SDP25

Team #15
Workspace Wizard

Meet the Team

Aryaman Ghura
CompE & CS

Software

Ishaan Salian
CompE

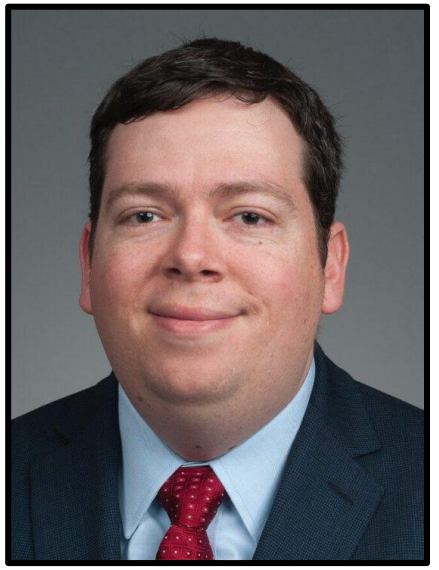
Motors & PCB
Logistics

Mary Esenther
CompE

Power Delivery

Kavya Manchanda
CompE

Processor & Camera
Budget



Professor Marco Duarte

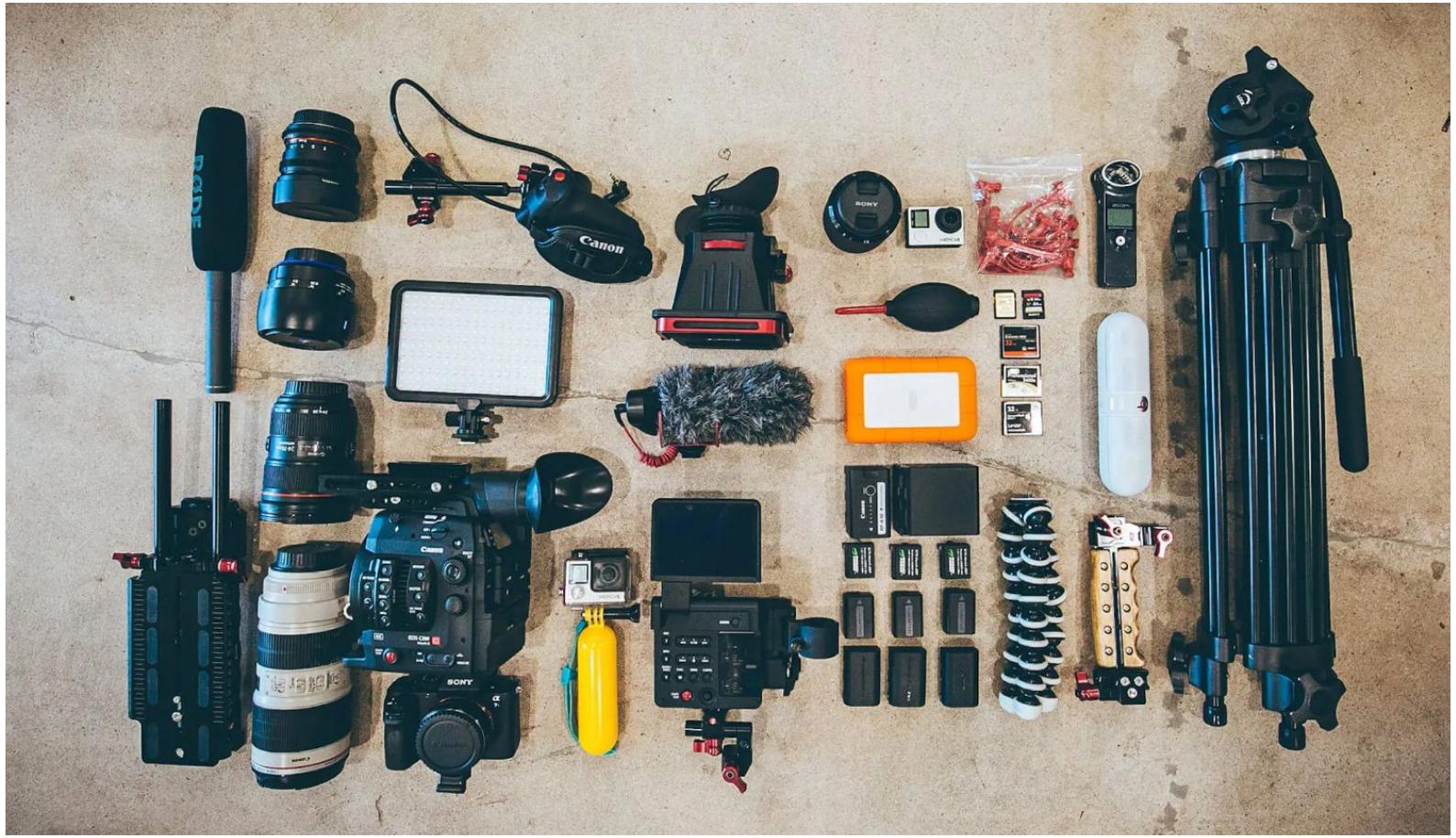
Advisor

Problem Statement

- A cluttered desk can negatively affect productivity and mental health.
- Studies show that messy workspaces increase stress. [1]
- 57% of Americans associate messy desks with laziness. [2]

[1] "The Psychological Consequences of Clutter," Psychology Today, 2021.
[2] L. Alton, "The Negative Relationship Between a Messy Desk and Productivity," Inc.com, Feb. 16, 2017.

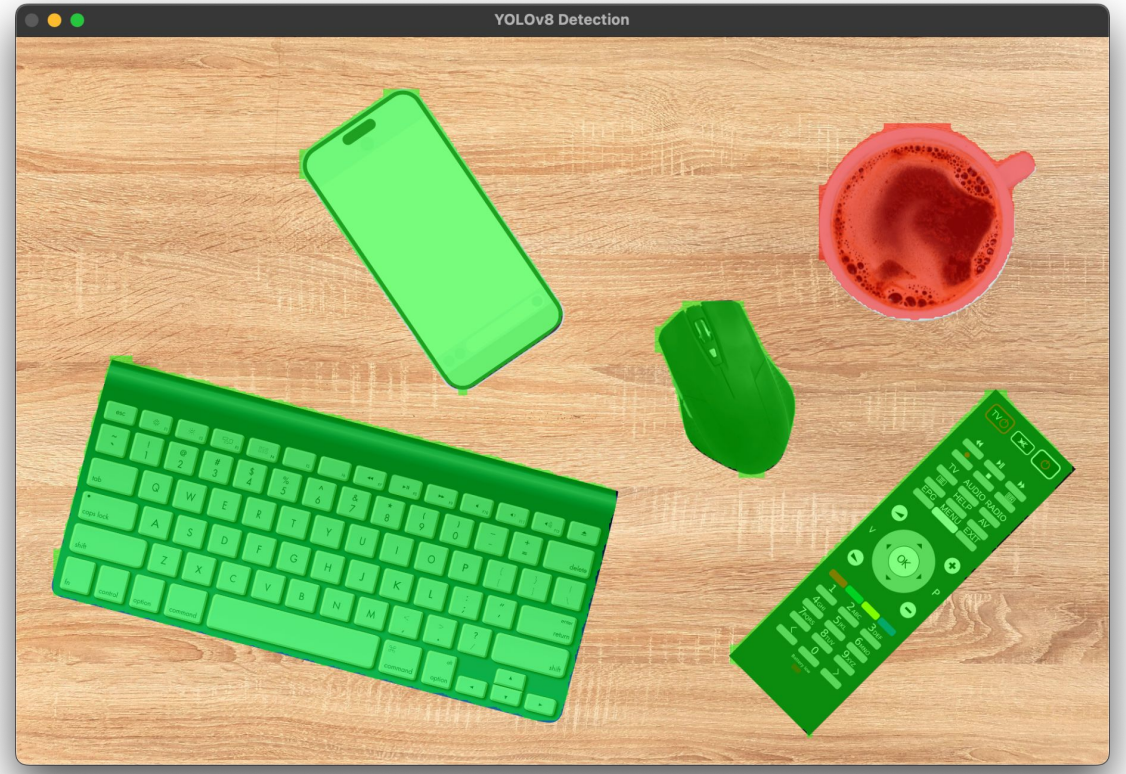
Background: Knolling

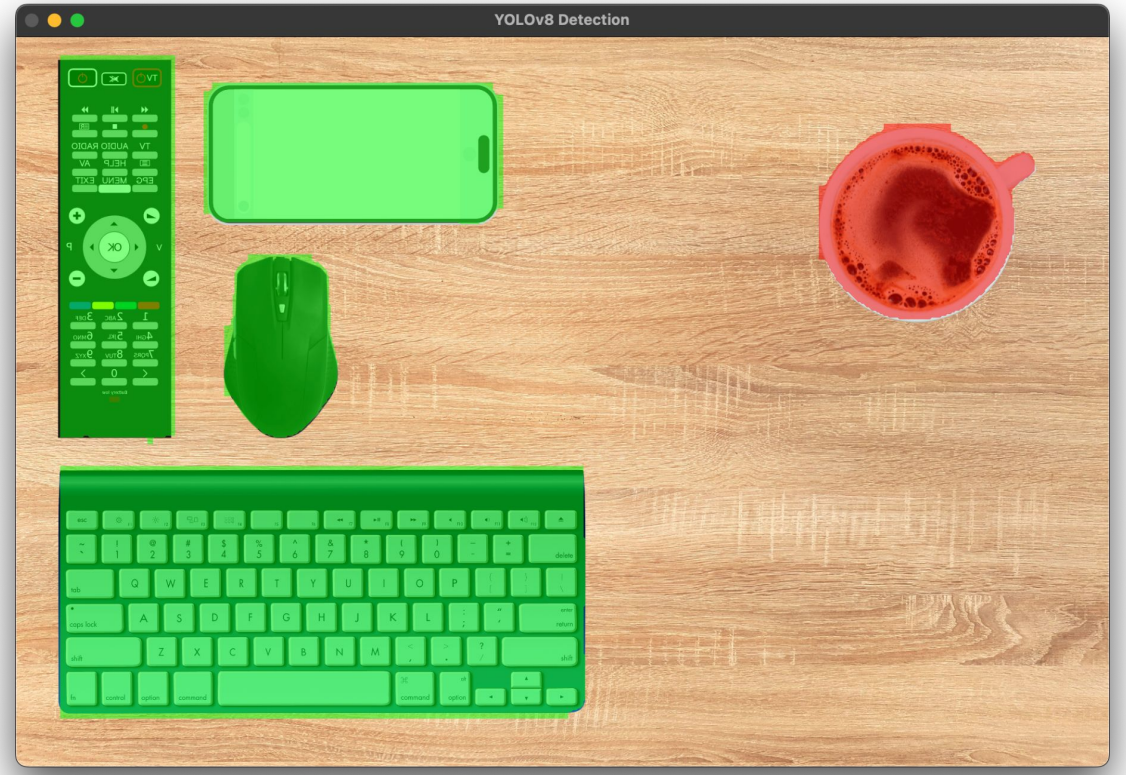


Project Goal

To create a distributed system that can organize your workspace autonomously by employing object detection, path finding and self charging.

