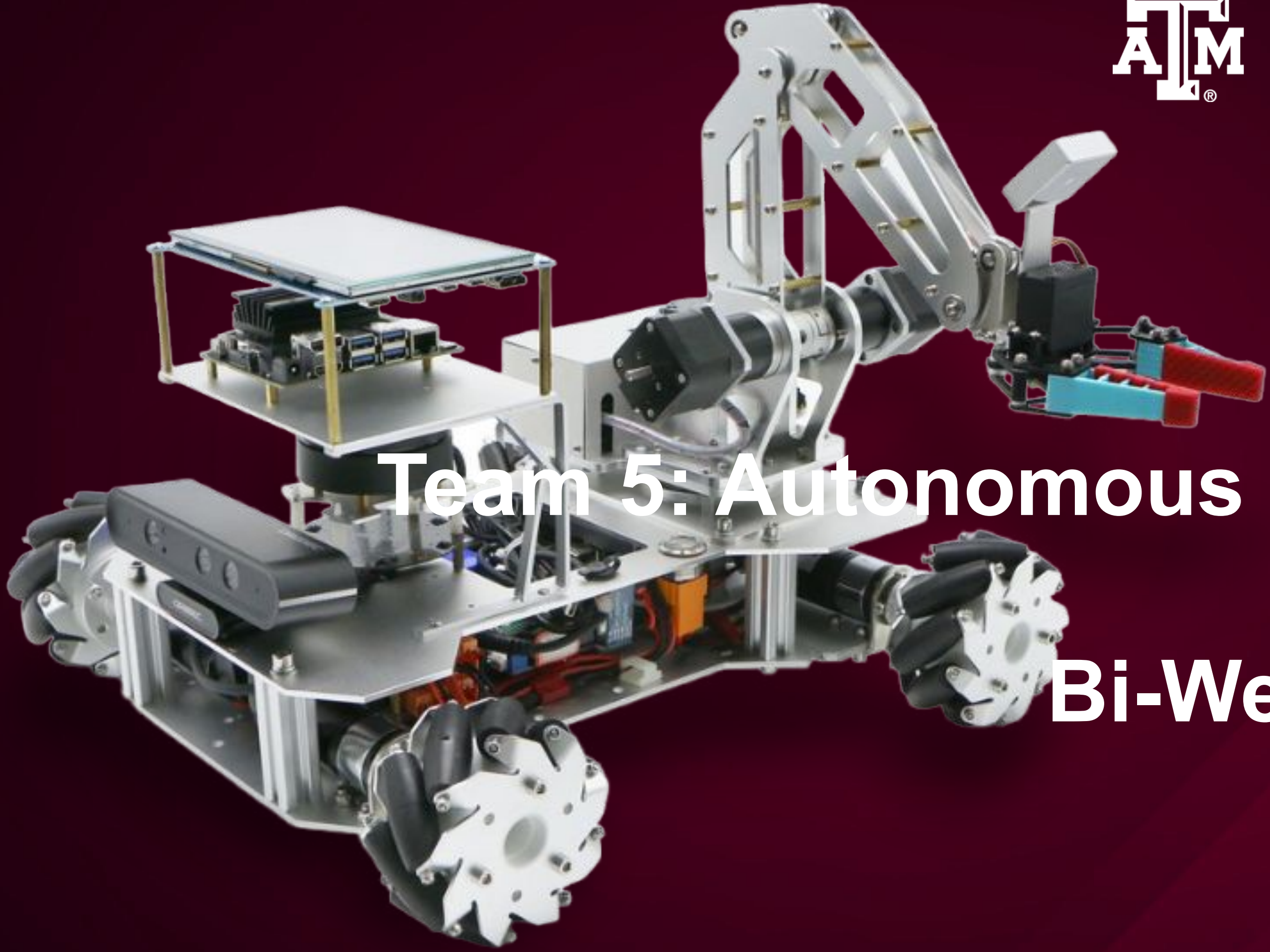
Dwight Look College of
ENGINEERING
TEXAS A&M UNIVERSITY



Team 5: Autonomous Object Picking Robot

Bi-Weekly Update 2

Christoffer Cox
James Dickson
Kathy Vo
Sponsor: Swarnabha Roy
TA:Niloofar Borzooei

