. We had 80% success rate for the cylinder object and 90% for the T-bar object.

Table 7: Cylinder Test Runs

Run	1	2	3	4	5	6	7	8	9	10
x (cm)	0	-2	0	0	-2	3	-4	-4	3	-2
y (cm)	-2	5	7	4	5	-3	-5	4	5	3
yaw	45.8	48.5	50.7	45.5	51.2	55.2	56.5	40.4	43	47.9
Pass/Fail	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Fail	Fail	Pass
Issue	/	/	/	/	/	/	/	Fail to lift	Fail to lift	/

Table 8: T-bar Test Runs

Run	1	2	3	4	5	6	8	7	9	10
x (cm)	0	1	5	0	0	0	0	0	-7	5
y (cm)	0	-3	-11	0	4	-2	4	7	-9	-11
yaw	60	50	46	35	45.5	45.8	45.5	50.7	41.8	60
Pass/Fail	Pass	Pass	Fail	Pass	Pass	Pass	Pass	Pass	Pass	Pass
Issue	/	/	Fail to lift	/	/	/	/	/	/	/

7.4 FVD Performance Evaluation

The set of system capabilities that were demonstrated in the FVD and FVD Encore are shown in Table 9. Our robot performed the set of tasks while meeting the desired system requirements.

Table 9: Targeted Performance Requirements for Fall Semester

Procedure	**Success Criteria**	**Requirements**
Move object around the workspace with accurate x, y position and yaw angle. Spawn it in Rviz before calculating grasp points within 2 cm translation and 2° rotation error.	Achieved **100% grasp point detection** within the camera's field of view with **< 0.5 cm translation and < 1° rotation error**. There is **less than 10 ms latency** between pose estimation pipeline and manipulation system.	**M.P.1**
Run motion policy several times to ensure that the policy plans movement inside valid zones 100% of the time. The desired plan should be calculated within 4 seconds.	Motion policy on real hardware was run more than 20 times and only plans within valid zones. The planner will not allow the plan to execute unless there are **no collisions and the movements are viable for the whole trajectory**. The planner takes **20-50 ms to plan**.	**M.P.2.1, M.P.2.2**

Run motion policy at least 10 times on the Kinova arms. For each run, change the position of the test object and stretch goal object. The arms should be able to pick up, manipulate, and place down the object reliably with no damage to the object.	Motion policy on real hardware was run more than 20 times on each object and is reliable. It passed at **80% success rate for the cylinder object and 90% for the T-bar object**. With all of those runs, the object was **not damaged** which met our requirement of not causing surface damage 80% of the time.	**M.P.3.1, M.P.6**
3D reconstruction using RGBD data, motion transformatiosn from IK, and post processing	Completeness of **99.79%** (>90%) and accuracy of **2.8 cm** (< 3 cm). Haudorff distance of **4.8 cm** and RMSE of **3.2 cm**, both below the 5 cm threshold	**M.P.4.1**, **M.P.4.3**
Full reconstruction pipeline runs from image capture to point cloud generation	Completes data collection and preprocessing in 4 minutes, reconstruction within **4-5 minutes**, below the 10-minute threshold	**M.P.4.2**

7.5 Strong/Weak Points

Our progress through the semester led to a successful demonstration during FVD and FVD Encore. While there are several strong areas of our system, there are also areas that need improvement. The strong and weak points of our system are highlighted below:

Strengths:

- **Achieved robust manipulation policy for multiple objects:** We successfully implemented and executed a manipulation strategy that works on a bin, cylinder test object, and T-bar stretch goal object. The manipulation is reliable, robust, and fast for all objects.
- **Minimum Sim2Real gap in motion policy:** Our motion planning and control strategies translated well from simulation to the real-world setup, requiring minimal tuning, which indicates a low sim-to-real gap.
- **Tested and validated 3D reconstruction pipeline with time optimization and denoising:** The 3D reconstruction pipeline was thoroughly validated and optimized, achieving fast execution times and incorporating denoising and alignment steps to enhance mesh quality.

