Autonomous Debris Collector Design Report

Andy Zhang, Hojun Byun, Omar Alinur
Electrical and Computer Engineering, Carnegie Mellon University
{andyz, hojunb, oka} @ andrew.cmu.edu

Abstract
Workshops are a place where safety is the most important standard. However, tools and items frequently fall onto the floor and present a safety hazard for workers. Our autonomous debris collector uses computer vision techniques to quickly and efficiently collect fallen debris from workshop floors and ensure workers do not need to devote extra time to scan the floor during work and pick up dropped tools.

Index Terms
Jetson AGX Xavier, OpenCV, iRobot, Universal Gripper, Localization,

I. INTRODUCTION

People working in mechanical workshops drop tools on the floor very frequently. If left ignored, the these tools can become problems for the following reasons:

1. People can harm themselves by stepping on or tripping over the tools.
2. Picking up these items can be time-consuming, tedious, and labor intensive.
3. Based on different lighting environments, these items can be difficult to detect with the human eye.

Our project aims to solve the problem by creating an autonomous robot that would pick up and collect these tools that fall on the floor. Named Autonomous Debris Collector, our robot will eliminate the potential for people to hurt themselves, waste time on picking up after themselves, and miss picking up all the fallen objects due to their poor eyesight.

With Autonomous Debris Collector (A.D.C. for short) the problem will become nothing but an afterthought as we aim to construct a robot that can quickly and autonomously collect small tools that have been marked with a distinct color at a rate of 2 tools per minute while having a 20 minute battery life.

II. DESIGN REQUIREMENTS

In order to achieve reliable autonomous small item retrieval in a timely and reasonable manner, we have set these following high-level requirements:

1. Reliable item pickup - The robot must achieve at least a 90% success rate when attempting to pick up items.
2. Item size - The scope of our project is aimed at items that are less than 0.7kg and 10cm in length so that the gripper has enough surface area and strength to effectively grasp the items.
3. Item Collection - The number of items collected should not exceed 10 so that there is enough room in our basket to carry the items without risk of them falling out or causing too much strain on the chassis and motors.
4. Operation time - The robot must have at least 20 minutes of operation time to ensure collection of the dropped objects in a workshop of moderate size (84 square meters) without the robot running out of power
5. Robot weight - The robot must weigh less than 10kg so as to not put too much strain on the wheel motors and cause the run time to be shortened due to the increased torque required by the motors
6. Robot speed - The robot must be able to travel at 0.4 m/s in order to efficiently search the workshop and find the dropped items before the robot base runs out of power.
7. Arm control accuracy - The robot's arm must have an arc length accuracy of at least 5cm so that the arm's position relative to the

item and basket can be accurate enough to reliable pick up and drop the item in the basket

8. Image processing latency - The robot must have a maximum of 10ms in latency between each item localization timestep so as to ensure proper navigation towards the item
9. Image processing reliability - The robot must have a 0% false positive rate and less than 7% false negative rate in order to ensure the correct items are being tracked with some wiggle room in the case of noisy lighting conditions
10. Path planning latency - The latency between each path planning update should be at most 50ms in order to maintain a steady trajectory towards the item

III. ARCHITECTURE AND/OR PRINCIPLE OF OPERATION

The architecture of ADC can be divided into two parts: the software components to perform computer vision and motion planning, and the hardware components doing the physical act of picking up the objects. This idea is represented by figure 2 on page 10.

The software would be handled by the Nvidia Jetson AGX Xavier, which is responsible for performing the computer vision, motion planning, the arm motion, and controlling the robot base.

The process would begin with the Intel RealSense Depth Camera D435i, where images are captured and sent to the Xavier for image processing, which would obtain the coordinates of the center of every object that is detected marked with our desired color. The Xavier would do this by first applying a gaussian blur, then converting the image to an HSV format, then running a color thresholding algorithm, then finally finding the contours and the centroids of each object. With these coordinates, the Xavier then runs a motion planning algorithm to determine where the robot would go by checking if the closest object to the robot is close enough to grab. If the object is not close enough to grab then the Xavier would send signals to the iRobot base to control the wheels of the base to move towards the object at a top speed of 0.4m/s. If the object is close enough to grab then the hardware components would come into play.