Classification of Objects





*Object detection model mentioned later

Preliminary Design Specifications

Overhead Computer

- Detect human absence with 98% accuracy to start knolling.
- Map and find viable paths for object placement.
- Align objects at ~90° angles with a variance of ±5°.
- Identify objects to avoid, like liquids or monitors, with 98% accuracy.
- Recognize table boundaries to prevent overshooting.

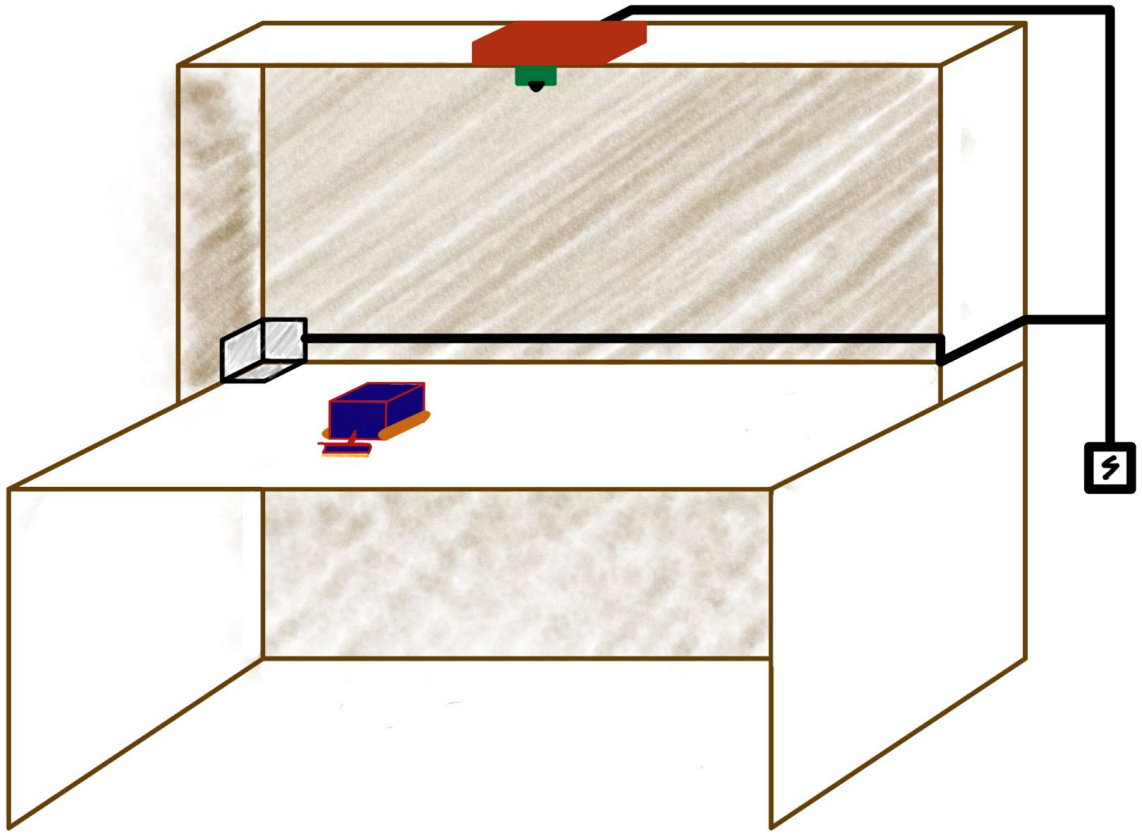
Robot

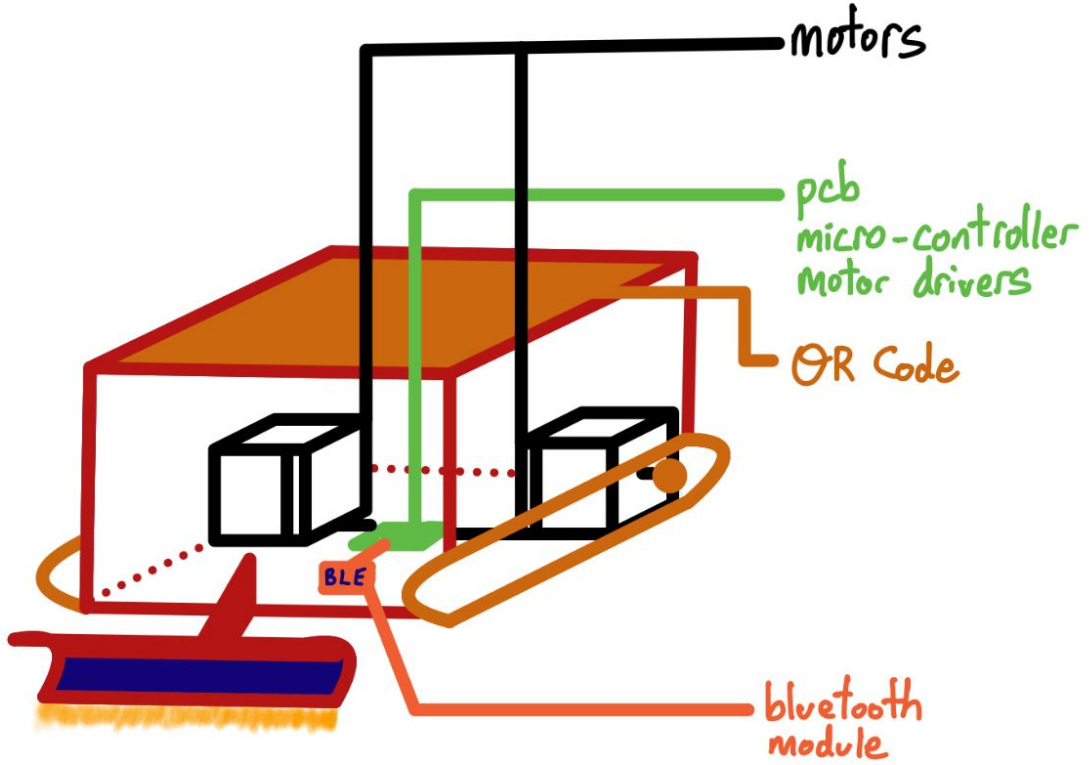
- Run for 30±5 minutes on one full charge.
- Knoll objects up to 4.5lbs, like a MacBook.
- Knoll up to 10 objects.

Testing Plan based on Preliminary Design Specifications

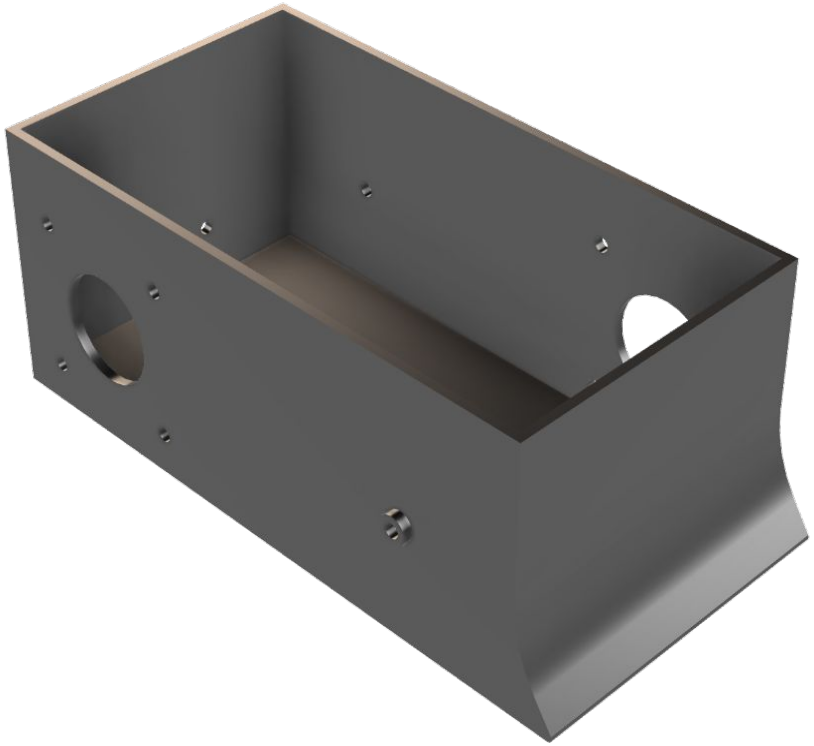
Design Specification	Testing Plans
Detect human absence	Robot movement starts only after 2 minutes of human absence in at least 98 out of 100 trials (reduced from 20 minutes for demo purposes)
Map viable paths for object placement	The robot should successfully navigate around obstacles to find viable paths without getting stuck, if there exists one.
Align objects at ~90±5° angles	Measure the angle of the knolled objects using edge of the table as a guide
Identify objects to avoid	System should identify various forms objects (open cup, bottle, user-defined objects) with correct classifications in at least 98 out of 100 trials
Recognize table boundaries	No objects, nor the robot, should exceed the table boundaries
Run for 30±5 minutes on one full charge	Prevent robot from reaching charging station until 25 minutes. Robot must still have enough charge to dock itself onto the charging dock.
Move objects up to 4.5lbs, like a MacBook	The robot should successfully push objects weighing up to 4.5lbs
Knoll up to 10 (varying) objects	The robot should knoll all 10 objects accurately in at least 95% of trials, ensuring no objects are missed or improperly aligned

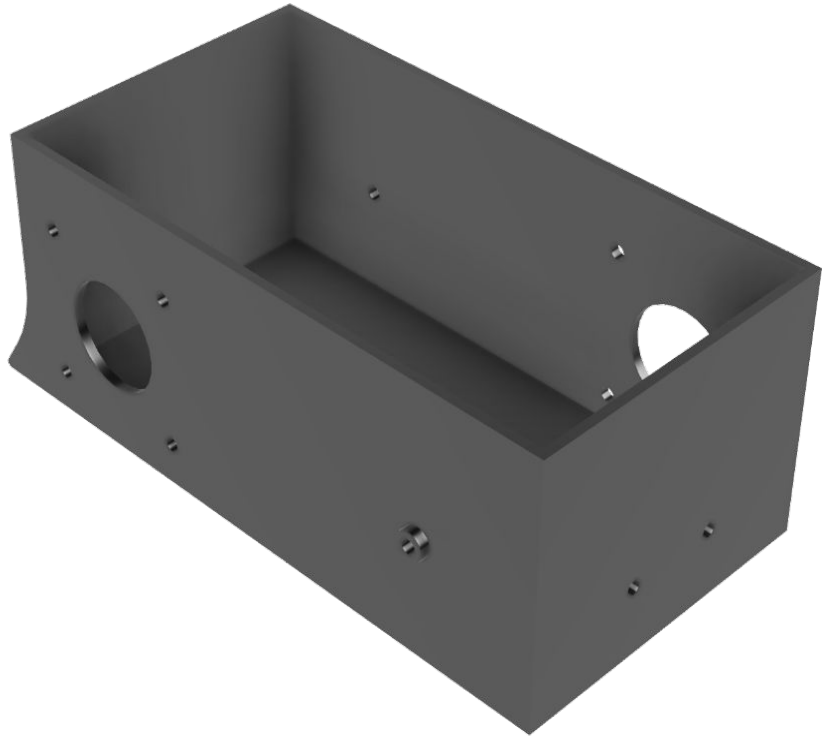
Mechanical Diagram (no change)