Executive Summary

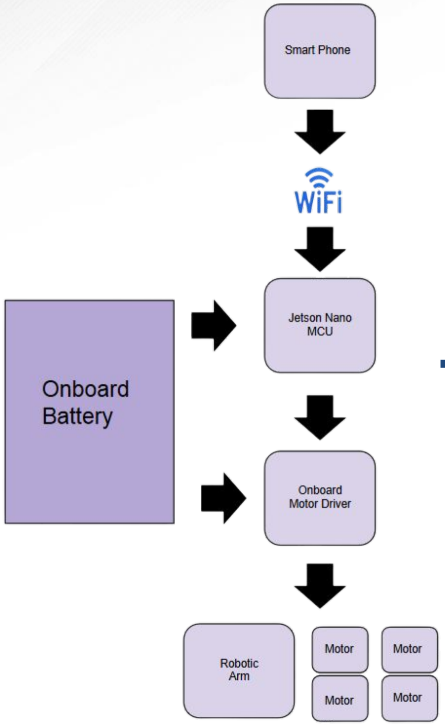
Problem statement:

- With millions of packages processed daily through warehouses, the need for safe and efficient sorting methods is of the utmost importance.
- Workplace injury rates are increasing as the demand grows.

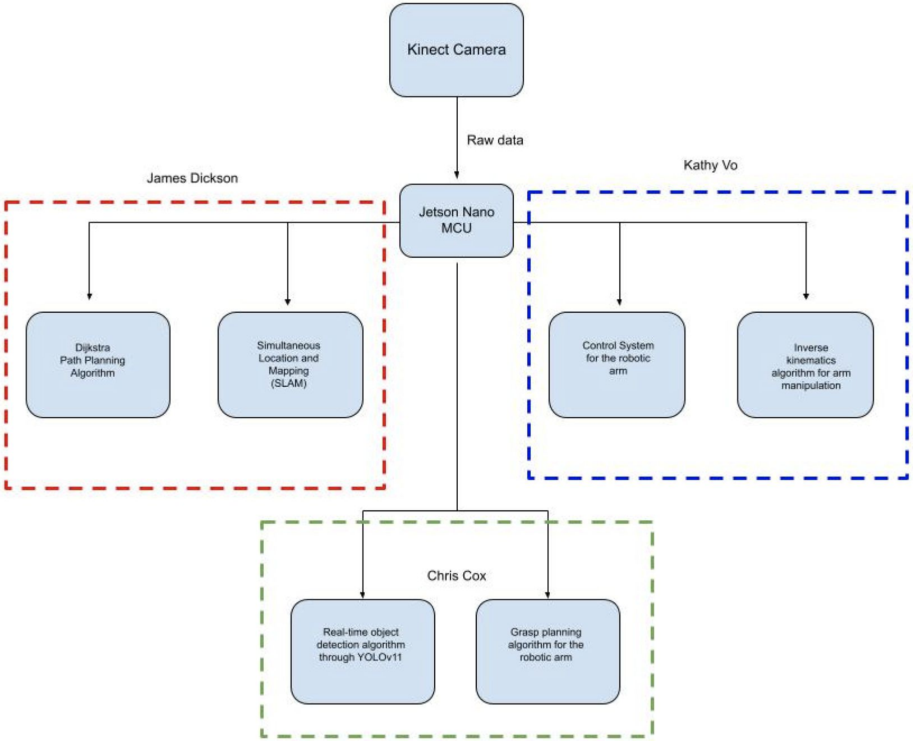
The autonomous object picking robot offers:

- The ability to navigate through indoor environments with no human control
- The ability to differentiate between objects depending on the color
- The ability to grasp and transport designated objects to a given location

Project/Subsystem Overview



Subsystems


Kinect Camera
Raw data
Jetson Nano MCU
James Dickson
Dijkstra Path Planning Algorithm
Simultaneous Location and Mapping (SLAM)
Kathy Vo
Control System for the robotic arm
Inverse kinematics algorithm for arm manipulation
Chris Cox
Real-time object detection algorithm through YOLOv11
Grasp planning algorithm for the robotic arm

