Carnegie Mellon University

16-681

Project Course I

Critical Design Review Report
COBORG

Team C:
Gerald D'Ascoli | Jonathan Lord-Fonda | Yuqing Qin | Husam Wadi
Feng Xiang

Sponsor:
Biorobotics Lab

May 13, 2021

Abstract

The COBORG system is a robotic backpack aimed at helping manufacturing workers hold and secure parts overhead. It accomplishes this goal by receiving voice commands from the user and identifying the target location with a vision system that tracks the user's hands. The vision system also provides localization for the robot so that the actuated manipulation system can bring the end-effector to the location identified by the user's hands, regardless of the user's shifting during the COBORG's operation. Currently, the three primary systems (vision, voice, and actuated manipulation) have been implemented and validated according to their respective requirements. The state machine that connects them has also been implemented, alongside a rudimentary version of the fully automated system that can function but lacks the key features of localization, stabilization, and reasoning. Future work for the COBORG backpack includes full integration of the current system with intelligent actuated manipulation, validation of the said system, the addition of collision detection, an extra motor for the arm, an aesthetic overhaul, and an added functionality.

Table of Contents

1. Project Description

Automotive and airplane manufacturing can often cause strain to the operator's arms after working for long periods of time, and the operator often requires a second person to assist them as they work [1] [2]. Specifically, the attachment of under-wing panels requires two individuals, one to hold the panel, and the other to attach it. A robotic backpack system could hold the panel for its wearer, allowing the operator to use their hands to secure the part. This Collaborative Cyborg Backpack Platform (COBORG) would allow a single individual to accomplish the entire task by themselves, and alleviate the stress and strain-induced throughout the workday. To accomplish this operational objective, the COBORG would have to be simple, accurate, and hands-free.

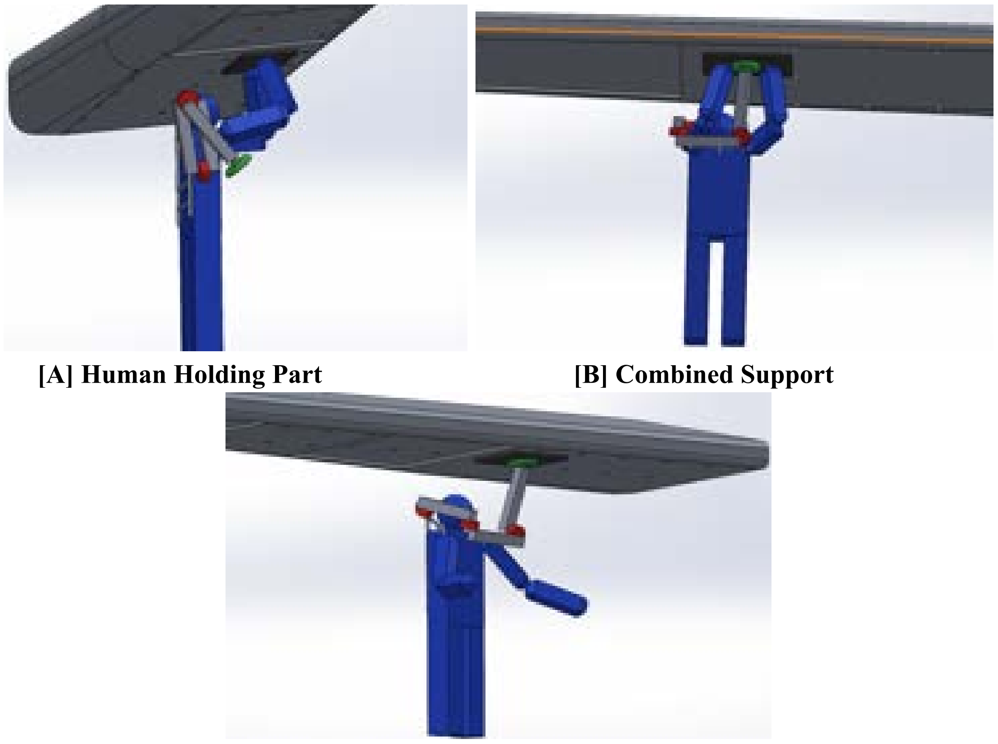
2. Use Case

2.1. Narrative

Jason works on the assembly of airplane wings at Boeing. When he gets to work he checks his to-do list and sees that his list is topped with a series of tricky assemblies. Knowing that he will require aid shortly, he walks over to the COBORG backpack arm station and signs out one of the units. Picking the unit up from its charging station, he straps the backpack on, adjusting the straps for comfort, and heads over to the plane he will be working on today. After completing some remedial tasks he is ready to move on to the trickier cases where he will require the arm's help. He switches on the backpack arm, which has been in a compact position and not using energy up until this point. Jason grabs the part he requires assistance with and holds it up over his head, fitting it into place (see Figure 1[A]). When the part is stabilized, he says, "COBORG, hold it," and the COBORG backpack arm detects his hands in 3D space and moves the robot arm to a position where it can push on the part and stabilize it (see Figure 1[B]). Jason lowers one of his arms from the part, now that the COBORG backpack arm is holding it, and uses a drill with his free hand to screw the part into place (see Figure 1[C]). While Jason's body shifts its position, the COBORG backpack arm adjusts to maintain the position of the end effector in 3D space, supporting the part regardless of Jason's position within the limits of the arm, with 5 degrees of freedom. Now that the part is fastened, Jason says, "COBORG, return home." The COBORG backpack arm returns to its compact position and goes into sleep mode, awaiting further instruction with minimal power usage.

Jason finds the next part for the next task and uses the backpack arm to help him secure it as well. After completing a series of similar tasks, Jason comes to a panel he will have to connect located in a dark area of the interior. Jason quickly switches the end effector of the COBORG backpack arm from the support paddle to the gripper, handily attached to the side of the backpack. He then locks his flashlight into the gripper, moves the arm into place so that he can see the necessary location, and says, "COBORG, stay." The COBORG robot arm maintains its position and orientation in 3D space while Jason moves around, reaching for the panel he can now see and finishes his task. He then says, "COBORG, return home," removes the flashlight from the gripper, and switches back to the support paddle for his next series of tasks. After completing all of his tricky tasks for the day, Jason returns the COBORG backpack arm to its charging station and signs it back in. While Jason completes the rest of his work for the day, the COBORG backpack arm charges, awaiting its next user.

2.2. Graphical Representation


[A] Human Holding Part

[B] Combined Support

[C] COBORG Hands-Free Support

Figure 1 - Graphical Representations of Use Case

3. System-level Requirements

Mandatory and desired performance and non-functional requirements are organized into their respective subsystems and shown in the subsections below. System-level requirements that do not pertain to more than one or all subsystems can be found in Table 1 below. Vision subsystem requirements can be found in Table 2 below. Voice subsystem requirements can be found in Table 3 below. Actuated manipulation subsystem requirement can be found in Table 4 below. Hardware, electrical, and sensor frameworks are conglomerate into one subsystem for the purpose of this requirements section and can be found in Table 5 below.