Weaknesses:

- **Constrained workspace:** The physical and operational workspace of the system is limited, reducing the range of object sizes, manipulation directions, and configurations we can robustly handle.
- **Decoupled data collection and processing pipeline:** Due to incompatibility of the ZED SDK with our Linux-based robotic control system, the RGB-D data collection was offloaded to a Windows laptop. This required decoupling data acquisition and processing into separate steps, preventing a seamless, real-time end-to-end pipeline.
- **Featureless test object:** The object used for evaluation lacks surface texture and geometric distinctiveness. This significantly degrades stereo depth estimation quality, as depth from stereo cameras relies on finding correspondences between image features. The lack of features limits reliable disparity computation, impacting both depth accuracy and reconstruction fidelity.

- **3D data from the current setup is not real:** Our setup uses a stereo RGB camera for depth, which computes disparity-based depth rather than measuring true physical distances. Compared to active depth sensors like ToF or structured light, this approach is inherently less reliable in low-texture scenes and introduces noise and scaling inaccuracies. As a result, the generated depth maps and meshes are not "real" and lack the precision of scanner-based methods.

8 Project Management

8.a Schedule

		January			February				March			April			August	September				October				November		
		Wk 1	Wk 2	Wk 3	Wk 4	Wk 5	Wk 6	Wk 7	Wk 8	Wk 9	Wk 10	Wk 11	Wk 12	Wk 13	Wk 1	Wk 2	Wk 3	Wk 4	Wk 5	Wk 6	Wk 7	Wk 8	Wk 9	Wk 10	Wk 11	FVD
TASK TITLE	DURATION	JAN 13-17	JAN 20-24	JAN 27-31	FEB 3-7	FEB 10-14	FEB 17-21	FEB 24-28	MAR 10-14	MAR 17-21	MAR 24-28	APR 31-4	APR 7-11	APR 14-18	AUG 25-29	SEP 1-5	SEP 8-12	SEP 15-19	SEP 22-26	OCT 29-3	OCT 6-10	OCT 20-24	OCT 27-31	NOV 3-7	NOV 10-14	NOV 17-21
Hardware																										
Camera Mounts																										
Design	1																									
Fabricate	2																									
Interface	1																									
Perception																										
3D reconstruction																										
Calibration	1																									
Setup pipeline	1																									
Sensor fusion	3																									
Single face 3d model	3																									
Stitching	5																									
Optimization (reduce runtime)	5																									
Environment Modeling																										
Aruco Marker Position	2																									
Calibration	2																									
Design Algorithm	1																									
Optimization	2																									
Object Pose Estimation																										
Calibration	1																									
Design Algorithm	2																									
Optimization	2																									
Manipulation																										
Controls																										
Arm Movement																										
Set up simulation	1																									
Naive controller single arm	3																									
Naive controller multi arm	3																									
Optimize controller design	5																									
Marker feedback	3																									
Grasping																										
Gripper control	3																									
Tuning	5																									
Pressure Sensor feedback	5																									
Planning																										
Task Scheduling																										
Design Policy (Naive)	5																									
Optimization (Naive)	4																									
Optimize Policy (Non-linear object)	6																									
Multi-Agent Path Finding (MAPF)																										
Global	2		familiar by this week																							
Local	2																									
Post-Processing																										
Ground Truth Comparison	4																									
Generate Report	4																									
Integration																										
Sim2Real	7																									
Testing (Q1)	2																									
Testing (SVD)	2																									
Testing (Q3)	2																									
Testing (FVD)	1																									

Figure 12: Full Schedule and Milestones

Our team faced significant initial challenges that distinguished us from other groups. Unlike teams working with established systems, we had to begin by defining our use case from scratch. Additionally, our entire system was built on ROS2, requiring us to independently configure both MoveIt2 and the Kinova system without the guidance typically available to other teams. The PhD students' expertise was limited to the previous ROS version, leaving us to navigate the setup alone. These foundational tasks consumed the majority of our fall semester. We also invested considerable time evaluating open-source tools, which presented a steep learning curve that we had underestimated.