Once the hardware components come into play, the Xavier would send signals to the two servos and the electric air pump to execute the pick-up protocol. To start, the 20kg digital servo would rotate our single-jointed robot arm downwards to deform the rubber pouch around the object. Afterwards, the Xavier commands the air pump to apply suction to the pouch, hardening it around the object and then command the previous motor to raise the arm back up. Then, the Xavier commands a different servo of the same model to rotate the arm's base towards the storage bin located on a wooden platform that would be hoisted above the robot base. This way, a signal can be sent to the air pump to pump air back into the pouch to reshape it, releasing the object on top of the storage bin. Finally, the second servo would rotate the arm's base back to its original position.

IV. DESIGN TRADE STUDIES

After going through several design iterations, we have weighed the benefits and detriments of each design choice to be made to come to our final design.

Hardware

A. Robot Base

Our initial plan revolved around making our own robot base that is large enough to collect all sorts of tools in a workshop ranging from small screwdrivers to large electric drills.

We soon realized that our initial scope for our project was too ambitious. So after lengthy conversations about changing our scope, we decided to build a much smaller robot that would focus on smaller tools to collect. This led us to the iRobot Create 4400. While building our own robot base would have given us more mechanical control over the parameters of our robot as we would be the ones constructing it, we came to the conclusion that we did not have enough experience in robotics to meet our requirements had we gone with this approach. Therefore, picking a robot base that is easy to program such as an iRobot is the best course of action.

B. Gripping Mechanism

Originally, we wanted to go with a two finger gripping mechanism but realized that that would require accurate positioning and unobvious point of contact determination. This led us to a universal gripper design which is more robust and simpler. The suction-based universal gripper that we decided on is well tested and can achieve a firm grip on a wide range of shapes and sizes.

The gripping mechanism for our robot needs to be able to grab tools of different shapes and sizes. This is why we decided to go with a universal gripper design as shown in Figure 1. The blue component is a flexible rubber pouch that is attached to a funnel and would be filled with finely-ground material. The funnel would have an electric pump/vacuum attached to the back end of it.

In order to grab an object, the gripper will:

1. Deform the rubber pouch around the object.
2. Use an electric pump/vacuum to suck the air out of the pouch, hardening the rubber around the desired object.

After these two steps, the desired object would be firmly embedded into the pouch, allowing easy manipulation for storage.

In order to release a grasped object, the gripper will simply pump air back into the pouch, returning the pouch to its original shape to release its hold on the object.

For construction, we need a rubber pouch, finely ground materials, a funnel, an electric air pump and vacuum, and a piece of cloth to make sure the pouch is sealed.

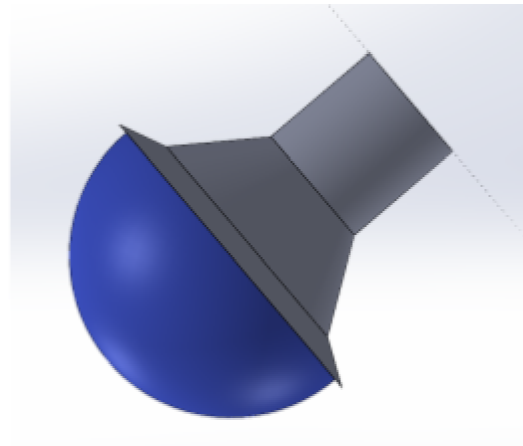

Figure 1: CAD Model of Universal Gripper

C. Camera

To assist in our item localization, we decided to use an Intel RealSense D435i camera since it has accurate RGB-D imaging, high resolution, high FPS, decent built in IMU, and comes with easy to use libraries that can do image filtering, calibration, and SLAM. Additionally, there is plenty of documentation which describes how to fuse the localization output from the camera with our wheel odometry through ROS.