Project Timeline

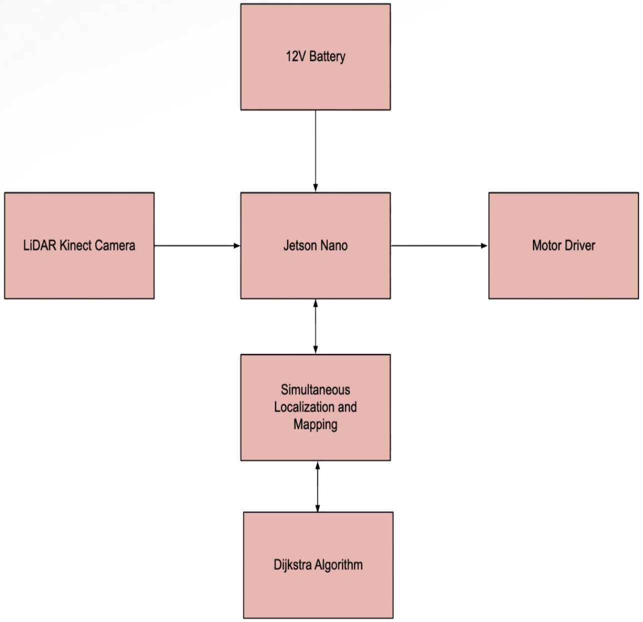
Finalize initial subsystem codes to prepare for integration (completed 1/20)	Integrate code for the Kinect camera with the Jetson Nano and motor controller (completed 2/3)	Autonomously move robot and arm using real world detection (to complete by 2/24)	Fully make the robot independent and combine all parts together (to complete by 3/17)	Test system and refine any bugs or issues (to complete by 4/14)	Final Demo and report (to complete by 4/26)

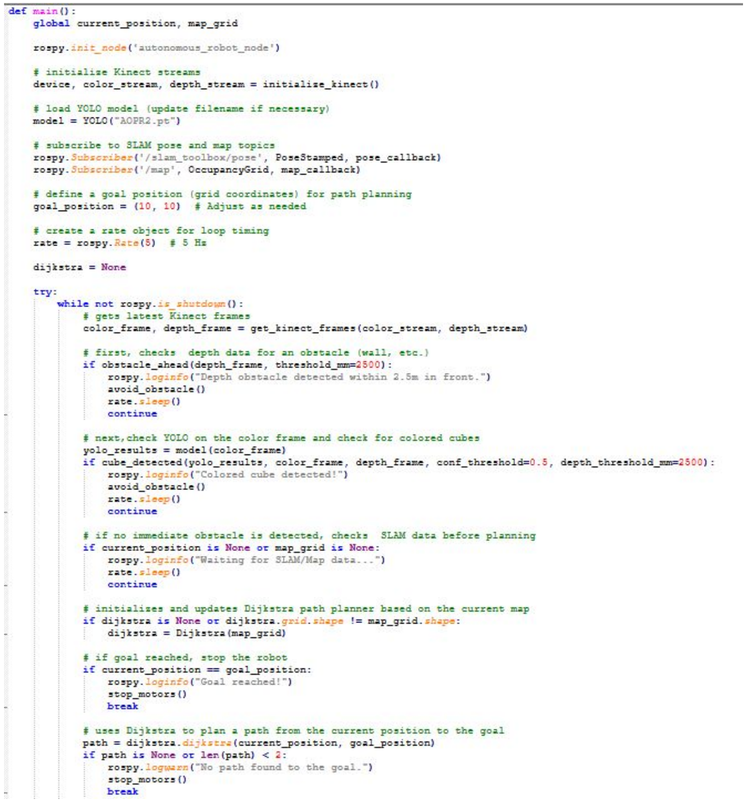
Object Navigation and Object Avoidance

Accomplishments since last update 20 hrs of effort	Ongoing progress/problems and plans until the next presentation
- Completed code integration of obstacle detection and navigation systems. - Installed new batteries and began testing limits of movement (max speed, turn speed, etc).	- Ongoing integration with obstacle detection system. - By next review, obstacle detection and navigation system are expected to be integrated.