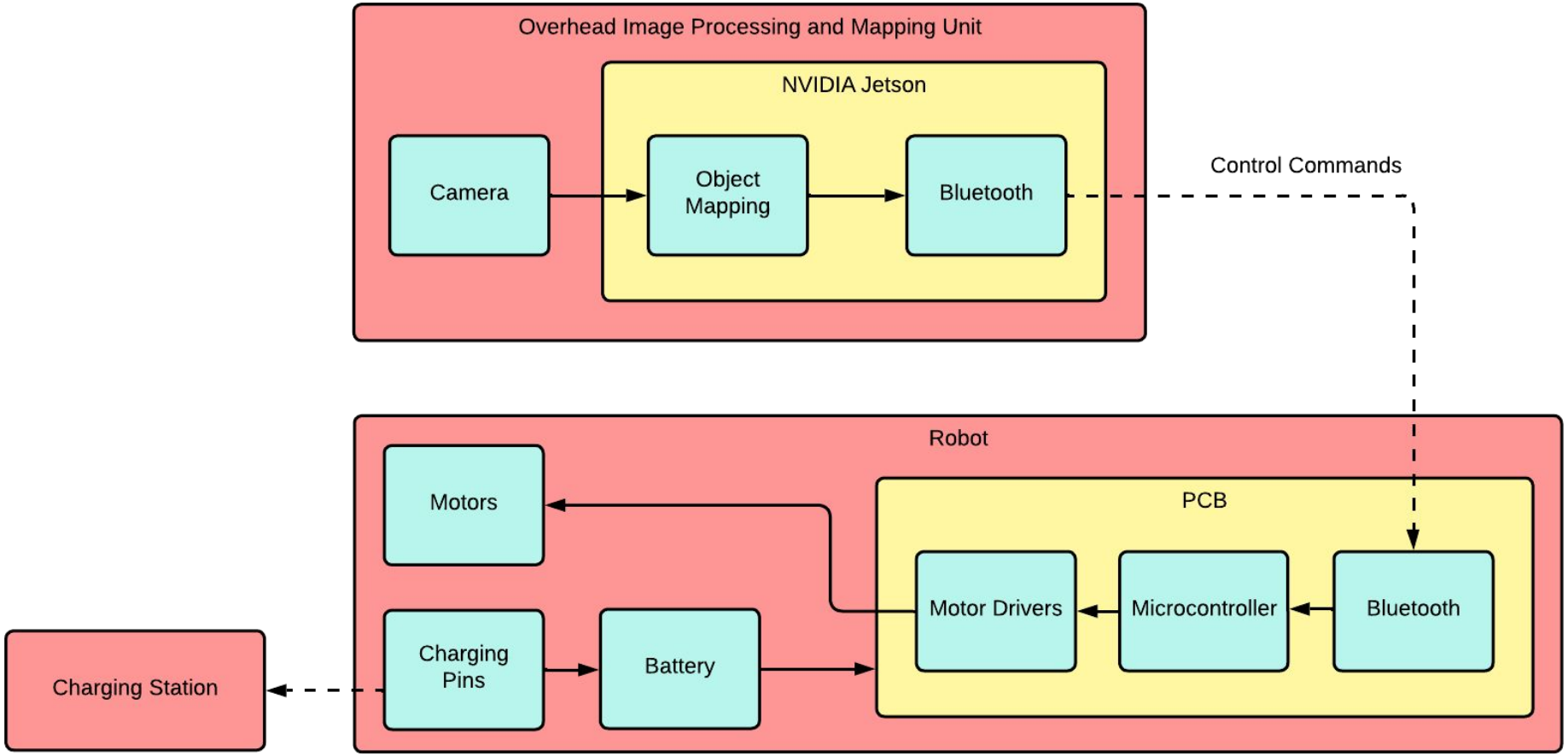


Current Design





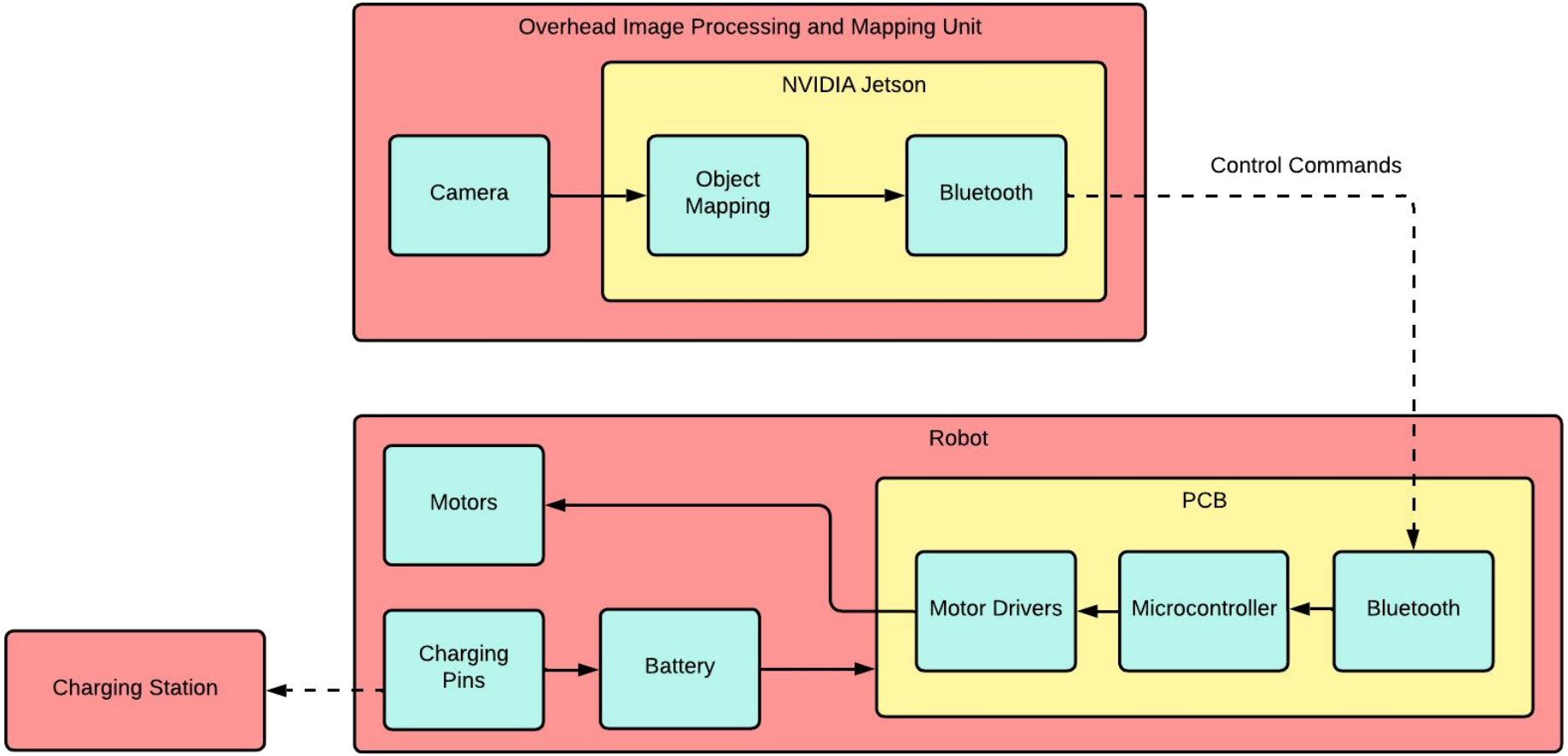
General Block Diagram



Overhead Computer:
NVIDIA Jetson Nano (Computer)
USB Webcam
Edimax N150 (BLE)

Robot:
ATmega328/P (MCU)
HC-05 (BLE)
TMC2209 (Drivers)
Stepper Motors
22V Li-Po Batteries

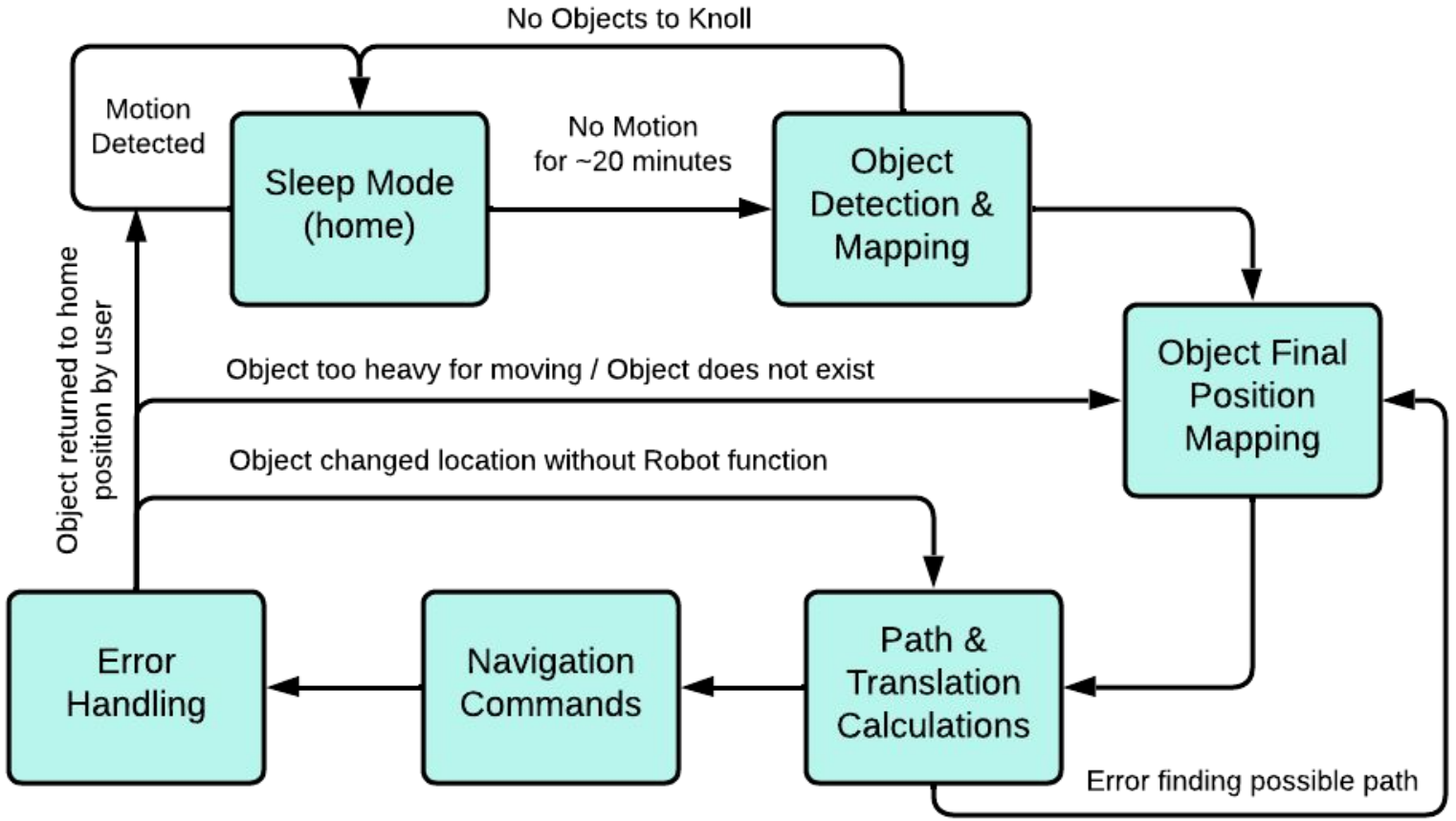
Updated General Block Diagram



Overhead Computer:
NVIDIA Jetson Nano (Computer)
JETSON CAMERA IMX219 (Webcam)
Edimax N150 (BLE)