D. Robot Arm

To rotate the arm that controls the gripper, we decided on an affordable torquey servo rated for 20kg which has enough power and accuracy to meet our requirements. We also decided to have the motion for gripping and storing the objects be as simple as possible. To accomplish this, the robot's arm is a single joint that would rotate about two axes of motion: one to move straight down to pick up the object in front of the robot and the other to rotate the arm to face the storage bin that is integrated into the chassis that is located next to the arm to release the object once the arm is rotated.

E. Chassis

The robot chassis should have the main goal of being robust enough to house all of our hardware components while minimizing cost. The plan that we have had throughout the design process revolved around 3D printing a chassis that would be placed over the iRobot base. But we came to the conclusion that a much simpler design would get the job done better as 3D printing a robust enough chassis would be difficult, especially in the case that the chassis happens to break and we would have to order and wait for a new one from TechSpark. The design that we have decided on is a 30cm x 20cm x 3cm wooden platform that would be hoisted above the robot base by 12cm using 4 stainless steel standoffs. We decided that this was the best design because we realized that the size of the robot base itself would not be able to house all the components, so we had to expand vertically. Most of the hardware components would have to lie on top of the robot base such as the GPU, CPU, the powerbank, the robot arm, and the air pump. The wooden platform's purpose is to give space for the storage bin as well as being a high-platform for the camera to have a slanted angle to look downtowards.

Software

A. Computer

For the robot's computer, we decided on the Nvidia Jetson AGX Xavier since it has ample processing power to support our computer vision and path planning software components. We initially planned on using the Jetson Nano, but after input from the course staff we realized that we could use the Xavier, a much more powerful GPU, for free since it was from the course parts list.

B. Localization

At first, we were going to manually fuse our sensor data to estimate our pose via an extended kalman filter. We then decided to use ROS packages instead since all that would be required to determine our poses would be specifying our motion model and then publishing our sensor data to the correct ROS topics.

C. Arm pick-up protocol

Our initial protocol was to have a two-jointed arm that would have a diverse range of motions to pick up different objects of different sizes and shapes. But we realized that this would be very complicated for us to implement with our given time as none of us had programmed 2D reverse kinematics motion for a robot arm before. So, we decided on the simplest design possible of having a single-jointed arm that would rotate about two different axes of rotation, one for picking up objects and one for rotating the arm towards the storage bin.

V. SYSTEM DESCRIPTION

Our system is divided into two major components: the physical system responsible for grabbing desired objects, and the software component that is responsible for item detection and motion planning.

A. Depth Camera

Our team looked at several different cameras during the design and planning process. We eventually settled on the Intel RealSense Depth Camera D435. We settled on this camera because it provided three important features: depth-based real-time recording, a wide field of view, and a 1920 x 1080 resolution. The high resolution allows the camera to properly focus and reduce any pixel noise in the real-time frames. This allows us to greatly trim down the possibility of false positives and false negatives due to pixel distortion. On the other hand, the depth-based and wide field of view (90 degrees) allows the camera mounted at the top of the robot to scan the floor both at short and long distances for debris. As the location of dropped objects is entirely random and across a large area, we need to make sure that the camera can see it. Finally, the camera is able to provide information about the "depth" of each pixel in near real-time (90 fps). By providing the depth of each pixel, we can much more easily optimize our path-finding and color-detection algorithms to identify the tools and see how far away they are. This also removes a lot of risks and uncertainty for detected objects' distance that could fluctuate due to lighting changes or angled views.

B. *Image Processing*

Our computer vision software is powered by OpenCV which has the necessary libraries to recognize and localize the marked items while having plenty of documentation as to how the libraries can be used. To localize the marked objects we will perform the following procedure: Gaussian blur the image for smoother image processing, transform the image to HSV to make the markers stand out, perform thresholding to remove everything in the image but the markers, find the contours around the markers, find the centroids of the contours, and finally add the markers' centroid positions to a list to be used for path planning. In order to differentiate the items, we will store their location in the terms of the global frame. All of the described procedures are accomplished through OpenCV functions.