Object Navigation and Avoidance

James Dickson





```
def main():
    global current_position, map_grid

    rospy.init_node('autonomous_robot_node')

    # initialize Kinect streams
    device, color_stream, depth_stream = initialize_kinect()

    # load YOLO model (update filename if necessary)
    model = YOLO("AOPR2.pt")

    # subscribe to SLAM pose and map topics
    rospy.Subscriber('/slam_toolbox/pose', PoseStamped, pose_callback)
    rospy.Subscriber('/map', OccupancyGrid, map_callback)

    # define a goal position (grid coordinates) for path planning
    goal_position = (10, 10)  # Adjust as needed

    # create a rate object for loop timing
    rate = rospy.Rate(5)  # 5 Hz

    dijkstra = None

    try:
        while not rospy.is_shutdown():
            # gets latest Kinect frames
            color_frame, depth_frame = get_kinect_frames(color_stream, depth_stream)

            # first, checks  depth data for an obstacle (wall, etc.)
            if obstacle_ahead(depth_frame, threshold_mm=2500):
                rospy.loginfo("Depth obstacle detected within 2.5m in front.")
                avoid_obstacle()
                rate.sleep()
                continue

            # next,check YOLO on the color frame and check for colored cubes
            yolo_results = model(color_frame)
            if cube_detected(yolo_results, color_frame, depth_frame, conf_threshold=0.5, depth_threshold_mm=2500):
                rospy.loginfo("Colored cube detected!")
                avoid_obstacle()
                rate.sleep()
                continue

            # if no immediate obstacle is detected, checks  SLAM data before planning
            if current_position is None or map_grid is None:
                rospy.loginfo("Waiting for SLAM/Map data...")
                rate.sleep()
                continue

            # initializes and updates Dijkstra path planner based on the current map
            if dijkstra is None or dijkstra.grid.shape != map_grid.shape:
                dijkstra = Dijkstra(map_grid)

            # if goal reached, stop the robot
            if current_position == goal_position:
                rospy.loginfo("Goal reached!")
                stop_motors()
                break

            # uses Dijkstra to plan a path from the current position to the goal
            path = dijkstra.dijkstra(current_position, goal_position)
            if path is None or len(path) < 2:
                rospy.logwarn("No path found to the goal.")
                stop_motors()
                break
```