Robot:
ESP32 (MCU+BLE)
TMC2209 (Drivers)
Stepper Motors,
22V Li-Po Batteries

Software Diagram



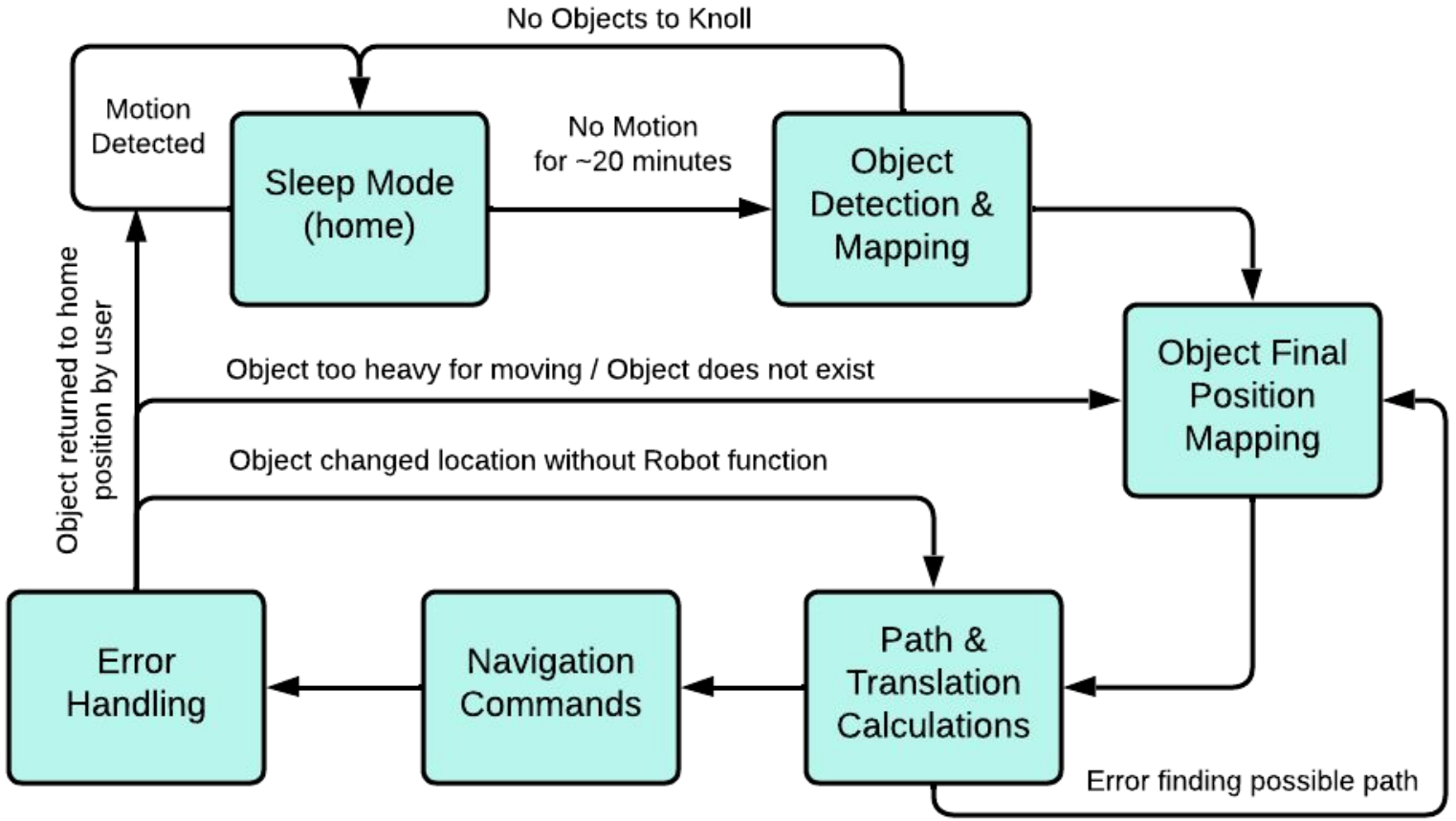
OpenCV:
Image processing and object detection.

PCL (Point Cloud Library):
2D mapping and object pose estimation.

ROS (Robot OS) (Hold):
Tools for robot control, motion planning, and sensor integration.

PyTorch/TensorFlow (Hold):
For building and training custom object detection models.

Updated Software Diagram

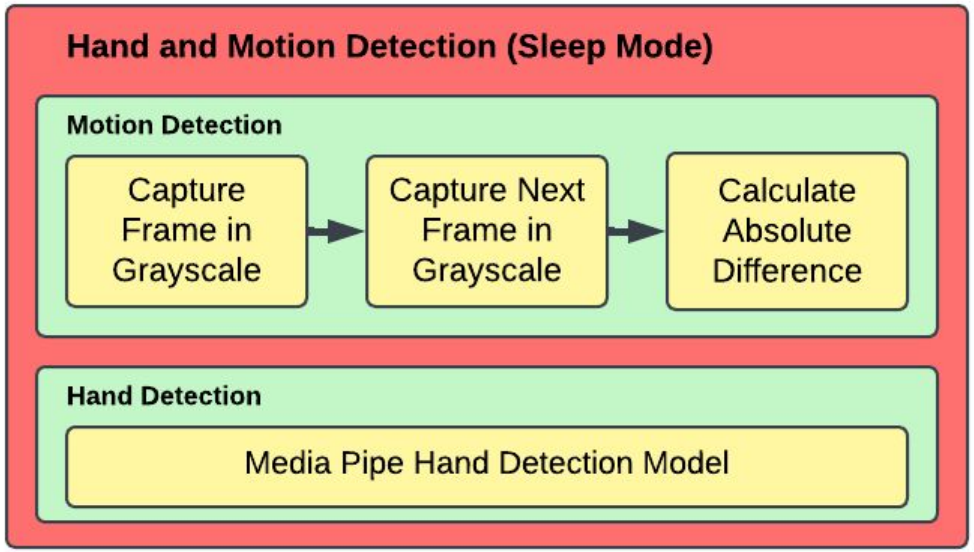


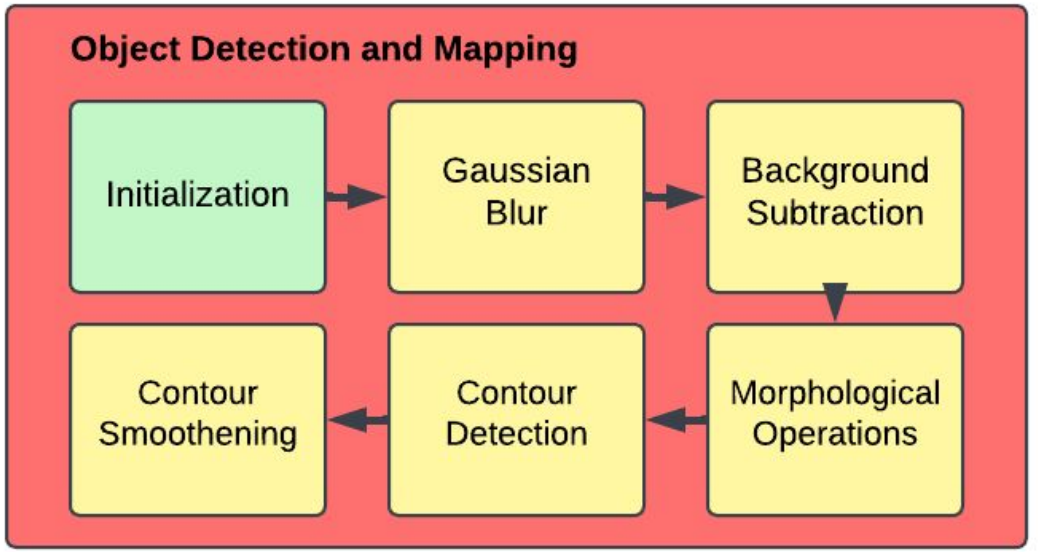
OpenCV, Media Pipe, DarkNet:
Image processing and object detection.