3.1. System-level Requirements

Table 1 - Hardware, Electrical, and Sensor Subsystem Requirements

ID	Requirement
N.M.4	Will operate safely.
N.M.5	Will be simple to operate.
N.M.6	Will be able to perform untethered for 20 minutes.
N.M.7	Will require minimal part modification to assist with assigned tasks.
N.M.8	Will be operable on a portable computer.
N.D.1	Will be able to operate standalone (no WiFi).

3.2. Vision Subsystem Requirements

Table 2 - Vision Subsystem Requirements (Changes Highlighted in Yellow)

ID	Requirement
P.M.1.1	(Detect indicated parts) Will have 60% accuracy of detecting indicated location within 6" in 3D space, and always within 12".
P.M.1.2	(Calculate object) Will detect the intended object within 5 seconds of when the move command is issued.
P.M.1.3	(Pose Detection) Shall detect the surface normal of the part with an error no greater than 45°.
P.D.1.2	(Texture Invariant) Must be invariant to part texture, specifically matte finish and gloss finish.

The only requirement from the vision subsystem that changed between the PDR and CDR was P.M.1.1. The addition of "always within 12" was added to place a worst-case maximum boundary on the acceptable error from the system.

3.3. Voice Subsystem Requirements

Table 3 - Voice Subsystem Requirements (Changes Highlighted in Yellow)

ID	Requirement
P.M.4.1	(Voice command) Will be able to understand the voice command 60% of the time.
P.M.4.2	(Voice command) Will be able to understand at least 2 unique voice commands, up to 8.
P.M.4.3	(Voice command) Will be able to understand commands of at least 2 words in length, up to 8.
P.D.2.1	Speaker will alert the user to state changes with an 80% success rate.
N.M.9	Audio feedback will be clearly audible in a representative work environment.

The only requirements from the voice subsystem that were added between the PDR and CDR were P.D.2.1 and N.M.9. These requirements were added after it was realized that the user would require audio feedback to interact with the voice subsystem properly. Therefore, a speaker, along with relevant requirements, was added to the project.

3.4. Actuated Manipulation Subsystem Requirements

Table 4 - Actuated Manipulation Subsystem Requirements (Changes Highlighted in Yellow)

ID	Requirement
P.M.2	(Move to object) Will reach within 6 in of the planned target position 60% of the time, and always within 12 in.
P.M.3.1	(Hold object) Will maintain the target's spatial position with less than 6 in of error margin.
P.M.3.2	(Hold object) Will be able to hold a representative part overhead.
P.M.5	(Release object) Will release object within 5 seconds of when the release command is issued.
P.M.6	(Compact arms) Will bring the full robot arm to within 20" of the point of attachment to the backpack.

The only requirements from the actuated manipulation subsystem that changed between the PDR and CDR were P.M.2 and P.M.3.2. The addition of “and always within 12 in.” was added to P.M.2 to place a worst-case maximum boundary on the acceptable error from the system. We found 12” to be a good approximation of the distance between the user’s shoulder blades, and is the usual distance between the user’s hands as they are holding the part. P.M.3.2 was changed from an abstract strength test (hold 2 lbs. at full extension) to something more relevant to the COBORG’s use case.

3.5. Hardware, Electrical, and Sensor Subsystem Requirements

Table 5 - Hardware, Electrical, and Sensor Subsystem Requirements

ID	Requirement
N.M.1	Will be ergonomic for spinal comfort. Will be comfortable to wear for 30 consecutive minutes.
N.M.2	Will weigh less than 40 pounds.
N.M.3	Will be aesthetically pleasing.

4. Functional Architecture

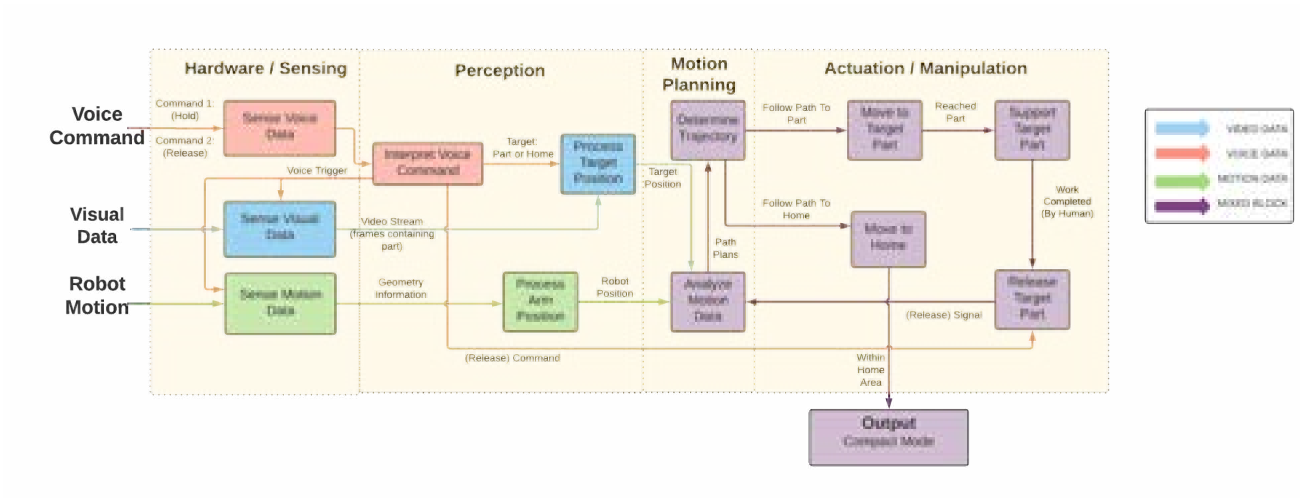


Figure 2 - Functional Architecture of COBORG System

The Functional Architecture shown in Figure 2 outlines the major functions of our system and the data flow between the subsystems. It contains three aspects: the input data, the output, and the four major subsystems. The input data comes from the hardware sensors on the COBORG system. Our system will interpret the data and generate a manipulation output through the robot arm. The details for each subsystem are introduced in the following paragraphs.

The System Inputs, as listed on the left side in Figure 2, are the data coming from the hardware (sensors) on the COBORG. Specifically, our system takes three kinds of input data: voice command, visual data, and robot motion. The voice commands from the user are captured by the microphone on the robot backpack. The visual input is used to localize the target object from the depth camera on the COBORG backpack. The motion data is another input that our system will use to analyze the robot arm motion and plan for trajectory.

The Hardware and Sensing Subsystem is responsible for capturing the inputs. This subsystem will keep track of the sensor data during system operation. Depending on the voice command (i.e. “Go here”, “Come back”), it will trigger the following subsystem processes. The video stream and the point cloud information will be fed into the Perception Subsystem and the motion data will be used in the later Motion Planning Subsystem.

The Perception Subsystem receives sensing data from the Sensing Subsystem, specifically the visual data. After the system interprets the voice command from the Sensing Subsystem, based on the voice command content, the system will detect the desired target (part) position using the visual data and retrieve the robot arm motion data to execute Motion Planning Subsystem.

The Motion Planning Subsystem will use the motion data and the target position information to determine the trajectory for the robot arm. The data will first come into the “Analyze Motion Data” block to generate possible path plans. The determined trajectory from multiple path plans will be forwarded to the Actuation and Manipulation Subsystem.