From these early struggles in fall semester, we learned to allocate more realistic timeframes for research and learning phases in spring. A critical success was our implementation of turntables for perception troubleshooting, which allowed parallel development of manipulation and perception modules without requiring full system integration each time. Most importantly, we significantly increased

our integration timeline. This adjustment proved decisive. We completed our final demonstration on schedule, with the last two weeks dedicated solely to resolving minor alignment issues rather than rushing incomplete components together.

8.b Budget

We have a total budget of $5,000. Our expenditures totaling $1,630.16 (32.6% of allocated funds) have been strategically directed toward establishing a fully functional development and testing environment. The camera system represents our largest investment at $569.00, providing the high-resolution imaging capabilities essential for perception tasks and object recognition. The lab setup components totaled $357.22, encompassing mounting hardware, structural supports, and workspace organization tools necessary for system integration and testing workflows. Display infrastructure constituted a significant portion of our budget, with monitors totaling $309.97. This included a 32-inch 2K primary monitor ($169.99) serving as the main display for development and demonstration purposes, ensuring clear visualization of outputs from the Magic system. A 16-inch portable 2.5K monitor ($129.99) provides critical flexibility between the MRSD and ARCS lab spaces, enabling mobile testing, rapid debugging, and real-time data viewing without workspace constraints. Testing equipment included two rotating turntables totaling $92.98, which enable 360-degree data collection and multi-angle perception testing without requiring full system reconfiguration, directly supporting our parallel development strategy. Test objects, including a collapsible laundry basket, organizational containers, and 3D printed object totaled $150.51 and provide varied geometries for manipulation algorithm validation. Additional purchases included lighting equipment ($49.99) for consistent perception conditions, specialized tape ($57.51) for calibration and marking, and various cables ($42.98) for reliable system connectivity. These acquisitions strategically position the team to efficiently execute perception, manipulation, and integration tasks throughout the project timeline while maintaining substantial budget flexibility for unforeseen technical requirements.

Table 10: Budget: Our expenditure so far at a glance

Part Name	SUM of Total Price
Cable	42.98
Camera	569
Lab Setup	357.22
Light	49.99
Monitor	309.97
Tape	$57.51
Test Object	$150.51
TurnTable	92.98
Grand Total	**1630.16**

8.c Risk Management

Since the fall semester, we identified five major risks and assigned a risk owner for each of them to ensure continuous monitoring and mitigation. These risks were chosen based on their potential to directly impact our project's technical success, timeline, and cost. Our primary focus remains on the risks that could cause major delays to the project schedule or prevent us from achieving successful

demonstrations. Table 11 summarizes the description, impact, type, and mitigation strategy for each risk. Detailed discussions for each risk are provided in Tables 13 through 17.

One of the most critical risks we continue to manage is the risk of robot arm collisions (R-1), which could result in severe hardware damage to the robot, environment, or inspection samples. To mitigate this, manual e-stop procedures are enforced during all testing activities so that any unexpected behavior can be quickly halted. In addition, we use a Cartesian planner that generates predictable, easily interpretable paths, and we review each planned trajectory before execution to ensure the arms are not at risk of colliding with each other, the environment, or the part.

Another significant concern is falling behind schedule due to a steep learning curve (R-2). With the complexity of integrating perception, manipulation, and sample analysis modules, delays can occur if team members struggle to ramp up on new technologies. To address this, we have proactively sought mentorship and early technical guidance from faculty and lab advisors.

Resource availability (R-3) remains a logistical risk, especially given dependencies on specific hardware components or specialized equipment. To reduce this risk, we have identified and prepared alternative resources in advance, ensuring minimal disruption in case primary tools become unavailable.

Additionally, the risk of end effector or test object failure (R-4) is critical, as the inability to perform demonstrations due to broken hardware could severely impact milestone completion. We have preemptively ordered spare parts and alternative test objects to ensure continuity in case of failures.