C. *Localization*

To localize the robot, we are using ROS which has libraries that can fuse our depth sensor, IMU, and wheel odometry while providing an easy to use platform for performing our image processing. All that is required by ROS to localize is the correct motion model description, odometry publishing, depth camera publishing, and IMU publishing. There is a ton of documentation on how to do this in ROS which makes the localization task a lot easier than if we were to do it from scratch without using ROS.

D. *Path Planning*

We will be writing our own path planning algorithm which will first move the robot throughout the workshop while noting the positions of all the objects it encounters. To move throughout the workshop, we will move in a lawnmower fashion. In order to avoid objects, we can force the robot to maintain a two foot distance from any obstacle by checking to see if the depth sensor detects an object that is within two feet of the robot and making the necessary heading adjustments. Once all the marked objects are localized, we will then move to each one based on dijkstra's algorithm so that we move to each item in the fastest way possible.

E. *Simulation*

In order to test our code while the robot is being developed, we are using a Gazebo simulator. Since Gazebo is meant to interface with ROS and contains an advanced physics simulator, along with models of our iRobot base and depth camera, integrating our code on the real robot should be a lot easier.

F. *Gripping Protocol*

When picking up detected objects, the arm would run through the following pick-up protocol everytime:

1. A motor would rotate the arm down so the gripper deforms around the item.
2. Air is suctioned out of the pouch to harden it.
3. Arm rotates back up using the previous motor
4. With the second motor the arm's base rotates towards the storage bin.
5. Air is pumped back into the pouch to reshape it, releasing the object into the storage bin.
6. Using the second motor the arm's base rotates to its default position.

This protocol allows us to build a very simple system for the arm.

VI. PROJECT MANAGEMENT

A. Schedule

We have set 3 phases for our project:

1. Project Proposal and Planning
2. Learning, Design, and Implementation
3. Performance Testing and Refinement

The first phase is already completed and during it, we brainstormed and honed in on what our project should look like at the end of the term. We are currently in the second phase where we are researching and learning about the specific details needed to complete the project. In the next month and a half, we plan to implement the individual features and integrate them into the robot. For our third phase, we will test and tune our system.

B. Team Member Responsibilities

Each team member is responsible for keeping each other accountable on their individual tasks. We have split up our work based on our previous project experiences as well as who is physically present in Pittsburgh or not, and there are three major components to the project: robot construction and motion control, computer vision, and localization and path planning. These tasks and their subtasks are divided up amongst the three members as shown below.

Team Member	Primary Responsibility	Secondary Responsibility
Hojun	Construct the arm/gripper and integrate the GPU/microcontroller with the robot base	Testing of gripper's reliability, robot's travel speed, item collection capabilities.
Omar	Motion model and sensor fusion for localization and path planning	Simulation, integration, and testing of each software component on the robot
Andy	Computer vision algorithm to detect tools marked with distinct colors	Help test tool/color detection performance and lighting variability

C. Budget

The bill of materials is included in page 11 in the Appendix.

Of the hardware materials we required, we have obtained the Nvidia Jetson AGX Xavier and the iRobot Create 4400 robot base for free from the 18-500 course parts list.

D. Risk Management

During the course of our project, we have already encountered several potential risks for the design and development of our robot.

Firstly, we considered the wide assortment of tools that could be present in workshops and fall onto the floor. Wrenches, nails, screwdrivers, sockets, clippers, and other tools all have very different weights and shapes. Therefore, we needed to ensure that our gripper could tackle all these types of items. To do so, we conducted gripper tests on various tools of varying shapes and sizes and in different positions on the ground. We used an air pump to power our gripper and tested rubber, nylon, and latex as the covering material. We eventually settled on rubber because it was able to pick up all the tools we tested it on and did not show any signs of wear or tearing. We intend to conduct more gripper material testing as we continue building our robot and encounter more types of tools and test cases.