The Arm Actuation and Manipulation Subsystem receives the trajectory as the input, and by controlling the robot arm, COBORG will follow the trajectory, move to the desired position, and stabilize the object overhead. Once the voice command trigger is received again (i.e. “Come Back”), the robot arm shall release the object and move back to the compact position, which is the system's final output.

5. Cyberphysical Architecture

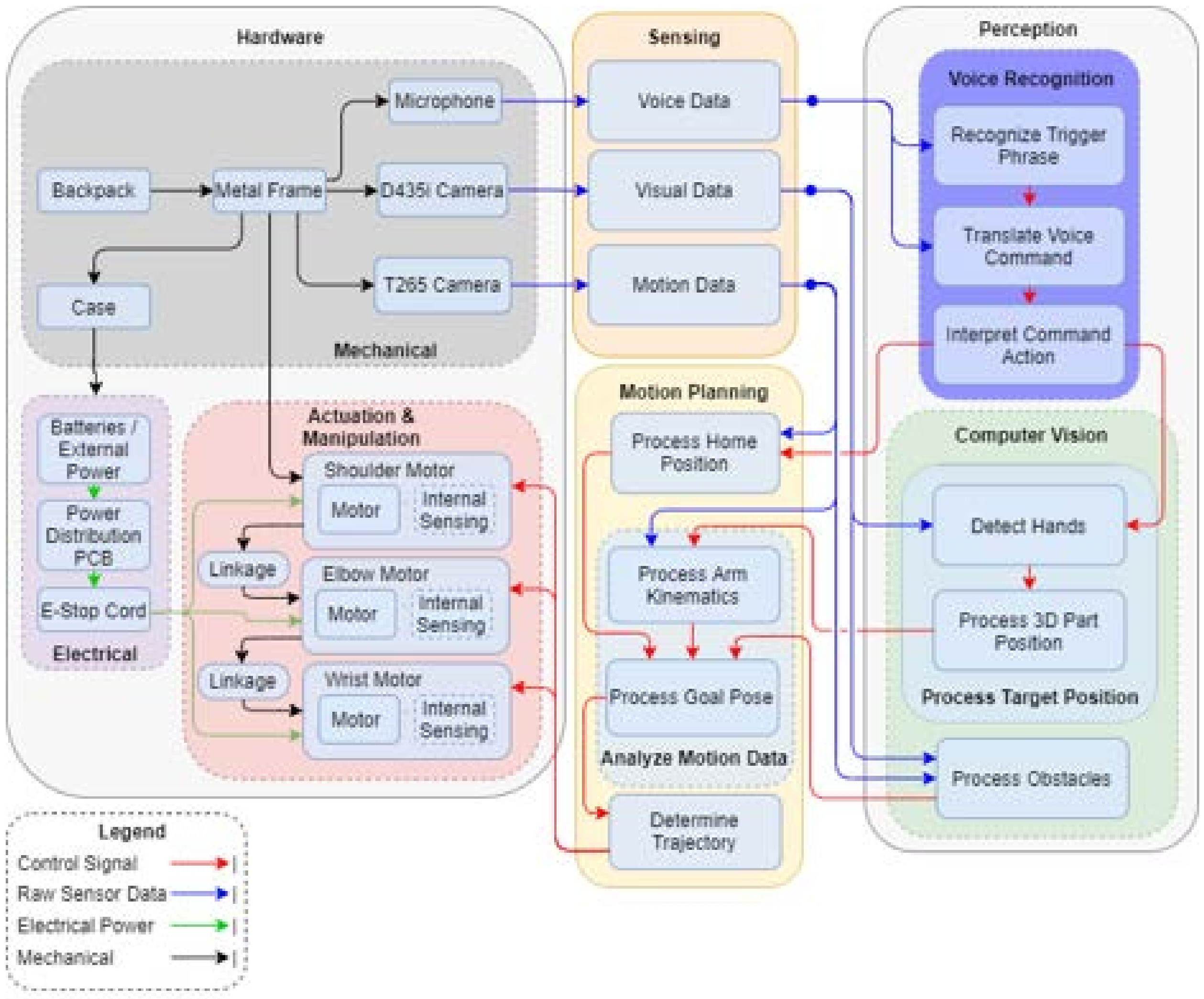


Figure 3 - Cyberphysical Architecture

The Cyberphysical Architecture shown in Figure 3 illustrates the interaction between the hardware actuation and the perception layer of the COBORG system. This elaborates on the general functionality of the COBORG described in the Functional Architecture in Figure 2 to detail how this functionality is communicated between the subsystems and physical components.

The perception layer reads information about the system state and other environmental data through the sensing block in the form of voice, vision, and motion data read from the microphone, D435i depth camera, and the T265 tracking camera respectively. The voice data is used in the voice recognition system to identify the keyword “COBORG” and interprets the phrase following the keyword to identify command actions. If the recognized command is to move to the target, then the vision system is triggered to detect the hands, process the 3D position of the part based on the hand positions, then feed that target position to the motion planning system. If the recognized command is to move the target to the home position, then the motion planning system is directly triggered to compact the arm. The motion planning system is also fed obstacle information from the vision system for obstacle avoidance in actuation.

The hardware components are controlled by the perception layer through the motion planning system. The COBORG arm has three HEBI motors in the form of a shoulder, elbow, and wrist joint stemming from outside the user's right arm. This, along with the rest of the hardware frame and components, is represented in the COBORG's Unified Robot Description Format (URDF). Using this URDF, these motors are actuated using a trajectory planned by the motion planning system. This trajectory is generated by planning from the arm's current position to the desired goal position, whether that be the home position or some target pose. This planned path will also avoid obstacles detected by the vision system.

The remainder of the hardware section describes the linkages between the motors, the physical connectivity of the sensors used, and the electrical system. The electrical system is described later as the COBORG custom power distribution PCB.

6. Current System Status

6.1. Spring-Semester Targeted System Requirements

The focus during the spring semester was on completing the development of the three basic subsystems and fully validating their requirements. The targeted requirements for the vision subsystem can be found in Table 6 below. The targeted requirements for the voice subsystem can be found in Table 7 below. The targeted requirements for the actuated manipulation subsystem can be found in Table 8 below. The targeted requirements for the hardware subsystem can be found in Table 9 below.

Table 6 - Targeted Vision Subsystem Requirements

ID	Requirement
P.M.1.1	(Detect indicated parts) Will have 60% accuracy of detecting indicated location within 6" in 3D space, and always within 12".
P.M.1.2	(Calculate object) Will detect the intended object within 5 seconds of when the move command is issued.
P.M.1.3	(Pose Detection) Shall detect the surface normal of the part with an error no greater than 45°.
P.D.1.2	(Texture Invariant) Must be invariant to part texture, specifically matte finish and gloss finish.

Table 7 - Targeted Voice Subsystem Requirements

ID	Requirement
P.M.4.1	(Voice command) Will be able to understand the voice command 60% of the time.
P.M.4.2	(Voice command) Will be able to understand at least 2 unique voice commands, up to 8.
P.M.4.3	(Voice command) Will be able to understand commands of at least 2 words in length, up to 8.
P.D.2.1	Speaker will alert the user to state changes with an 80% success rate.
N.M.9	Audio feedback will be clearly audible in a representative work environment.