Lastly, integration challenges between systems (R-5) could introduce substantial delays if discovered late. To mitigate this, we have adopted a continuous integration approach—performing unit tests, validating subsystems independently, and integrating one subsystem at a time with clear validation checkpoints.

While we have not yet encountered catastrophic failures, smaller instances like minor integration difficulties and scheduling pressures have arisen. However, due to active risk management practices, such as proactive communication, early troubleshooting, and flexible task allocation, these risks have been successfully contained. We maintain a living risk register that is reviewed weekly, ensuring that we are responsive to evolving project dynamics.

Table 11: Risk Management

S. No.	Description	Consequence	Consequence Level	Risk Type	Likelihood	Mitigation
R-1	Robot arms collide with itself or other objects	Robot arms/inspection samples/environment outside of table is	5	Technical, Programmatic, Cost	3	During testing, someone is always on the manual e-stop
R-2	Falling behind schedule because of learning curve	Schedule delay	3	Schedule	4	Seek mentorship early
R-3	Resource availability	Might fall behind on schedule	4	Schedule	3	Identify alternative resources as backup
R-4	End effector or test objcet breaks	Cant do demo	5	Technical, Cost	3	Order spares
R-5	Integration issues between subsystems	Schedule delay	5	Schedule	3	Perform unit test and subsystem validation experiments continuously. We will integrate one subsystem at a time

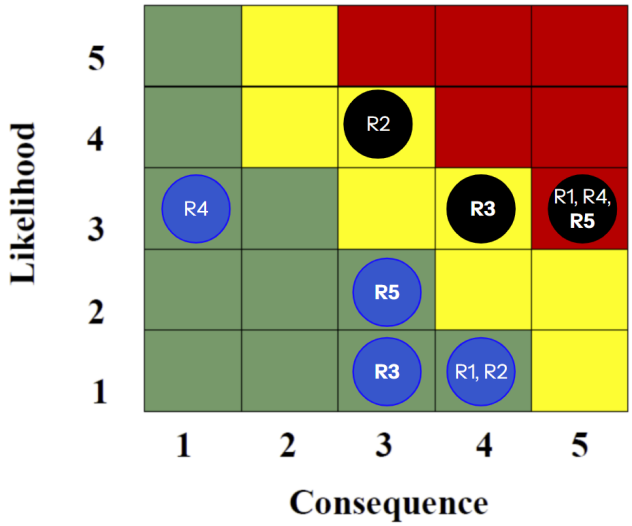


Figure 13: Risk Likelihood-Consequence Table

8.d Risk Tracking

One way we have been actively managing risks throughout the semester is by not only implementing mitigation strategies in real time but also systematically logging their occurrence and resolution. Table 12 captures our tracking methodology. For each risk, we recorded when it was first observed under the ”Occurred yet?” column and documented the mitigation approach taken. Once an effective resolution was achieved, the ”Need of Action still?” column was updated to ”No,” along with a description of the implemented solution to maintain transparency and traceability.

Early logistical risks, such as delays in setting up the robot table or having only one computer available, were quickly addressed by establishing a workspace ahead of testing and enabling remote shared access through SSH and TeamViewer. Technical issues, which formed a majority of the risks, were systematically handled. Calibration challenges with the ZED stereo camera, software integration issues with Kinova MoveIt packages, and noisy depth map outputs were resolved through manual calibration, firmware updates, platform shifts (e.g., switching from MuJoCo to Gazebo for simulation), and software workarounds. Compatibility issues due to library updates, such as the Numpy 2.0 update breaking the perception pipeline, were mitigated by selectively locking software versions and updating the requirements documentation.

Hardware reliability concerns were also proactively managed, such as purchasing a spare bin after repeated testing deformed the original container. Safety risks, like the robot colliding with unidentified objects, were countered by always having an operator stationed at the emergency stop during active testing and demonstrations.