Pybullet:
Visualization and Testing

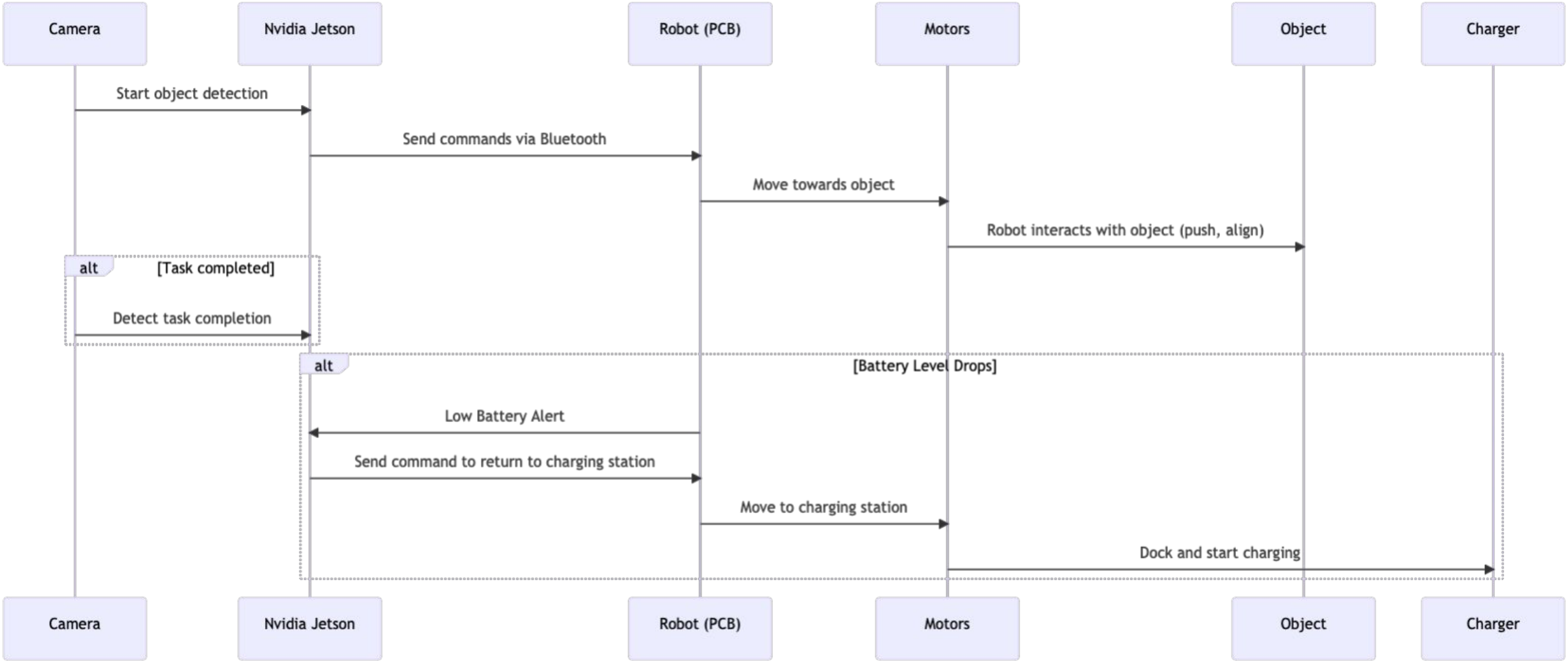
PCL (Point Cloud Library) (Hold):
2D mapping and object pose estimation.

Detailed Software Diagram





Process Diagram (no change)



MDR Deliverables

Overhead Computer + Software

- ✔ Camera and Jetson successfully working with software executing on inputs from camera.
- ✔ Ability to detect human presence and if current environment is knolled according to specifications.
- ✔ Ability to detect a randomly placed computer monitor, coffee mug, and phone on the desk and differentiate between the objects requiring knolling and those to be avoided.
- ✔ Ability to map each “knoll-able” object to a final “knolled” location virtually.
- X Ability to find viable path for robot to propel object from initial position to its final “knolled” location using the A* algorithm (desk will be divided into a grid for robot to find path).

Robot

- ✔ Robot is capable of traversing the table using Serial commands from computer.
- ✔ First iteration of the robot body is ready.

Power Delivery

- X Finalize charging mechanism. -> (MECHANISM CHANGED)
- X Successfully charge a 5V LiPo using the finalized charging mechanism.

USB Camera v/s CSI Camera

CSI camera has:

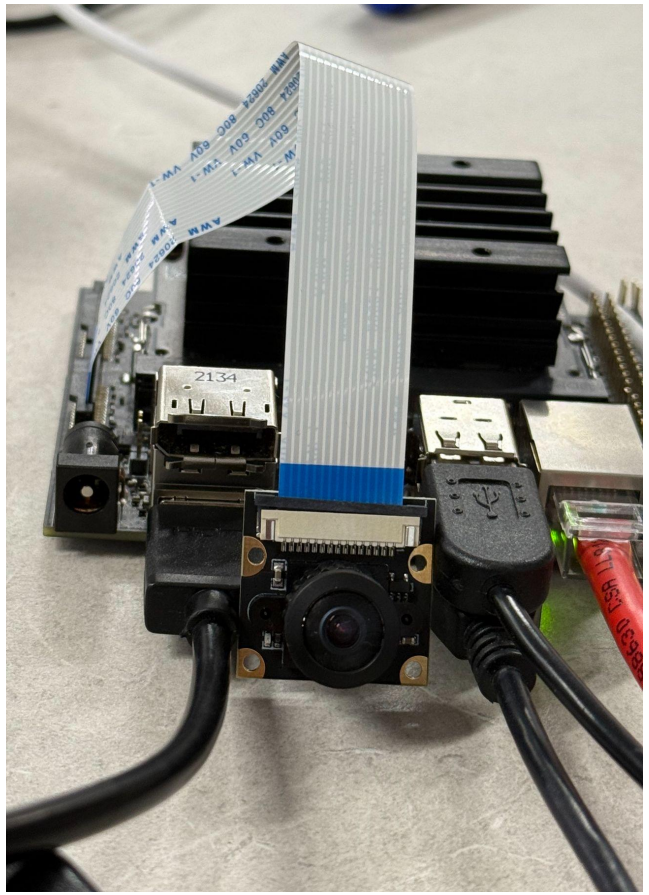
- Direct access to the GPU and 2 reserved slots for data transfer unlike multiple USB ports.

- Lower overhead as USB camera relies on CPU for image data transfer

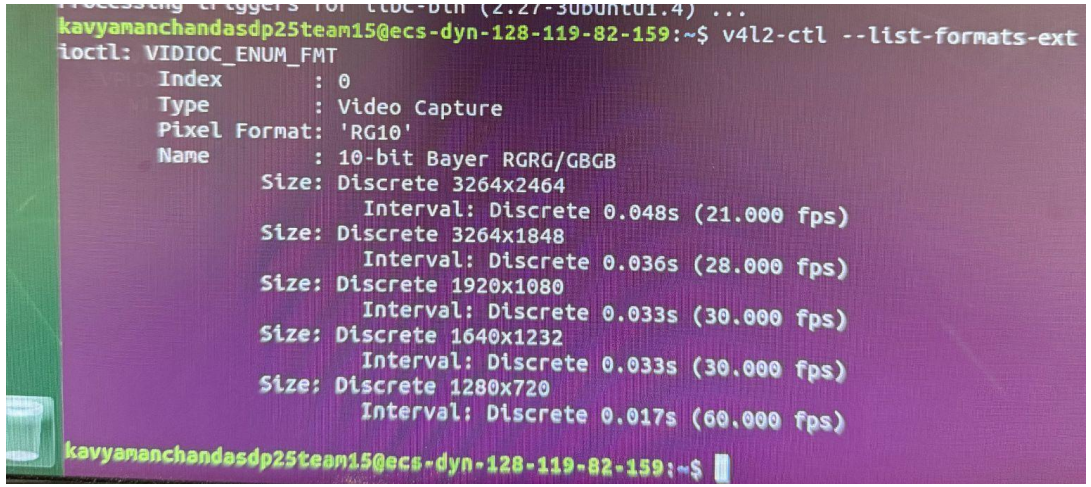
- Shorter setup time and lesser lines of code needed

- Support for NVIDIA Argus framework for image capture and processing. Argus supports real-time processing, and features like auto-exposure, auto-white-balance and de-noising.

- Compact size



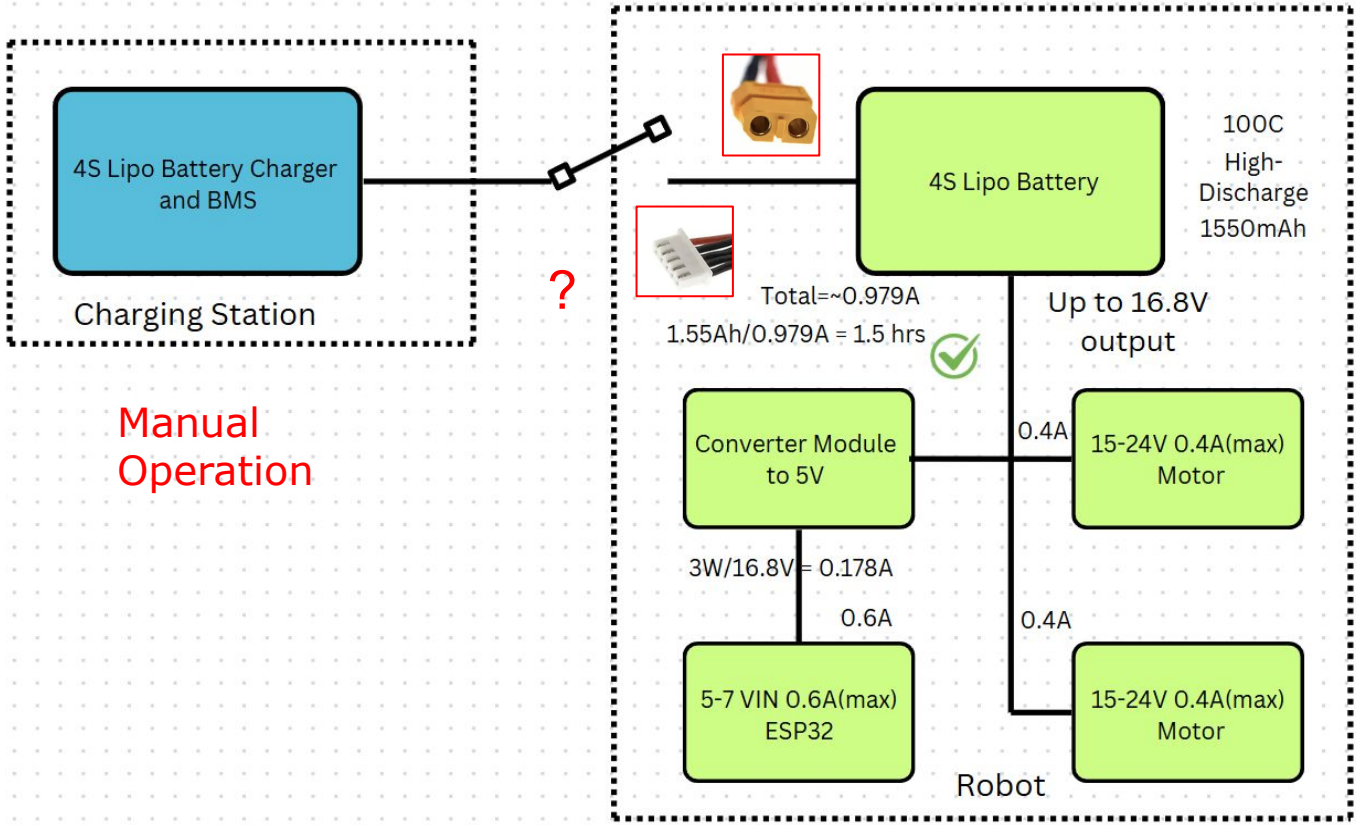
Capabilities of the Jetson Camera



We are using size 1920 x 1080 and framerate 30 fps for the demo

- **Full HD resolution**
- **Low enough processing power and can handle object detection**
- **30 fps means real-time change and no lag. 60 fps requires more resources from the computer**
- **System cannot keep up with the high frame rate**