Table 8 - Targeted Actuated Manipulation Subsystem Requirements

ID	Requirement
P.M.2	(Move to object) Will reach within 6 in of the planned target position 60% of the time, and always within 12 in.
P.M.3.2	(Hold object) Will be able to hold a representative part overhead.
P.M.6	(Compact arms) Will bring the full robot arm to within 20" of the point of attachment to the backpack.

Table 9 - Targeted Hardware Subsystem Requirements

ID	Requirement
N.M.1	Will be ergonomic for spinal comfort. Will be comfortable to wear for 30 consecutive minutes.
N.M.2	Will weigh less than 40 lbs.

6.2. Overall System Depiction

COBORG is a collaborative robot arm that can help people hold objects overhead. The inputs to the system come from the user's voice, which triggers the system to start moving to the target and stabilize it in place. To achieve this goal, our system consists of several subsystems, including hardware framework, electrical framework, sensing, perception, motion planning, and actuated manipulation. The current status of the overall system is depicted in Figure 4 below. In the Spring semester, we completed the subsystem development on the current hardware framework and validated their performance separately. By the end of the SVD encore, we also did the initial subsystem integration on the current hardware framework. The PCB board shown in Figure 4 is also used in the current system as part of the electrical framework.

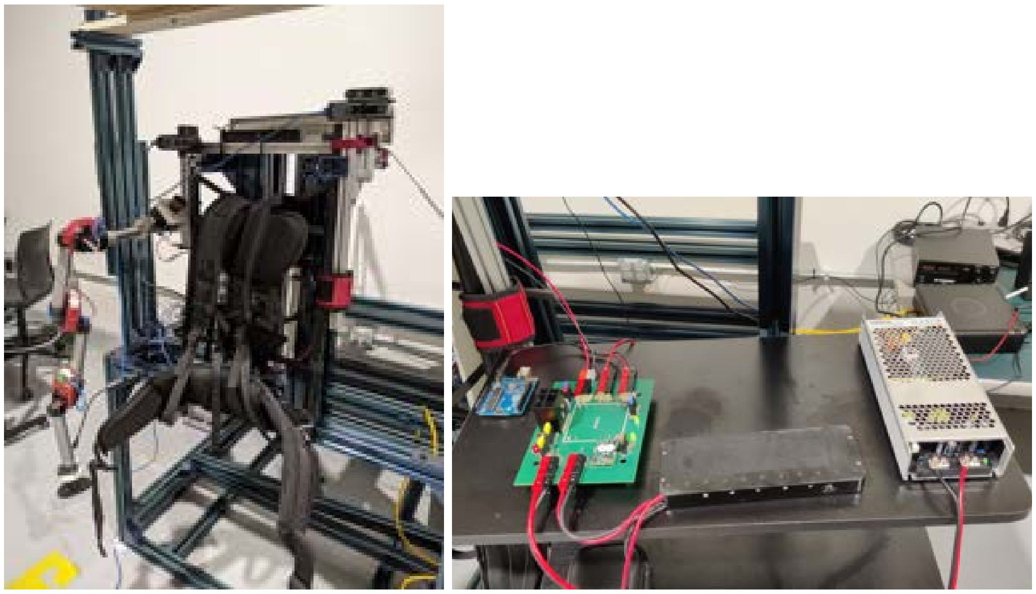


Figure 4 - Overall Depiction of COBORG (left) and the PCB (right)

6.3. Subsystem Descriptions/Depictions

6.3.1. Hardware Framework

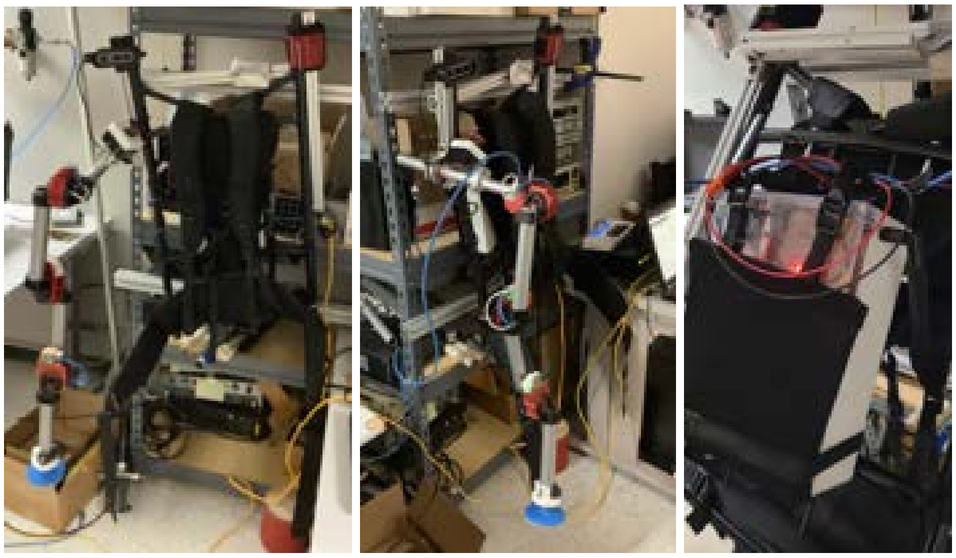
The Hardware Framework of the COBORG is broken down into three subcomponents:

- Frame
- Case
- Manipulator Arm

The frame consists of the structural foundation of the robot that is mounted to the user. This includes the backpack that the user is wearing as well as the structural metal frame attached to the backpack straps. The manipulator arm and the case are structurally secured to the frame. See Figure 5 (A) for pictures of the as-built frame.

The manipulator arm is installed to the frame and serves as the mobile unit of the robot. This includes the aluminum linkages, motors, and end effector. See Figure 5 (B) for pictures of the as-built manipulator arm.

The case is the container box that houses all head-end equipment to the robot. A majority of the electrical framework components are housed in the case. Access holes are installed on the exterior of the case such that the wiring can safely travel into and out of the case. See Figure 5 (C) for pictures of the as-built case.