Overall, this structured logging and immediate mitigation approach enabled us to stay agile, minimize the impact of unforeseen issues, and maintain steady progress across all subsystems. Every major risk identified was either resolved or brought under control effectively, demonstrating the robustness of our risk management process. Only risks with sufficient documentation and clear resolution pathways have been included to ensure clarity and focus in our reporting.

Table 12: Risk Tracking

S. No.	Risk No.	Issue	Owner	Occured yet?	Need of Action still?	Resolution
1	R-3	Robot table/workstation comes in later, so set up will hold testing and measurements for a bit	All	Yes	No	We set up new table early before we started any experiments
2	R-3	Only have one computer	Kailash	Yes	No	We set up SSH and Teamviewer so more than one person can use the same time
3	R-2	ZED Calibration Issues	Shreya	Yes	No	Manually calibrate
4	R-2	Kinova Gen3 readme for MoveIt is not accurate	Megan	Yes	No	Debugged line by line
5	R-2	No current tools to easily convert xml to urdf file	Emma	Yes	No	Manually convert xml to URDF file
6	R-2	Error in existing Kinova Official Repository	Megan	Yes	No	Remove all gripper component from URDF
7	R-2	OpenCV functions for stereo rectification were unable to give a comprehensive disparity map	Kartik	Yes	No	Switched to ROS wrapper which needs minimum support from SDK
8	R-2	Self Calibration failed	Shreya	Yes	No	Updated firmware for camera
9	R-2	Losing depth frames	Shreya	Yes	No	Turned off auto exposure for camera. Will need to be caibrated for specific lighting and those conditions will have to be maintained
10	R-2	MoveIt2 official resources are unreliable	Emma	Yes	No	Did not use the official resources (e.g. MoveIt SetUp Assist) and create config file manually
11	R-2	Noise is not due to occlusion as assumed before. It is retained despite using SAM2.		Yes	No	Tested out various noise removal methods and found a fix for dense noise
12	R-5	ZED SDK and CUDA compatability issues	Shreya	Yes	No	Reinstalled firmware, purged CUDA and ZED SDK, and set up everything from scratch
13	R-2	Noisy Depth Map	Shreya	Yes	No	Switched to windows for data collection
14	R-5	All packages updated to Numpy 2.0 which broke the entire pipeline	Shreya	Yes	No	Tested out a few different versions and figured out the correct combination. Updated requirements.txt to ensure repeatability.
15	R-5	MuJoCo/Moveit integration had many compatibility issues, especially for grippers.	Megan, Emma, Kailash	Yes	No	Move to Gazebo
16	R-4	Bin deforming after testing too many times	Emma	Yes	No	Purchased a second bin just in case for the demos
17	R-1	Robot hitting un-indentified object during testing	Kartik	Yes	No	During testing and demo, always have someone at e-stop

9 Conclusions

9.a Lessons Learned

Throughout the development of MAGIC, we gained a deep appreciation for the complexity of building a fully integrated dual-arm robotic inspection system. While individual components such as perception, manipulation planning, calibration, and reconstruction could each be developed in isolation, making these components operate reliably as a unified system required significantly more time, iteration, and engineering discipline than originally anticipated. Our perception pipeline demonstrated this early on. We adopted a YOLO-based detector, but the model introduced strong practical constraints: the overhead camera had to maintain a strict bird's-eye viewpoint of the object, the handle had to remain visible at all times, and the handles themselves needed to be color-coded to ensure consistent detection. These limitations forced us to carefully structure our hardware configuration, data collection process, and object presentation strategy.

Manipulation planning presented similar challenges. We experimented with multiple planning pipelines and configurations, often encountering issues such as unreachable grasps, inconsistent arm behavior, and unstable trajectories. It was only through repeated iteration, advisor feedback, and discussions with Ph.D. students that we converged on a planner setup capable of stable execution. One of our most impactful decisions was to develop perception and manipulation in parallel using a turntable to simulate multi-view capture. This allowed us to test the complete pipeline early, long before dual-arm integration, and it helped us identify hidden assumptions and transform inconsistencies that would have otherwise appeared much later.