Secondly, we considered the ability of our universal gripper to grab various types of tools and fallen objects on the workshop floor. While the gripper was tested to work on more "softer" objects and smaller tools, there was always the possibility that sharp edged tools (nails, screwdrivers, etc.) could potentially penetrate the soft surface of the gripper and render it useless. Additionally, heavier or more slippery objects like sockets would be harder to pick up using suction. One solution we consider is attaching a solenoid-head to ensure that all dropped magnetic items are picked up. This meant that we sacrifice some weight on the gripper's top but ensure that most metal tools are able to be picked up.

Thirdly, we were also worried about the robot's stability when the arm rotates to pick up objects. Due to the length and extended nature of the arm, the robot's center of mass could shift whenever the arm is swinging around and grabbing objects. To counteract this, we plan to construct our arm with as less mass as possible to offset the weight balance and potentially add a weighted counterbalance on the opposite side to ensure that the robot maintains a stable position.

Finally, we considered the challenges and risks associated with the computer vision portion of the project. Since workshops have a large variation in lighting both overall and in various places that the robot could travel to. To prevent this risk, we decided to use OpenCV and use auto-calibration to automatically adjust the color threshold every use. Additionally, we set our use case to bright and well-lit environments to avoid any instances of insufficient or distorted lighting.

To combat the challenges of having a remote semester and team members being separated physically, we made sure to set weekly team goals and keep each other updated during the scheduled team meetings and an additional internal meeting every week. That way, we are able to meet our deadlines and help each other whenever obstacles arise.