(A) Front View **(B) Side View** **(C) Rear View**

Figure 5 - As-Built COBORG Framework

6.3.2. Electrical Framework

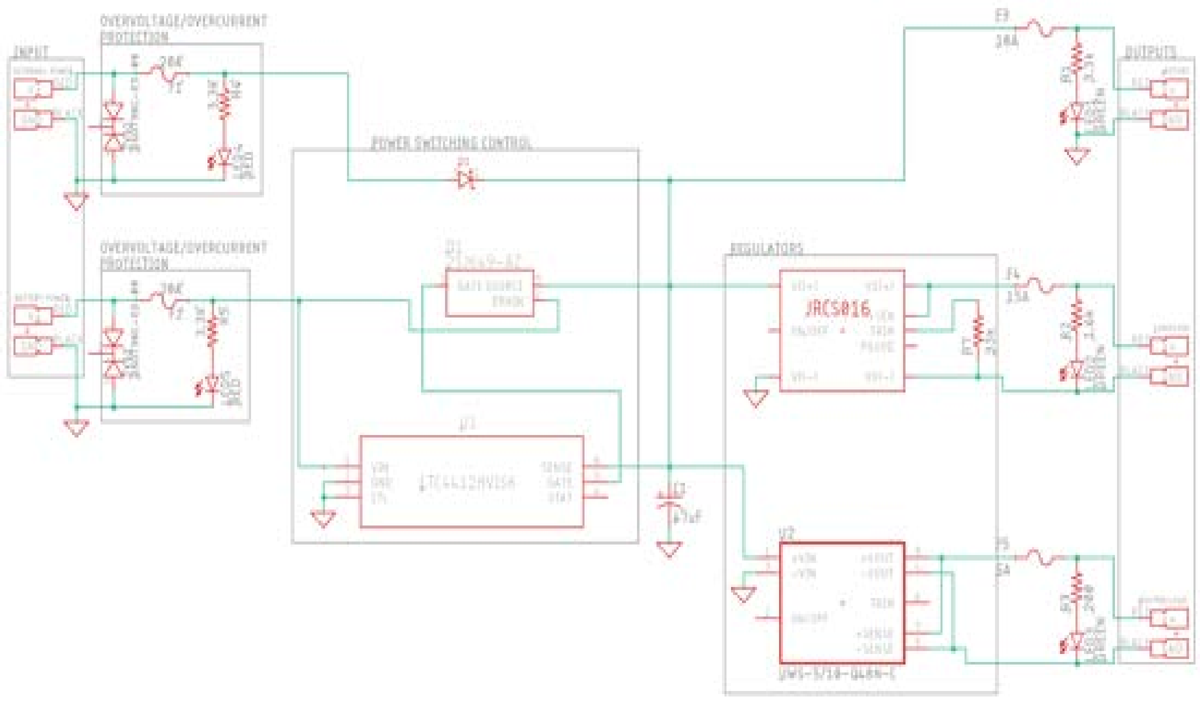


Figure 6 - COBORG Power Distribution PCB Schematic

The COBORG system can be powered by either an external 36VDC power source (i.e. wall outlet) or by the onboard 36VDC battery packs. Power is then distributed to the motors, computer, and router/5V logic via the COBORG Power Distribution Printed Circuit Board (PD PCB), shown in Figure 7 and schematic shown in Figure 6. This PD PCB prioritizes power from the external power source over the battery. The internal batteries are always plugged into the system, but when the external power is connected, the logic on the PCB switches the power sourcing to draw the full 3-10A (~100-400W) from the external supply and only draw ~10mA (~0.4W) of current from the battery. This allows the battery to maintain charge and also keeps the batteries in a state of minimal discharge to allow for safer charging; charging which will also source from the external power. This 36VDC input directly powers the COBORG arm motors at the recommended 36VDC. The PCB has two switching voltage regulators to convert the input 36VDC to 19VDC and 5VDC for the computer and router/5V logic respectively. The 5VDC power is achieved by feeding the 36VDC into a "18VDC-75DC to 5VDC" converter (UWS-5/10-Q48P-C). The 19VDC power is achieved by feeding the 36VDC into a variable power supply that accepts 18VDC-85VDC and outputs a voltage related to the trim resistor connected to the PCB. Based on the equation given on the datasheet, $\mathfrak{C}_{\downarrow\uparrow\aleph} \epsilon \frac{\tilde{\imath}\ \psi\psi_{\upsilon}\ \omega\psi\psi\aleph_{\downarrow}}{\aleph_{\downarrow}{}^{\upsilon}\ \acute{\eta}}$, with a desired $\aleph_{\downarrow} \leq$ 19VDC, a 37kΩ resistor was selected, which due to tolerances on both the resistor and the regulator put the output voltage to ~18.7VDC.

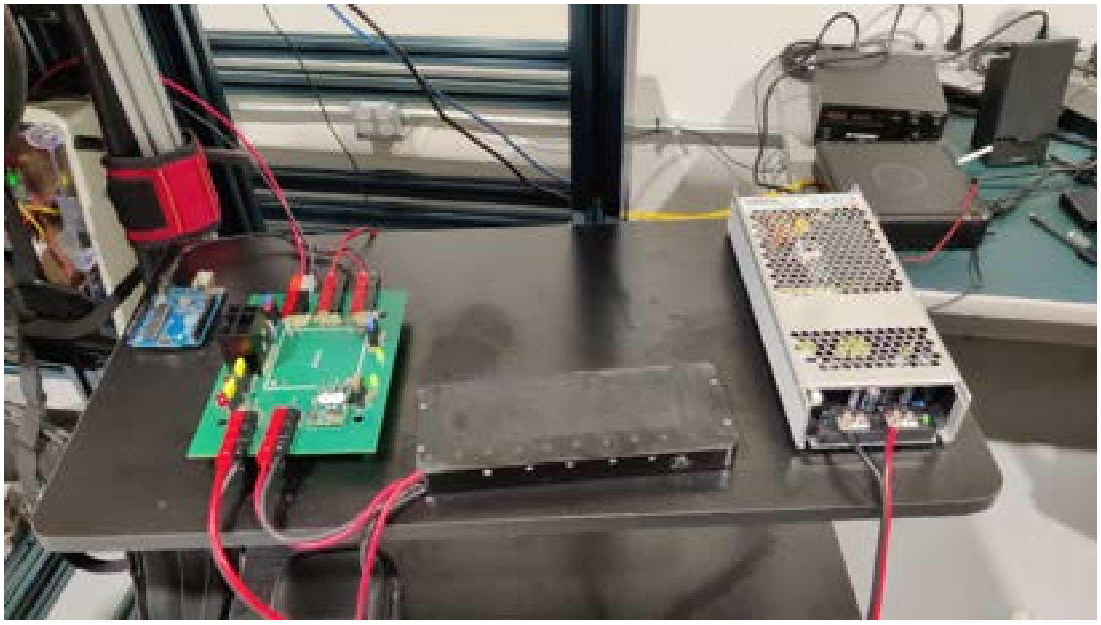


Figure 7 - COBORG Power Distribution PCB w/ Both Power Sources Plugged in

6.3.3. Sensors

The following sensors are used on the COBORG:

- Intel Realsense D435i depth camera (1)
- Intel Realsense T265 tracking camera (1)
- IMU sensor (internal to each HEBI motor) (3)
- Microphone (1)

The Realsense cameras and microphone are currently mounted onto the frame and close to the user's shoulder(s). The D435i depth camera is mounted on the user's right shoulder, which has a better view of the user's hands and robot arm. The T265 tracking camera is mounted on the user's left shoulder. By defining the translation between these two camera frames, the robot arm shall follow the hand movement shown in the depth camera. IMU sensors exist internally in each of the three HEBI motor models being used on the COBORG. Figure 8 demonstrates the position of these sensors.