As we approached the final stages of the project, integration became increasingly efficient. Most

of the remaining issues required only small corrections, such as applying ICP refinement to compensate for residual camera or hardware offsets that software calibration alone could not resolve. At the same time, we learned important lessons about team coordination. Ensuring that at least one member from each subteam participated in cross-subteam work sessions became essential for preventing misalignment. A notable example occurred when the perception team, who are not usually the ones to launch the system, accidentally flipped the arms' IP configuration over a weekend while the manipulation team was unavailable and made lots of changes. The TF chain had to be completely rebuilt. This reinforced the importance of shared documentation, consistent communication, and routine joint validation.

Working with emerging frameworks such as ROS 2, MoveIt 2, and modern 6D pose estimation models also taught us the realities of engineering on rapidly evolving platforms. We frequently encountered incomplete documentation, evolving APIs, and subtle behavior differences, all of which pushed us to strengthen our debugging practices and create custom tools when standard interfaces proved insufficient. Hardware reliability posed additional challenges, as several apparent software issues were ultimately traced back to calibration drift, sensor misalignment, or physical connection errors. Collectively, these experiences underscored the importance of disciplined engineering practices, from thorough testing to consistent calibration.

9.b Future Work

If MAGIC were to be developed into a commercial product or the foundation of a startup, several key areas of future work would guide the next stage of development. First, the perception system would need to be generalized beyond its current constraints. This includes fine-tuning 6D pose estimation models on a broader range of industrial components, reducing reliance on color-coded features, incorporating robust multi-sensor fusion, and creating scalable tools for collecting and labeling customer-specific datasets. Improving generalization would be essential for supporting real-world variability in lighting, texture, and part geometry.

Second, the manipulation subsystem would require industrial-grade reliability and safety enhancements. Automated calibration routines, drift detection, simplified operator procedures, and safety-certified control frameworks would be necessary to support continuous factory deployment. The lessons learned from issues such as transform inconsistencies and arm mounting mistakes highlight the importance of automating as much of the calibration and verification process as possible.

Third, inspection throughput and reconstruction accuracy would need to be improved. Integrating higher-precision sensors such as structured-light or laser scanners, accelerating reconstruction algorithms, and coupling deviation analysis directly with downstream CNC toolpath generation would enable MAGIC to function as part of a closed-loop manufacturing workflow. This capability would deliver meaningful time savings and reduce manual labor in hybrid additive/subtractive manufacturing environments.

Finally, validating market fit through pilot deployments would be a critical next step. Early adopters could include aerospace manufacturers, WAAM fabrication shops, and heavy-industry producers who face pressure to reduce inspection time while improving consistency. Through paid pilot programs, we would refine system performance, quantify economic value, evaluate return on investment, and determine a sustainable hardware and service model. These steps would ultimately position MAGIC as a viable product offering for automated quality inspection in advanced manufacturing.

In summary, MAGIC evolved from a collection of independent prototypes into a fully integrated

robotic inspection platform capable of addressing real industrial challenges. The lessons we learned such as technical, organizational, and market-focused have prepared us to envision how the system could mature into a deployable commercial solution. With further development in perception robustness, manipulation reliability, reconstruction accuracy, and customer validation, MAGIC has the potential to transition from a research prototype into a commercially impactful inspection system.

References

[1] Admin. Wire arc additive manufacturing (waam) technology: An overview, September 2024. Accessed: 2025-04-30.

[2] Carnegie Mellon University College of Engineering. Advanced inspection techniques for wire arc additive manufacturing, n.d. Accessed: 2025-04-30.

[3] RAMLAB. Waam explained - wire arc additive manufacturing. `https://www.ramlab.com/resources/waam-101/`, 2023. Accessed: 2025-04-30.

[4] H. J. Son, B. W. Seo, C. J. Kim, et al. Coordinate system setting for post-machining of impeller shape by wire arc ded and evaluation of processing efficiency. *Scientific Reports*, 14:18262, 2024.