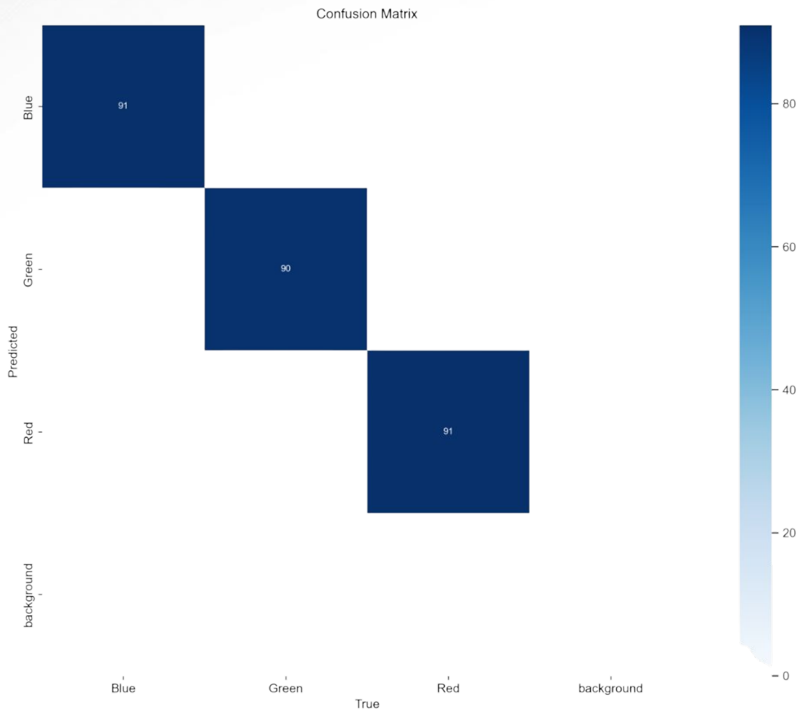
Object detection and Grasping

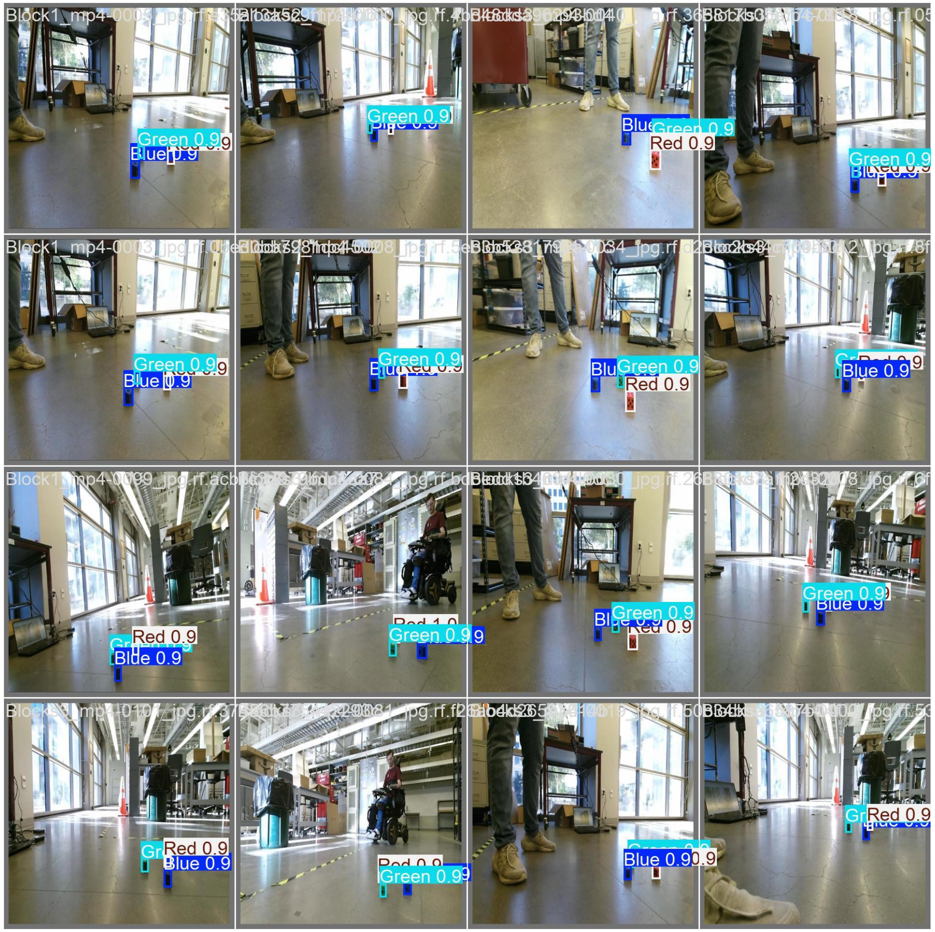
Chris Cox

Accomplishments since last update 20 hrs of effort	Ongoing progress/problems and plans until the next presentation
- Interfaced object detection code with the microcontroller - Trained 3 additional YOLOv11 models with different datasets and weights - Began working on extracting the 3D data from the Kinect for the navigation and 3D object recognition	- Finalize code extracting 3D data to be implemented onto Gazebo - Work on programming object recognition combined with the robotic arm