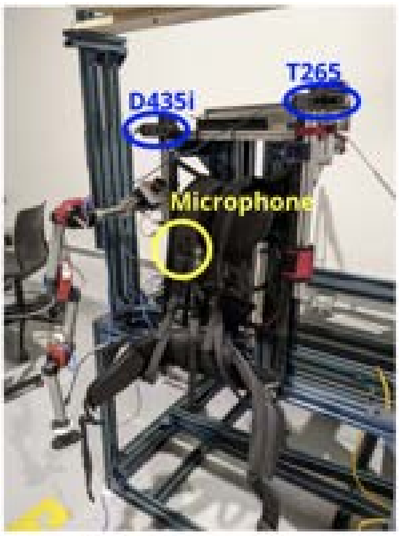


Figure 8 - Sensor Positions

6.3.4. Perception: Vision

The vision subsystem aims to determine the 3D target part position and surface normal by utilizing a hand detection algorithm combined with point cloud information. Intuitively, the hand position could be inferred as the goal position of the target since users always want to put their hands on the target to hold it overhead. By localizing the hand position on the part, the part goal position can be easily retrieved. For the spring semester, the vision system will implement 2D hand detection, 3D bounding box extraction, and post-processing which includes the goal position calculation and surface normal estimation.

The vision subsystem uses the 2D hand detection algorithm to localize the hand in the camera frame. The current choice for the hand detection algorithm is the open-source algorithm YOLO v3 (You Only Look Once) [3]. YOLO is a popular neural network-based method that is commonly used in detection tasks. The network already has a custom implementation on hands detection, which is trained on two large hand datasets and shows great precision results on the validation set. The pretrained model is also interfaced with ROS by cooperating with another open-source ROS wrapper built for YOLO. An example of hand detection results combining with ROS is shown in Figure 9.

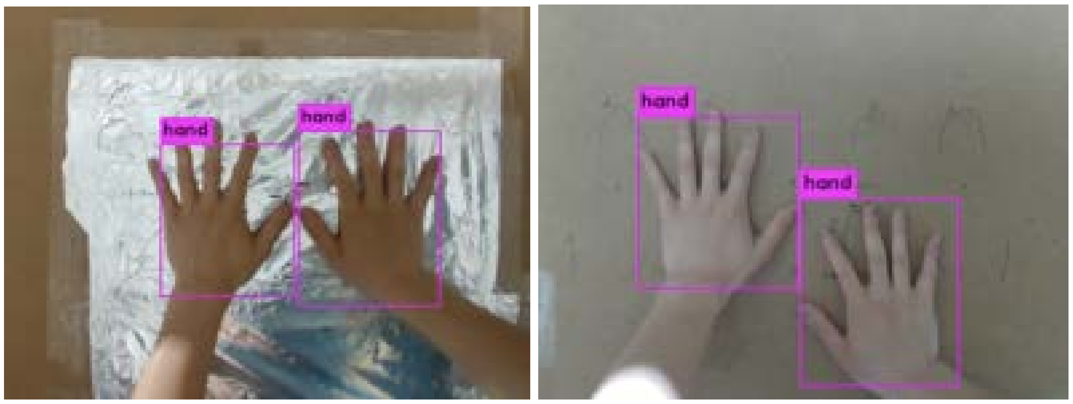
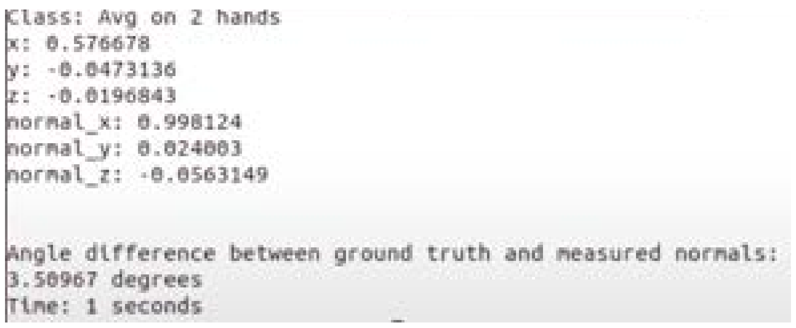


Figure 9 - Sample Outputs for 2D Hand Detection Results

Once the 2D bounding boxes are retrieved, the system will start looking for the point cloud information from the depth camera (D435i) to draw the 3D bounding boxes. In detail, within each 2D hand bounding box, the system looks for the minimum depth and maximum depth from the point cloud and uses that range of the depth to draw the 3D bounding boxes around the hands. The size of the 3D bounding boxes remains the same as the 2D bounding boxes except extra depth information is provided. This estimation process is applied to every 2D hand bounding box outputted from the previous step.

To localize the target and obtain the surface normal, our system further does post-processing on those 3D bounding boxes. In detail, the system first takes the average of the 3D bounding boxes' center positions as the goal position. If there is a single hand in the frame, the center position of the hand will be set as the goal position. If multiple hands show in the frame, the average of the hand positions will be indicated as the goal position. After obtaining the goal position of the target, it further searches the nearby points around the goal position to estimate the surface normal for the target board. The 3D goal position and surface normal will be used in the later motion planning system and stabilization task.

To ensure the communication between different subsystems, all of the processes discussed above are currently implemented within ROS. By using ROS publishers and subscribers, the data could be easily transferred between different processes. A sample output from the ROS topic is shown in Figure 10, which summarizes the 3D target position(i.e. x, y, z) and also the surface normal vector (i.e. normal_x, normal_y, normal_z). It also outputs the overall execution time for the vision system, which will be compared with the requirement in the later validation testing.



```
Class: Avg on 2 hands
x: 0.576678
y: -0.0473136
z: -0.0196843
normal_x: 0.998124
normal_y: 0.024003
normal_z: -0.0563149

Angle difference between ground truth and measured normals:
3.50967 degrees
Time: 1 seconds
```

Figure 10 - Final Output from ROS Topic

6.3.5. Perception: Voice

The voice recognition subsystem serves as the interface between the user and the COBORG for the user's desired functionality. The voice recognition system uses the open-source PocketSphinx platform for its optimal recognition for fully offline/untethered functionality. For implementation on the COBORG system, the dictionary was optimized to recognize commands and the word "COBORG" because COBORG is not in the standard English dictionary. To improve recognition, the given model was trained on the new dictionary for roughly a day to optimize the model for the optimized dictionary. The input for the voice recognition system is the audio stream from the microphone mounted on the COBORG backpack strap. The audio output is the speaker mounted on the COBORG. The input audio is processed using pyaudio and the output is served to the speaker as mp3 files using pydub built on pyaudio. The voice recognition system is built on the voice_recog ROS node, which once a valid command from the user is recognized, publishes the command code to the /voice_commands ROS topic. From here, the main_state_machine subscribes to this topic to process the voice command and dictate to the other subsystems the commanded function.

The user triggers the command recognition by saying the keyword "COBORG". Once "COBORG" is recognized, the COBORG plays a short jingle to alert the user that it is now listening for a command. From here, the user can command the robot to move to the part target established by the aforementioned vision perception algorithm by saying "Go Here". Once the user has secured the part, the command "Come Back" will bring the COBORG off the target part and back to the compact home position. After the keyword is said and either of these commands is recognized, the COBORG plays a short positive success jingle to confirm that the command was successfully interpreted. Additionally, the user can trigger the command prompt with "COBORG" and say the command "Stop" to trigger the soft emergency stop function. Alternatively, the user can say "Stop Stop Stop" without preceding with the "COBORG" trigger for a more easily accessible and natural command to emergency stop the COBORG's current function. Once either of these e-stop commands is recognized, the COBORG plays a jarring alert sound. If the "COBORG" keyword is recognized but the voice system does not recognize any valid commands in the following phrase, it will play a negative failure jingle to alert the user that no command was detected and therefore the COBORG will not act.