A Appendix

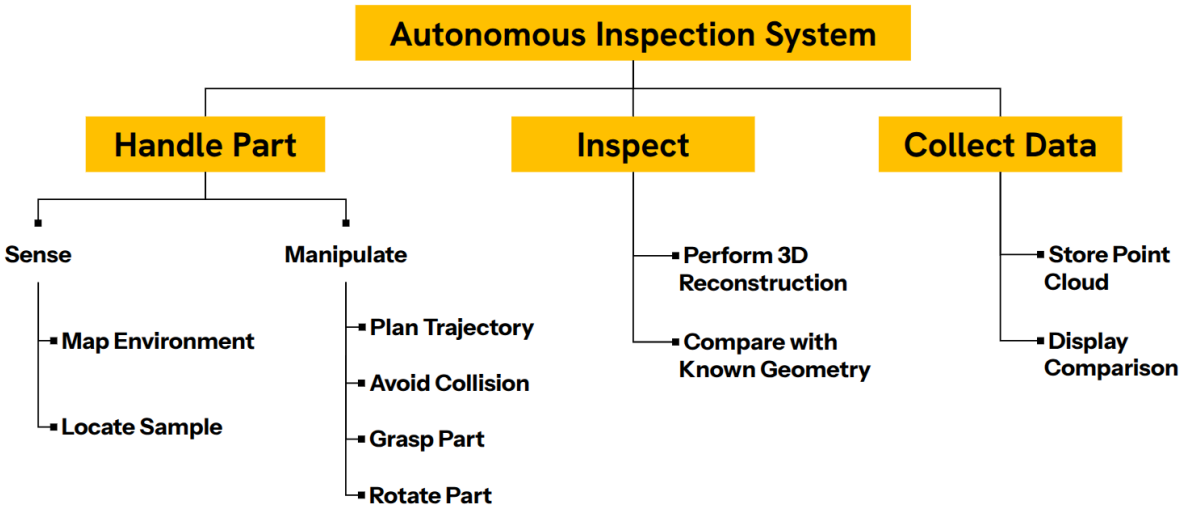


Figure 14: Objectives tree

Table 13: Robot Arms Collide

Risk Title	R1 - Robot Arms Collide	**Date Submitted**	11/30/2024
Risk Owner	Megan	**Date Updated**	-
Description		**Risk type**	
Multiple robot arms may collide due to inadequate synchronization or inaccurate spatial awareness.		Technical, Cost	
Consequence		**Metrics**	
Damage to the robot arms, tooling, workspace, and inspected sample, potentially causing system downtime.		**Likelihood**	**Consequence**
		3	5
Mitigation Plan			
Action		**Outcome**	
Implement robust collision avoidance algorithms and conduct extensive simulation testing. During testing, someone is always on the manual e-stop.		Reduced likelihood of collisions, leading to improved operational safety and system reliability.	

Table 14: Falling Behind Schedule Because of Learning Curve

Risk Title	R2 - Falling behind schedule because of learning curve	**Date Submitted**	11/30/2024
Risk Owner	Kartik	**Date Updated**	-
Description		**Risk type**	
The team requires additional time to understand and adapt to new robotic system components or software.		Schedule	
Consequence		**Metrics**	
Project delays and increased development costs.		**Likelihood**	**Consequence**
		4	3
Mitigation Plan			
Action		**Outcome**	
Provide comprehensive training sessions and allocate time for initial experimentation; Seek mentorship early.		Accelerated team proficiency, enabling the project to return to its intended timeline.	

Table 15: Resource Availability

Risk Title	R3 - Resource availability	**Date Submitted**	11/30/2024
Risk Owner	Emma	**Date Updated**	-
Description		**Risk type**	
The team is sharing the robot arms with 2 PhD students and have limited access, In addition, we only have one lab pc and have to share with 5 people.		Schedule	
Consequence		**Metrics**	
Delayed testing and integration phases, slowing overall system development.		**Likelihood**	**Consequence**
		3	4
Mitigation Plan			
Action		**Outcome**	
Schedule lab time efficiently Set up SSH and Teamviewer		Improved utilization of resources and steady project progress.	