PDR Power System

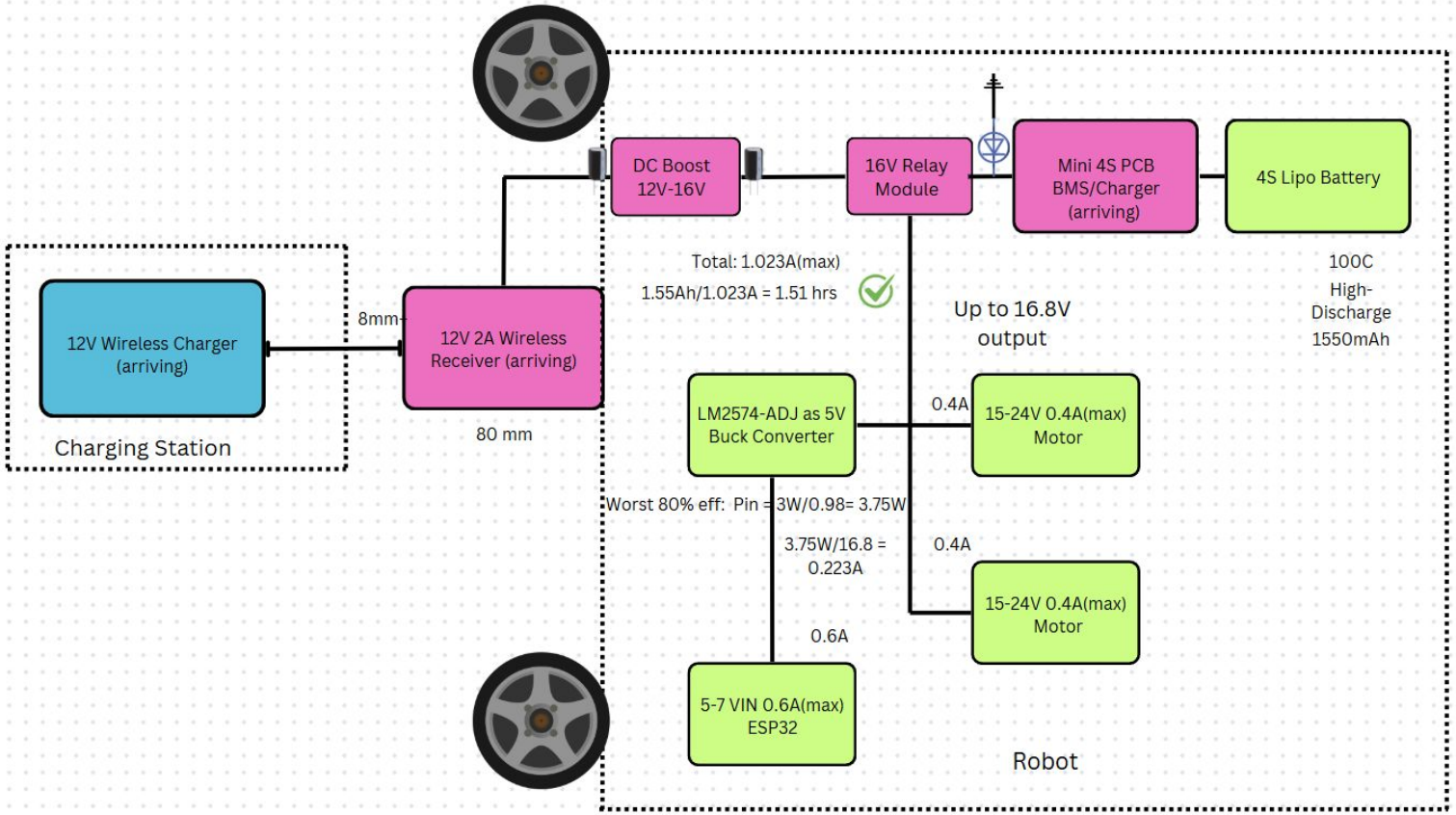


Issues:

- At 4S, both XT60 and JST needed
  - Would need to be held in place and/or extended via other wires
  - Motors would not distribute force correctly if robot just ran into it
  - High enough V for sparks, burned wire; useful materials like steel wool cannot handle

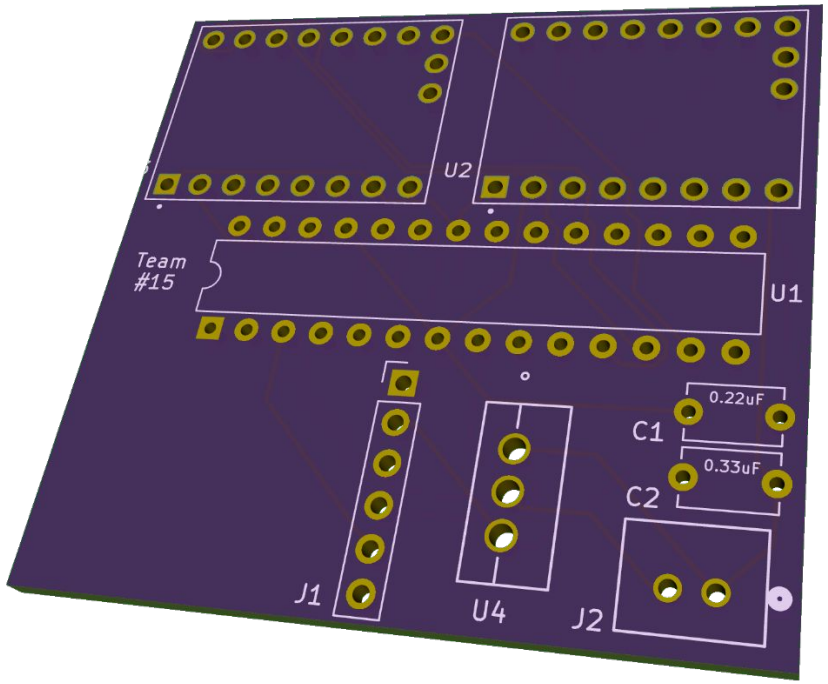
- Wall-connected balance chargers require lots of manual inputs to work; set cells, voltage, mAh…
- Plus not pass-through

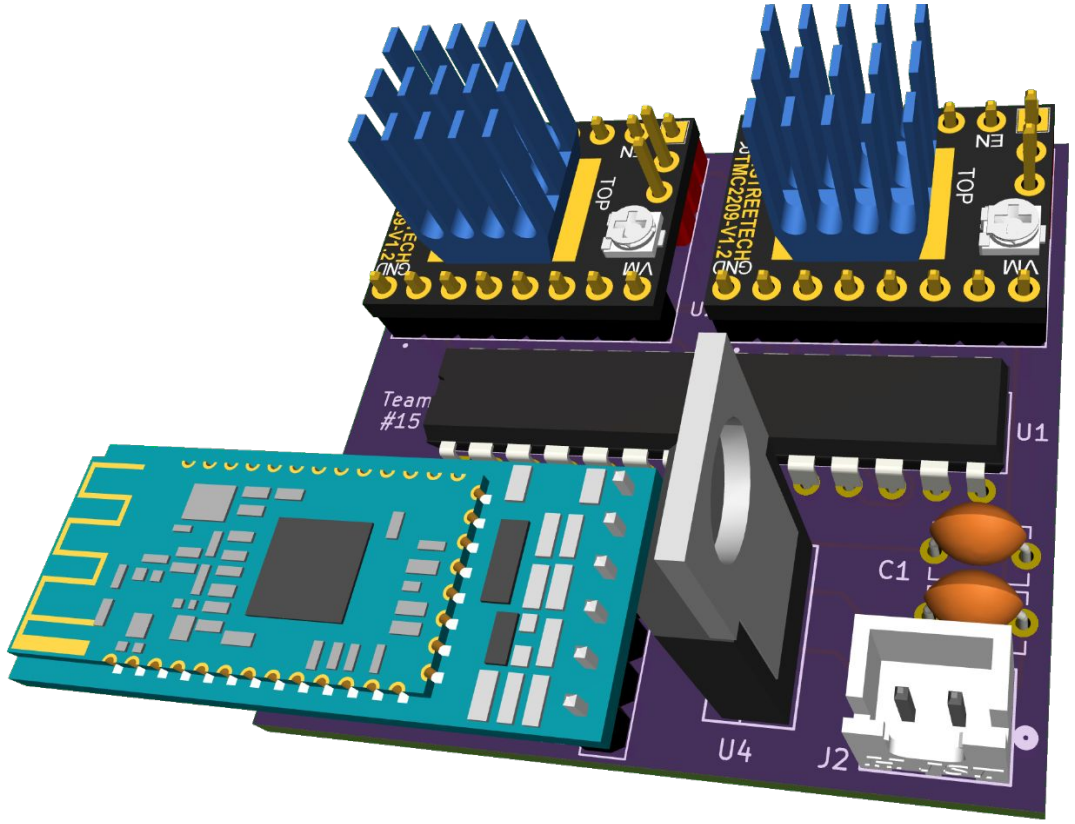
MDR Power System



- No mechanical issues that may require more power drain
- Charger/BMS PCB handles inconsistent voltages
- Receiver fits on bottom of (next) robot model
- When low voltage, relay switches as soon as charging station is reached
- Green = part of demo

PCB Plan (from PDR)