6.3.6. Actuated Manipulation

The actuated manipulation subsystem stands at the end of the pipeline, providing the physical actuation and force output needed to aid the user in their intended task. This subsystem is merely a husk, carrying out a task set forth by the previous subsystems. The intention is derived from the output of the voice subsystem. Directed precision and aim are derived from the output of the vision subsystem. Similar to how a human's muscles need a brain to be commanded, the COBORG's manipulator arm needs visual and auditory perception to be commanded intelligently. In other words, given the high-level goal command interpreted by the voice subsystem and main state machine as well as the end goal pose from the vision subsystem, the actuated manipulation system will actuate its various joints to move the end effector from its current standby state to the end-goal state within a reasonable amount of time and through a viable pathway that is minimally intrusive to the user and its environment.

As of the end of the spring semester, the current state of the actuated manipulation system was centered around the aforementioned primary use case: to provide force and support to a panel between a user's two hands either forward or overhead. Figure 11 below displays the physical build of the robot at its predefined standby state (i.e. home or compact position). The manipulator arm is a serial linkage system with three revolute joints with a flat pad installed on a spherical joint at the end of the arm, serving as the end effector. Each revolute joint is installed at a specific angle relative to the principal axes of the environment and between each other. The entirety of the arm is situated on the right side of the user, somewhere between the top of their shoulder and their elbow joint, depending on the height and build of the user. The linkages connected to each revolute joint are thin-walled steel tubes. The revolute joints are HEBI X-Series actuator motors and the mounting brackets between the joints and linkages were also made by HEBI. Data and power are serially connected between motors where the linkages serve as the conduit between motors. From the proximal to the distal motor, the names of the motors are as follows: base motor, elbow motor, and wrist motor. Figure 11 below displays the build of the manipulator arm as well as the rest of the hardware frame.

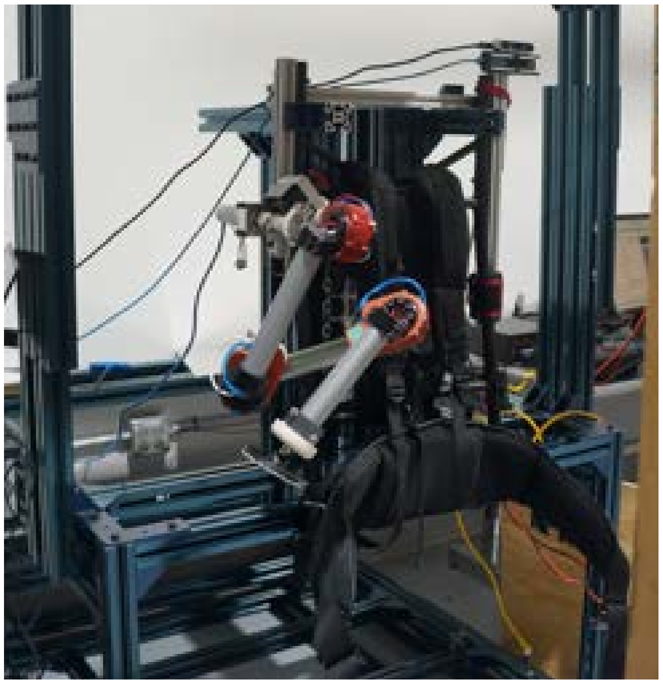


Figure 11 - Manufactured State of COBORG, Mounted onto Testing Frame

At the standby state of the actuated manipulation subsystem, the manipulator arm is in its compact/home position as shown in Figure 11 above. As of this spring semester, the primary high-level action to perform is for the end effector to push onto a part that is situated within reach of the manipulator arm and in front of the user. The goal state is located somewhere on the surface of the part and between the two hands of the user (assuming that the user is using two hands to support the part while giving the push command and both the hands are in view of the depth camera used by the vision subsystem). As of this spring semester, the goal state that the manipulator arm is processing is the goal position output from the vision subsystem. To get the goal state from the vision subsystem, the actuated manipulation subsystem first subscribes to the */goal* rostopic, and waits a short amount of time before extracting one message. That extracted goal state is then transformed from the *camera_link* frame to the *world* frame of the robot URDF, a frame at which the actuated manipulation subsystem uses.

After picking up the goal state from the vision subsystem, the robot arm will first actuate to its ready position which consists primarily of rotation of the base and elbow motors such that the end effector pad is orientated outward and in the direction the user is facing. This intermediate position was implemented to prevent the manipulator arm from reaching an undesirable orientation while actuating to the goal state. The next intermediate position is at a position that is an X-offset away from the goal position. This position was employed to encourage final X-axis directional push forces that are perpendicular to the surface of the board in front of the robot, though this is considered an optional intermediate position for the manipulator arm. Should the actuated manipulation subsystem be unable to solve for this intermediate position within a set amount of time, the next state is the goal state. Once the end effector reaches the arm, the actuated manipulation subsystem waits for the next command which is the "Go Home" command.