APPENDIX

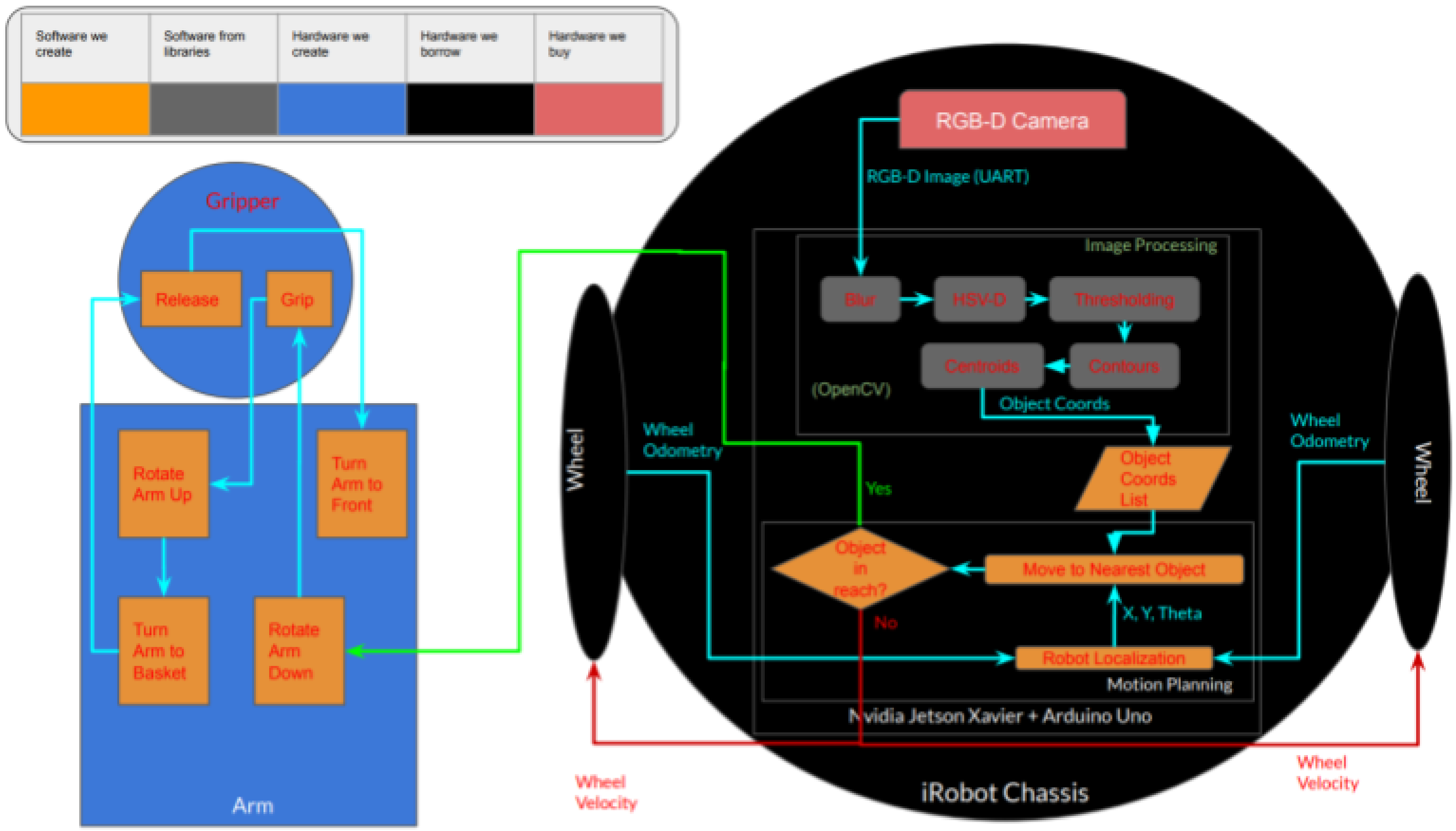

Figure 2: System Block Diagram

Bill of Materials

Part	Website	Notes	Shipping Cost	Base Cost + Tax (per quantity)
Depth Camera	azon.com/Intel-RealSense-Depth-Camera-D435i/dp/		0	210.52
iRobot Create 4400		Robot Base	0	*Taken from Course Parts list
Nvidia Jetson AGX Xavier		GPU to handle computer vision and motion planning	0	*Taken from Course Parts list
-8 Stainless Steel Flush-Mount aptive Nut, 4-40 Thread Size, .062" min Panel Thick	https://www.mcmaster.com/catalog/127/3470/	Can let us screw stuff into the roomba. 25 per pack	0	8.56
8 Stainless Steel Male-Female eaded Hex Standoff, 1/4" Hex, 2" Length	https://www.mcmaster.com/catalog/127/3536	Standoff from the case so we can have clearance between the case and box for the pump and other devices. Might need larger standoffs depending on pump size etc.	0	2.39
ow-Profile Button Head Torx achine Screw, 18-8 Stainless eel, 4-40 Thread, 3/8" Length	er.com/machine-screws/18-8-stainless-steel-button-	Screws into threads. 25 screws per pack.	0	9.08
arge Round Latex Balloons	om/AnnoDeel-Balloons-Birthday-Decorations-Assor	Flexible rubber for the gripping mechanism	0	18.99
Pump and Vacuum DC Motor - 4.5V and 1.8 LPM	eBBhDiARIsADyBcE6Yh_PC1nYLoGNTc4jP3RusT	Two airpump system for vacuuming and pumping air into the rubber pouch	0	6.95
20kg Digital Servo	6CNKQX4&pd_rd_r=595e0a29-e931-4b50-a2fc-1b2	Two servos for two-servo system for moving the arm	0	16.99
Silicon Tubing (1m)	4661?utm_source=youtube&utm_medium=videodes	Tubing for the gripping mechanism	0	2.5
-Connector for Silicon Tubing	4662?utm_source=youtube&utm_medium=videodes	flexible rubber for the gripping mechanism for the gripper	0	4.5
ir Pump and Vacuum Valve	4663?utm_source=youtube&utm_medium=videodes	Valve for controlling airflow for the gripper	0	2.95