Object detection and Grasping

Chris Cox



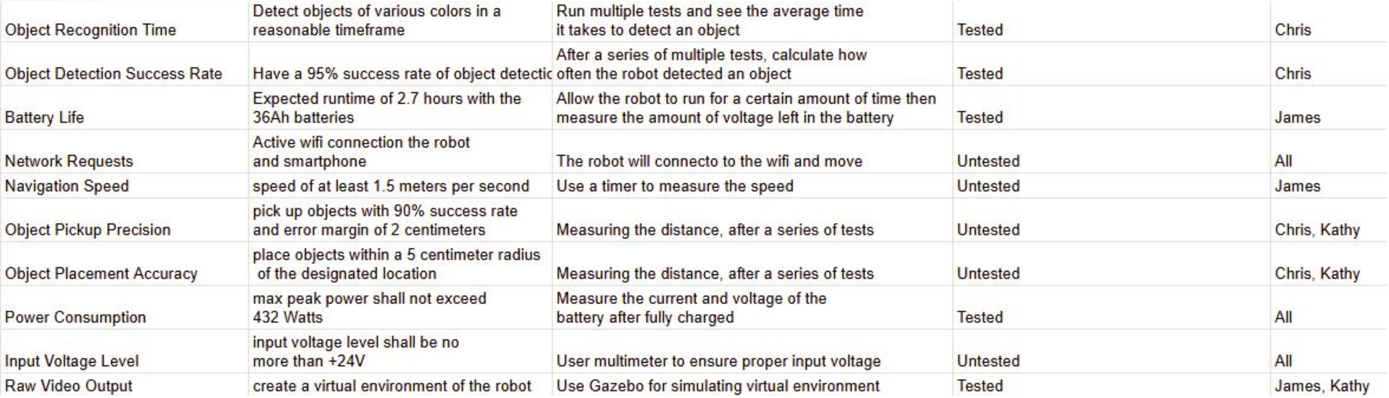


Arm Manipulation

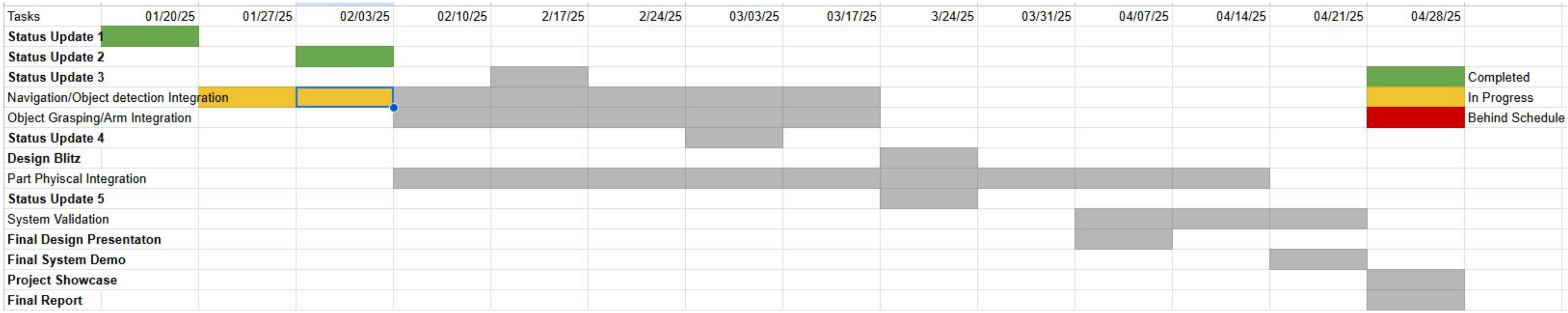
Kathy Vo

Accomplishments since last update 10 hrs of effort	Ongoing progress/problems and plans until the next presentation
- Approval from Sponsor, bought a new 6 servo robot arm	- Updating python/downloading more packages to help run the object detection code on the Jetson - Send data from the Kinect camera to the Esp32 microcontroller

Execution & Plan



Object Recognition Time	Detect objects of various colors in a reasonable timeframe	Run multiple tests and see the average time it takes to detect an object	Tested	Chris
Object Detection Success Rate	Have a 95% success rate of object detectic	After a series of multiple tests, calculate how often the robot detected an object	Tested	Chris
Battery Life	Expected runtime of 2.7 hours with the 36Ah batteries	Allow the robot to run for a certain amount of time then measure the amount of voltage left in the battery	Tested	James
Network Requests	Active wifi connection the robot and smartphone	The robot will connecto to the wifi and move	Untested	All
Navigation Speed	speed of at least 1.5 meters per second	Use a timer to measure the speed	Untested	James
Object Pickup Precision	pick up objects with 90% success rate and error margin of 2 centimeters	Measuring the distance, after a series of tests	Untested	Chris, Kathy
Object Placement Accuracy	place objects within a 5 centimeter radius of the designated location	Measuring the distance, after a series of tests	Untested	Chris, Kathy
Power Consumption	max peak power shall not exceed 432 Watts	Measure the current and voltage of the battery after fully charged	Tested	All
Input Voltage Level	input voltage level shall be no more than +24V	User multimeter to ensure proper input voltage	Untested	All
Raw Video Output	create a virtual environment of the robot	Use Gazebo for simulating virtual environment	Tested	James, Kathy



Tasks	01/20/25	01/27/25	02/03/25	02/10/25	2/17/25	2/24/25	03/03/25	03/17/25	3/24/25	03/31/25	04/07/25	04/14/25	04/21/25	04/28/25	
Status Update 1	🟩														
Status Update 2			🟩												
Status Update 3					▨									🟩	Completed
Navigation/Object detection Integration		🟨	🟨	▨	▨	▨	▨	▨						🟨	In Progress
Object Grasping/Arm Integration				▨	▨	▨	▨	▨						🟥	Behind Schedule
Status Update 4							▨								
Design Blitz									▨						
Part Phyiscal Integration				▨	▨	▨	▨	▨	▨	▨	▨	▨			
Status Update 5									▨						
System Validation											▨	▨	▨		
Final Design Presentaton											▨				
Final System Demo													▨		
Project Showcase														▨	
Final Report														▨	