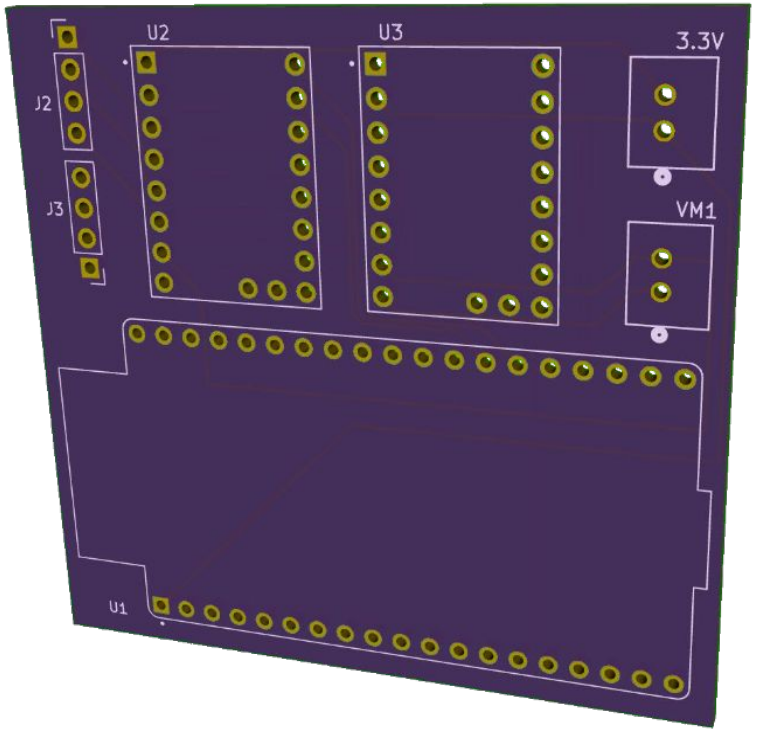


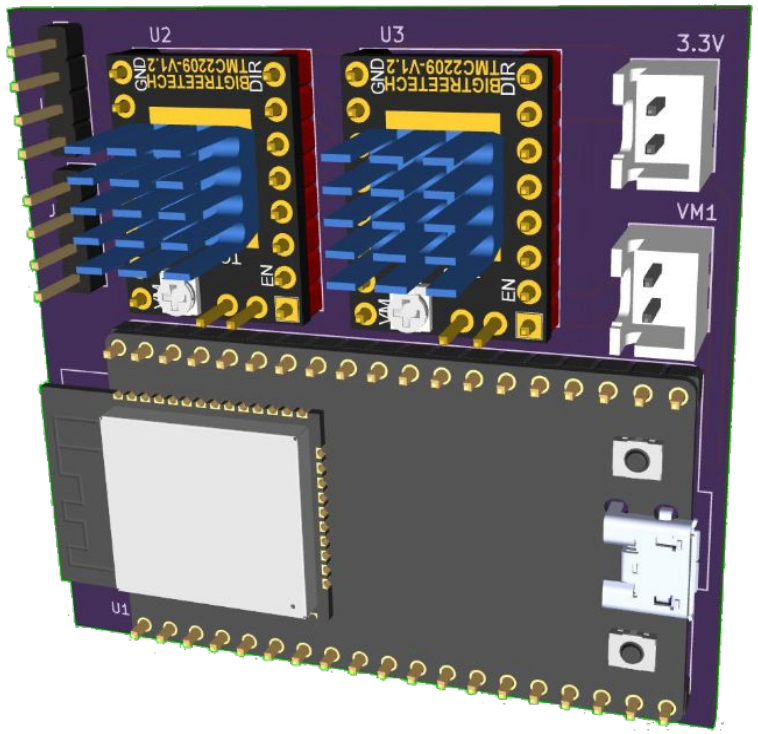


Problems:

- Separate BLE Module (HC-05)
- Limited processing power (ATmega328/P)

Updated PCB Design (MDR)

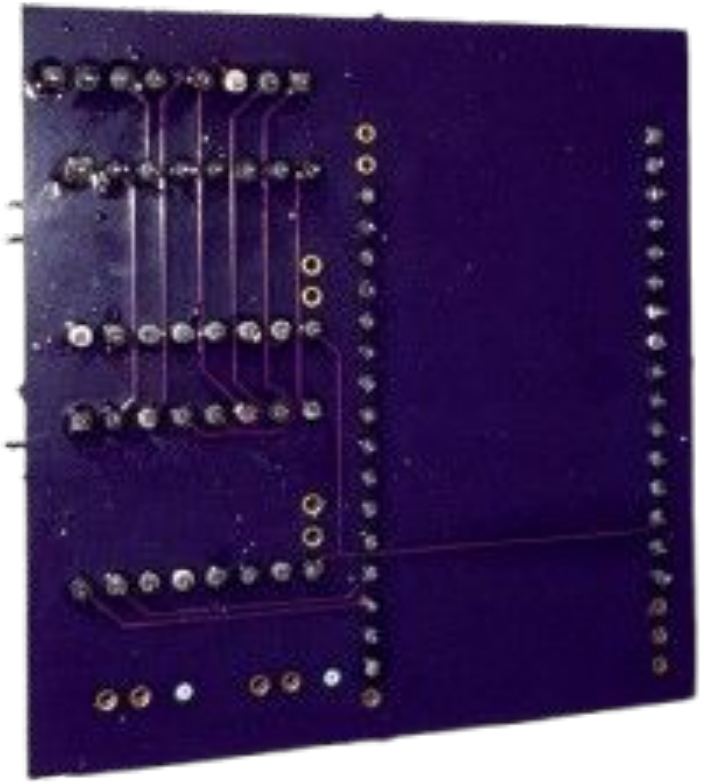




Solution:

- ESP32 combines a powerful MCU with on chip BLE
- 3 UART ports available, for additional features in the future

Current PCB



Solution:

- ESP32 combines a powerful MCU with on chip BLE
- 3 UART ports available, for additional features in the future

Problems with current PCB

- Selected the wrong ESP32 dev-board.
- Designed the PCB for 3.3V circuitry, but the design was later changed to use 5-7V unregulated input.
- Decided to use a single power supply regulated from 15V to 5-7V, eliminating the need for two separate power pins.
- Unable to test with the PCB; the demonstration will be conducted on a breadboard instead.

CDR Deliverables

Overhead Computer + Software

- Meet all remaining MDR Deliverables.
- Translation of calculated path to navigational commands for robot.
- Integration of all above software systems with one another and on the Nvidia Jetson.
- Demo of robot knolling three rectangular objects of weight below the robot's knolling threshold that are scattered across a study desk with ample space in between objects for robot to propel the object in any direction. The study desk will be initialized with a fourth object that must be ignored but considered during path planning.
- Finalized setup of the body of the processor and camera on top of the desk.

Robot

- Robot (including revised PCB design) is able to navigate according to BLE commands from the Jetson.
- Next iteration of the robot body integrates suitable drivetrain.

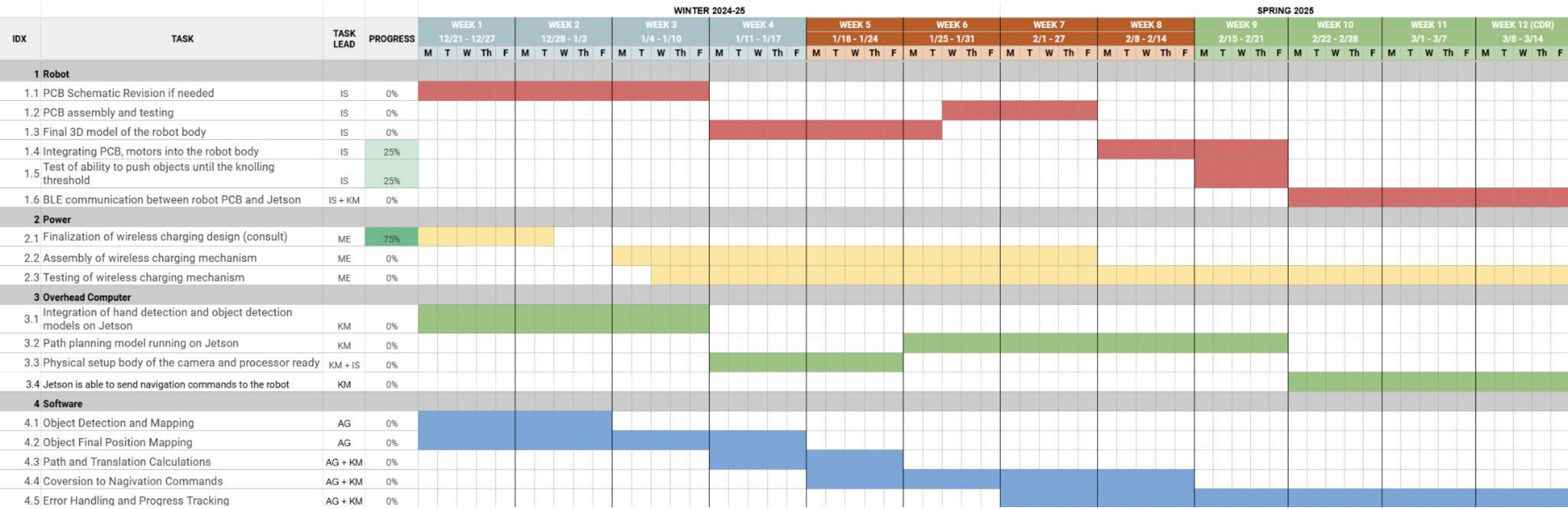
Power Delivery

- Finalize charging mechanism.
- Successfully charge a LiPo (minimum 3.7V) using the finalized charging mechanism.

Project Expenditures (so far)

Item	Cost	Reason	
Tracks	$22.90	For the robot wheels	
Jetson Nano Camera module	$38.42	Camera compatible with Jetson Nano	Legend:
PCB print	$5.03	Robot PCB	ROBOT
PCB shipping	$21.24	PCB Shipping Cost	HARDWARE
NVIDIA Jetson Nano	**	Overhead Computer	MISC.
Total	$87.59		
Remaining	$412.41		

Timeline (To CDR)



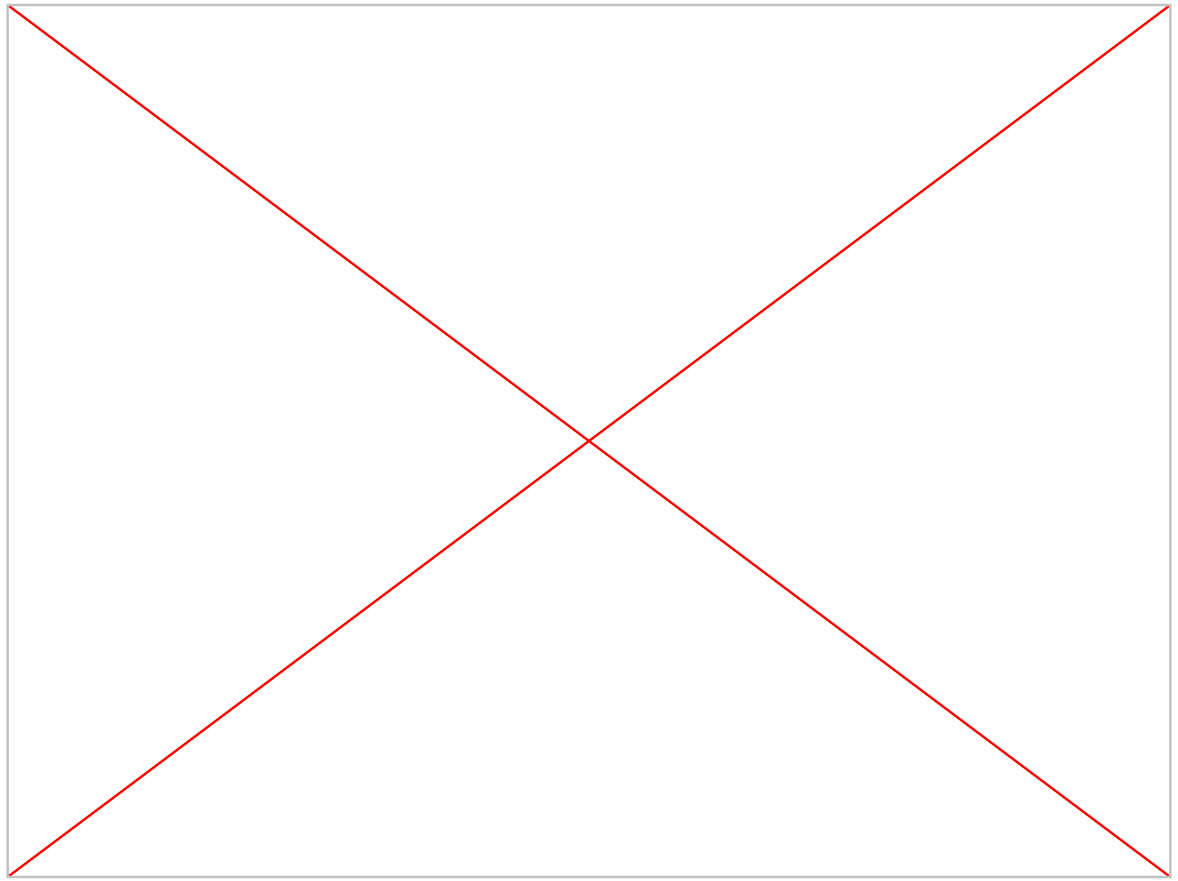
IDX	TASK	TASK LEAD	PROGRESS
1	**Robot**		
1.1	PCB Schematic Revision if needed	IS	0%
1.2	PCB assembly and testing	IS	0%
1.3	Final 3D model of the robot body	IS	0%
1.4	Integrating PCB, motors into the robot body	IS	25%
1.5	Test of ability to push objects until the knolling threshold	IS	25%
1.6	BLE communication between robot PCB and Jetson	IS + KM	0%
2	**Power**		
2.1	Finalization of wireless charging design (consult)	ME	75%
2.2	Assembly of wireless charging mechanism	ME	0%
2.3	Testing of wireless charging mechanism	ME	0%
3	**Overhead Computer**		
3.1	Integration of hand detection and object detection models on Jetson	KM	0%
3.2	Path planning model running on Jetson	KM	0%
3.3	Physical setup body of the camera and processor ready	KM + IS	0%
3.4	Jetson is able to send navigation commands to the robot	KM	0%
4	**Software**		
4.1	Object Detection and Mapping	AG	0%
4.2	Object Final Position Mapping	AG	0%
4.3	Path and Translation Calculations	AG + KM	0%
4.4	Coversion to Nagivation Commands	AG + KM	0%
4.5	Error Handling and Progress Tracking	AG + KM	0%