Figure 3: Bill of Materials

Schedule

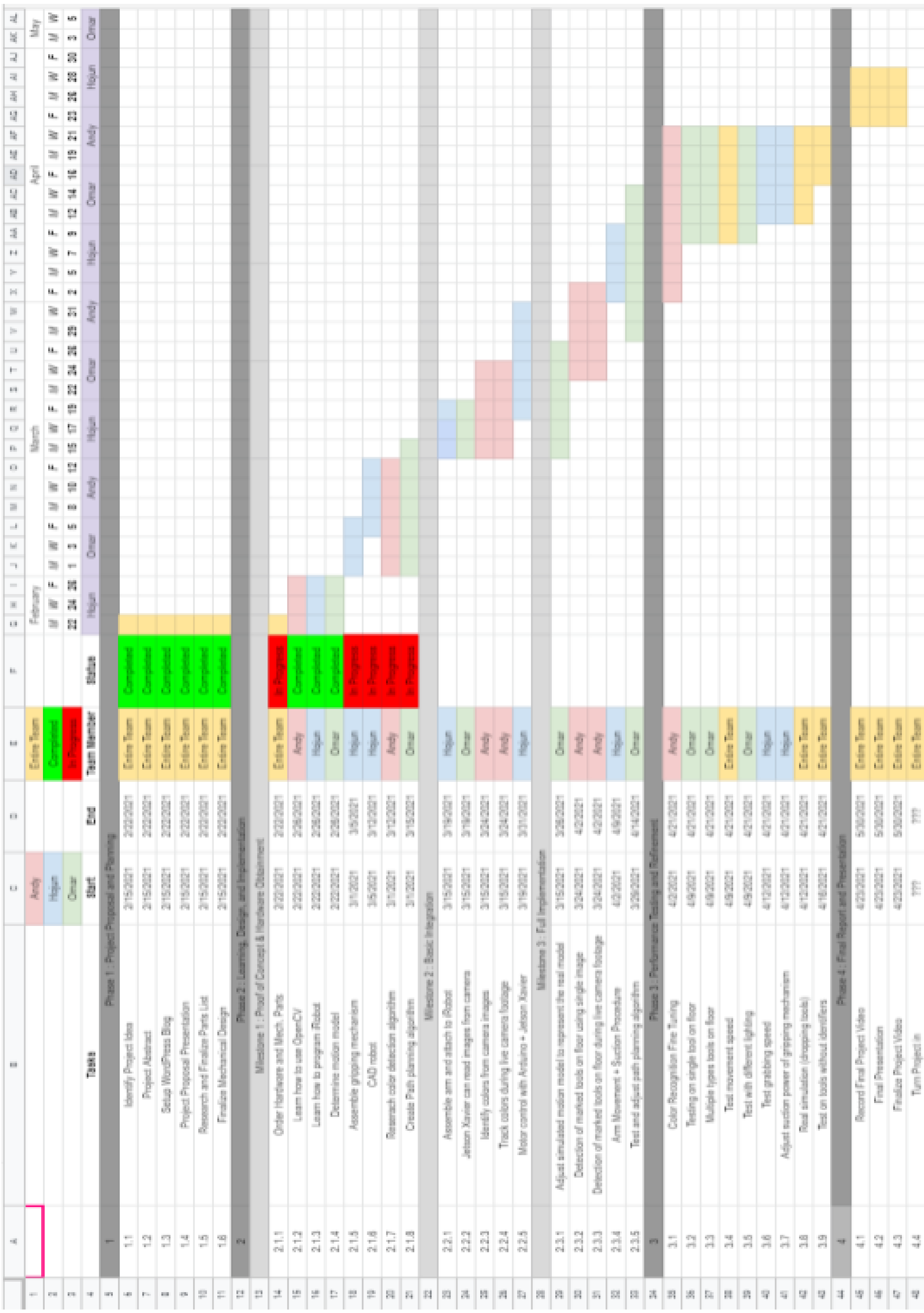

Figure 4: Gantt Chart