Table 16: End Effector or Test Object Breaks

Risk Title	R4 - End effector or test object breaks	**Date Submitted**	11/30/2024
Risk Owner	Kailash	**Date Updated**	-
Description		**Risk type**	
The end effector may become damaged due to excessive force, wear, or improper use.		Technical	
Consequence		**Metrics**	
Can't do demo.		**Likelihood**	**Consequence**
		3	5
Mitigation Plan			
Action		**Outcome**	
Keep backup end effectors and test object readily available and ensure quick-change mechanisms for rapid replacement.		Minimal downtime during failures, maintaining operational continuity and project timelines.	

Table 17: Integration Issues Between Subsystems

Risk Title	R5 - Integration issues between subsystem	**Date Submitted**	11/30/2024
Risk Owner	Shreya	**Date Updated**	-
Description		**Risk type**	
Subsystems fail to communicate or function cohesively due to compatibility or design mismatches.		Schedule	
Consequence		**Metrics**	
Delays in testing and operation, with potential rework required for subsystems.		**Likelihood**	**Consequence**
		3	5
Mitigation Plan			
Action		**Outcome**	
Perform unit test and subsystem validation experiments continuously. Integrate one subsystem at a time.		Improved interoperability, reduced integration delays, and smoother system functionality.	

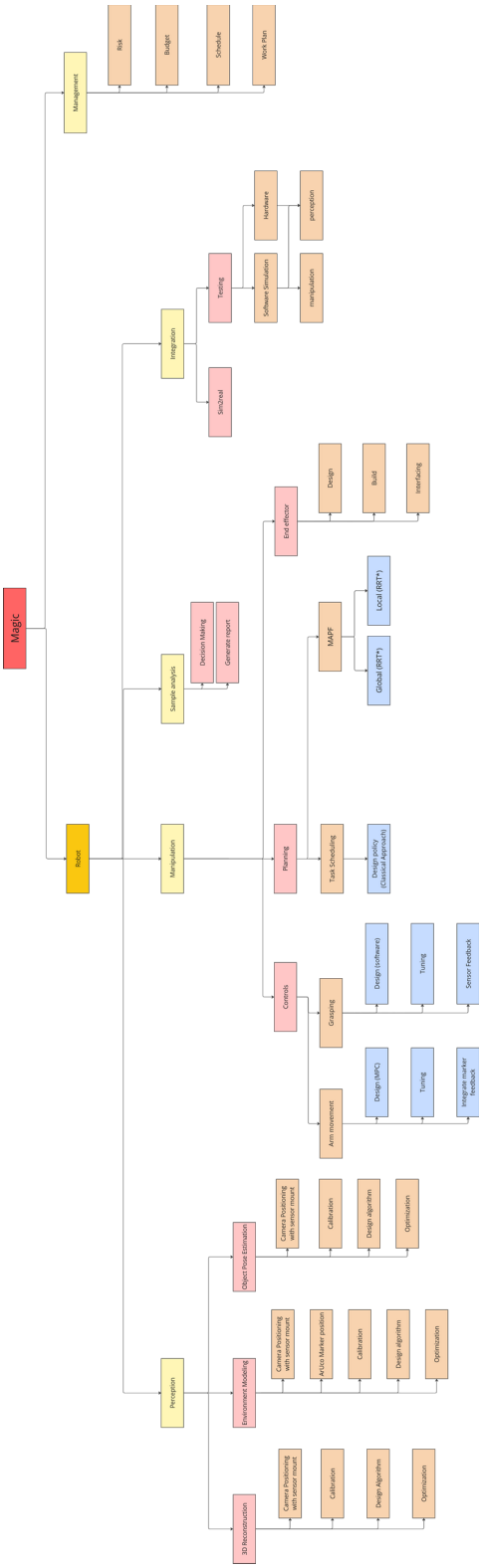


Figure 15: Work Breakdown Structure