WINTER 2024-25 | SPRING 2025

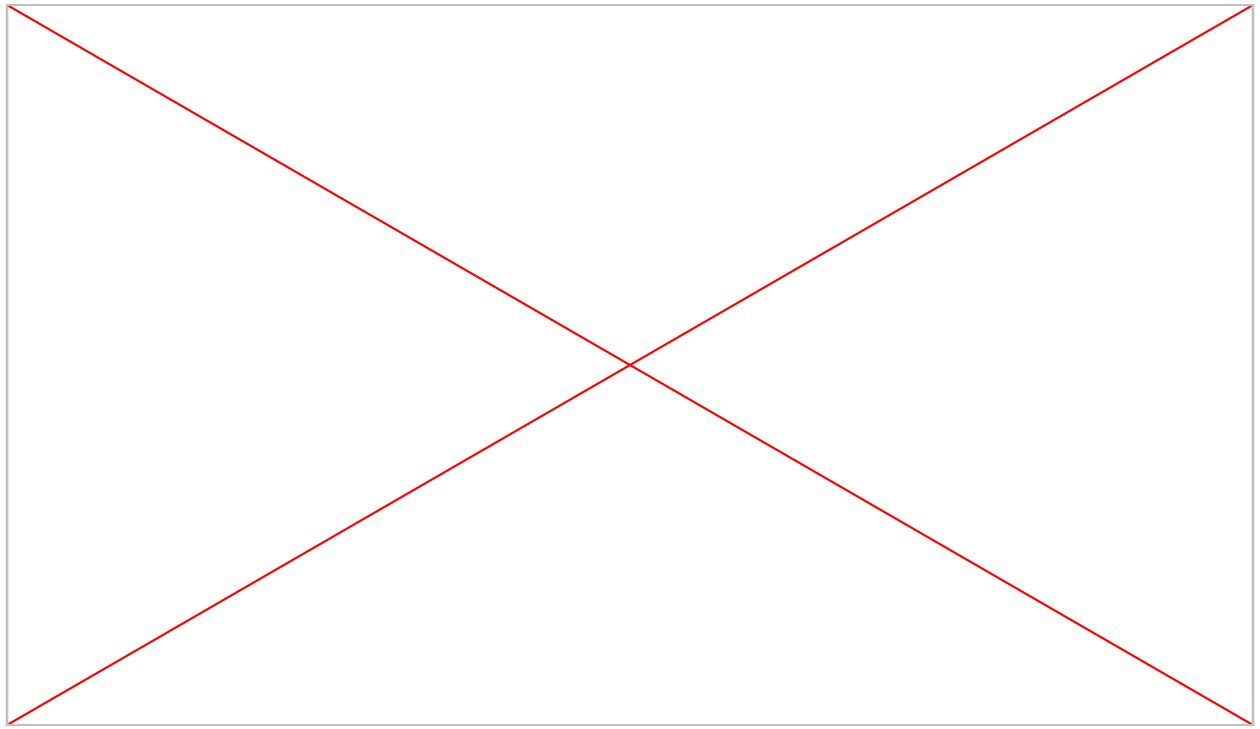
WEEK 1 12/21 - 12/27	WEEK 2 12/28 - 1/3	WEEK 3 1/4 - 1/10	WEEK 4 1/11 - 1/17	WEEK 5 1/18 - 1/24	WEEK 6 1/25 - 1/31	WEEK 7 2/1 - 27	WEEK 8 2/8 - 2/14	WEEK 9 2/15 - 2/21	WEEK 10 2/22 - 2/28	WEEK 11 3/1 - 3/7	WEEK 12 (CDR) 3/8 - 3/14
M T W Th F	M T W Th F	M T W Th F	M T W Th F	M T W Th F	M T W Th F	M T W Th F	M T W Th F	M T W Th F	M T W Th F	M T W Th F	M T W Th F

https://docs.google.com/spreadsheets/d/13YVHZK6UHJRp5mrNjDS3swXoNIhkR-2ZMn9xTu6tttw/edit?usp=sharing

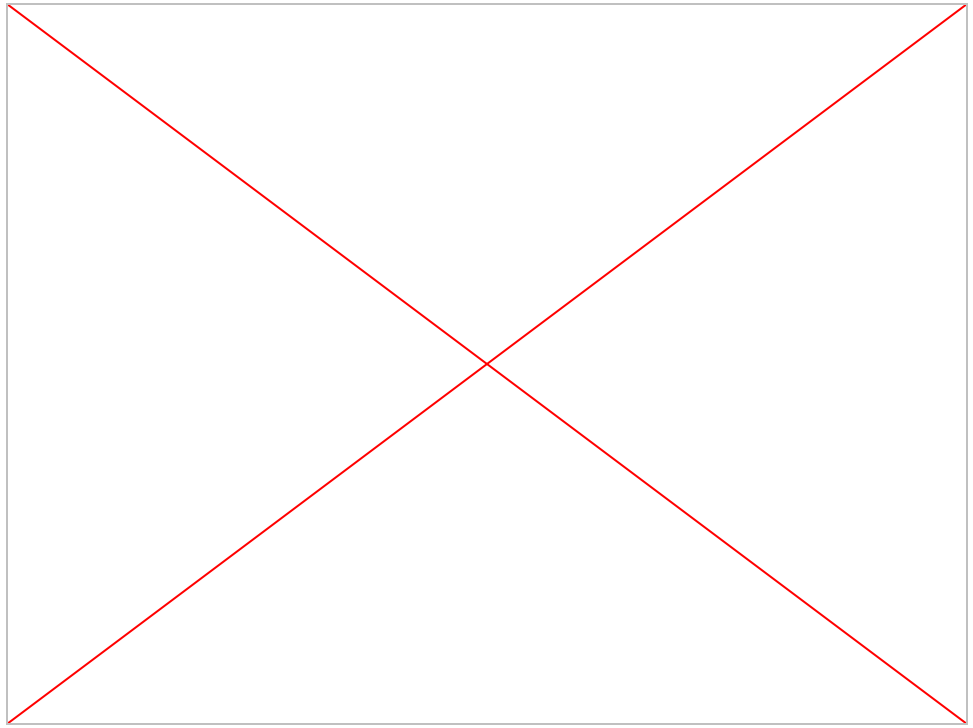
Demo Backup Video: Robot Movement



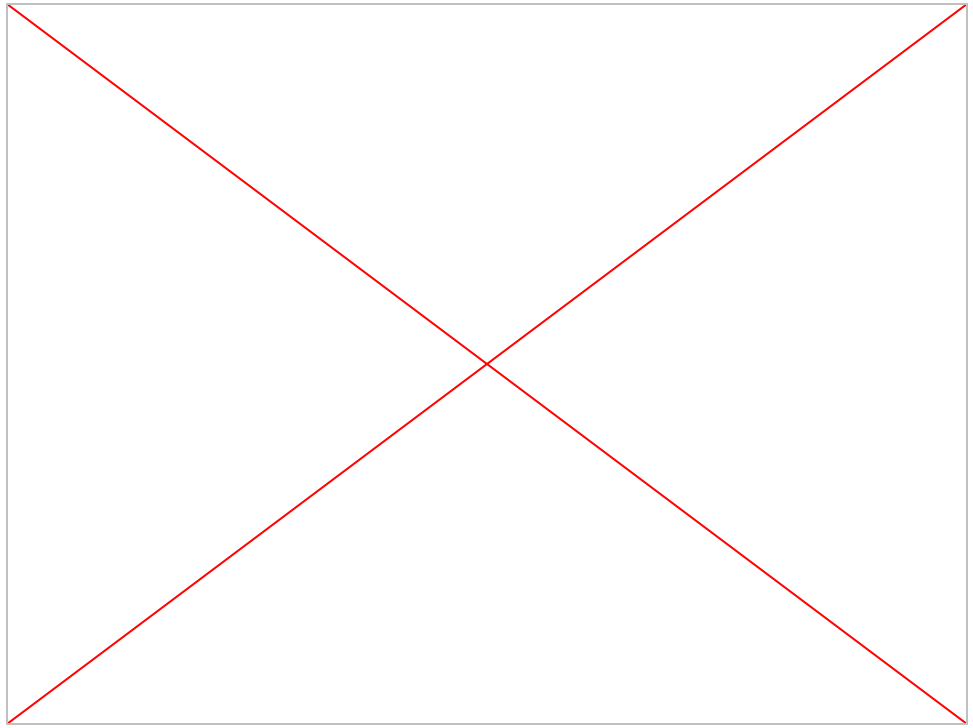
Demo Backup Video: Jetson Camera



Demo Backup Video: Object Detection



Demo Backup Video: Object Detection with Remote Ignored as not a “knollable” object



Thank you!
Any Questions?