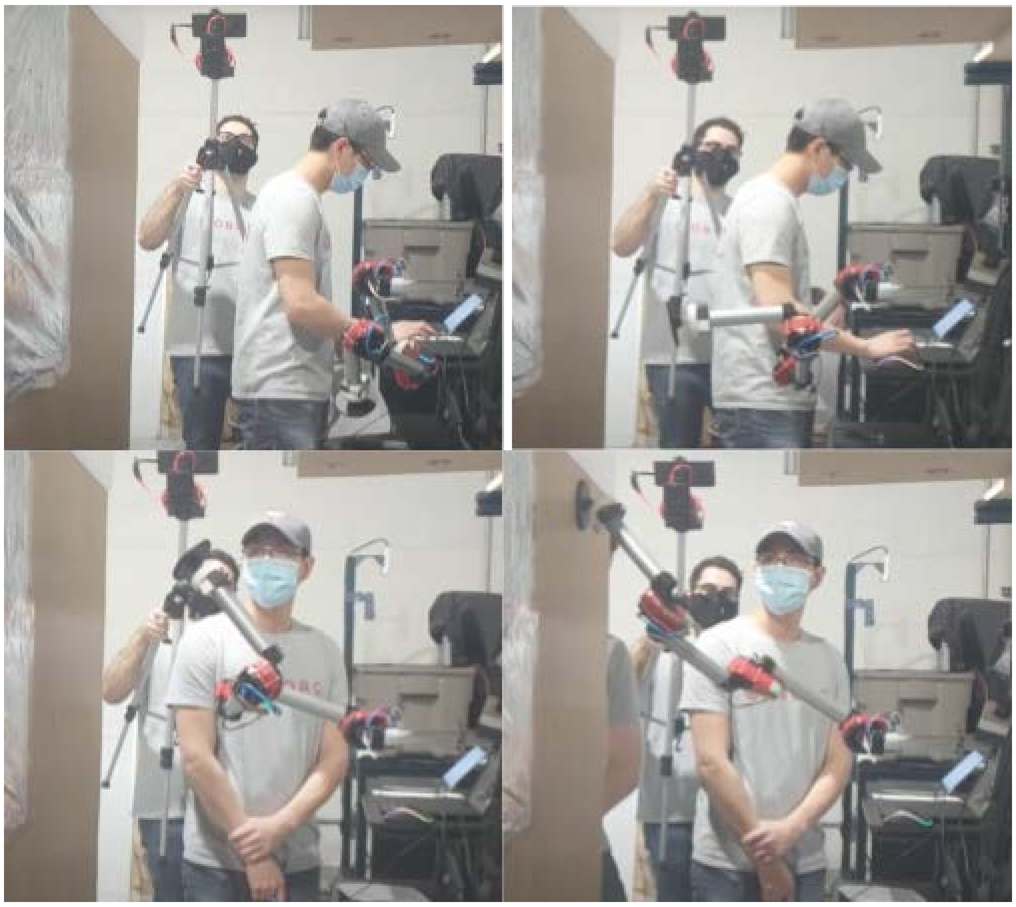


Figure 12 - Actuated Manipulation Push Out Process Screenshots: compact/home position (top left), ready pose (top right), X-offset to push out (bottom left), final goal position (bottom right)

To pull back from the goal state and actuate the manipulator arm to its home position, the robot arm first goes to the intermediate position that is a set X-offset away from the goal state. This is an optional position that, if the subsystem is unable to solve for a viable path within a set amount of time, the next goal state is the ready position of the robot arm. After reaching the ready state, the robot arm goes to its compact/home position and waits on standby for the next command to be issued. In essence, the robot arm will pass through the same intermediate points that it went through to get to the goal position.

6.4. Modeling, Analysis, and Testing

6.4.1. Vision Subsystem Validation

Before SVD, the vision subsystem performance is evaluated in below experiment setup. To easily measure the ground truth and the prediction value, we fixed the COBORG on a frame. The depth camera is on the left shoulder pointing straight to the board. To validate the consistency of the performance, 10 predefined positions on two different part textures (i.e. shiny, matte) are prepared on the board to test. Figure 13 shows the experiment setting for vision system validation testing. These 10 predefined positions cover different moving directions of hands and different distances between two hands. The testing is then repeated four times to validate the system consistency and essentially collect 36 valid measurements. Table 10 below shows the summary results from the testing.

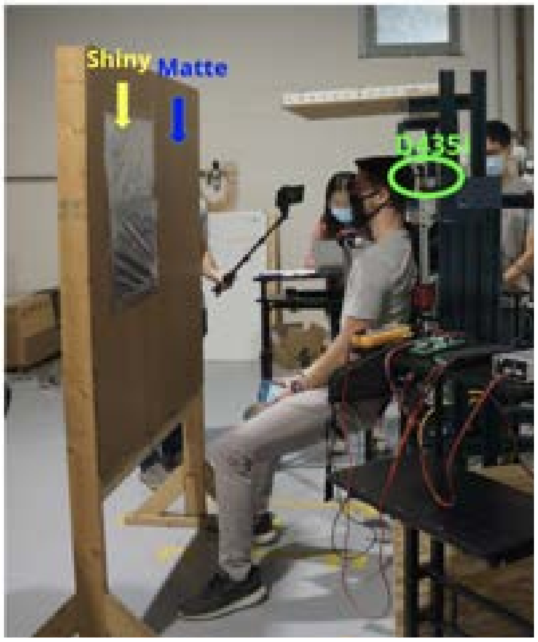


Figure 13 - Vision System Validation Setup

To summarize the results, the system performs well in all the testing cases and meets our requirements for the vision system. The averaged euclidean distance error is about 4 cm (1.57 in.) compared to the ground truth value. As our requirement mentioned, the euclidean distance error shall be within 6 inches (see more details in section 3). Our validation performance meets the requirement and remains a large buffer for our system. Also, the surface normal angle error is about 4.4 degrees compared to the true angle, which is also much less than 45 degrees mentioned in the requirement.

Table 10 - Vision System Validation Results Summary

Distance Error		Surface Normal Angle Error	
Sample Size	36	Sample Size	36
Sample Mean (m)	0.042m	Sample Mean (deg)	4.4deg
0.305m (12")	<1e-30	45 deg	<1e-30
0.1524m (6")	<1e-30	10 deg	0.0086
0.059m	0.0093	9 deg	0.0308
0.056m	0.034		

6.4.2. Voice Subsystem Validation

The results in Table 11 are from an isolated validation test with the microphone mounted on the COBORG using Gerald's voice for the test. 150 voice commands were tested including the command to go to the user's goal target, the command to compact to the arm, and the emergency stop command. A successful output was defined as an accurate recognition of the intended command in at most 2 tries. This criterion was defined based on irritation, if a command takes more than 2 tries to recognize then it irritates the user and defeats the purpose of convenience of voice control. After each trial of three commands, a randomly generated sentence of at least 20 words was read to test if the voice system would detect the keyword "COBORG" trigger in random speech. The goal of this test is to validate the 60% recognition accuracy dictated by requirement P.M.4.1.

Table 11 details the results from the validation test. Based on the two chances for successful command recognition, 147 out of 150 commands were successful with 1 false-positive keyword trigger from random sentences. Table 11 also details the results of a binomial test which confirms that the probability of getting 60% accuracy is ~100% (1-1.303e-27). The results elaborate to express that the chance of having at least a 94% command accuracy is 97% (1-0.0302). Additionally, considering a single missed command as a failure using the same test trials, the system successfully recognized 131 out of 150 commands with 1 false-positive trigger. Even with these stricter qualifications, the probability of getting 60% accuracy based on the binomial test is ~100% (1-3.99e-13), and getting 94% accuracy is around 95% (1-0.0427). This clearly expresses the validation of requirement P.M.4.1.

The other voice system requirements are validated implicitly. P.M.4.2 requiring at least 2 unique voice commands is satisfied because there are currently 5 unique voice commands for 4 unique actions (2 commands for emergency stop, one inside the keyword trigger structure and one independent). P.M.4.3. requiring commands of at least 2 words in length is satisfied by all recognized commands currently being 2-4 words in length. P.D.2.1 requiring the system speaker to alert the user to state changes with an 80% success rate is satisfied because the code structure ties together the audio feedback code and the /voice_command ROS topic publishing code so that it works with 100% accuracy. N.M.9 requiring that the audio feedback be clearly audible is satisfied by the design, placement, and volume of the system speaker.

Table 11 - Voice System Validation Results Summary

Conditions of 2 Attempts for Successful Recognition		**Conditions of 1 Attempt for Successful Recognition**	
Number of Commands	150	Number of Commands	150
Number of Successes	147	Number of Successes	131
Number of False Positives	1	Number of False Positives	1
60%?	<1.303e-27	60%?	<3.99e-13
92%?	0.004	92%?	0.0089
94%?	0.030	94%?	0.043

6.4.3. Actuated Manipulation Validation

Before SVD, the actuated manipulation subsystem was tested on 3 pre-measured positions on a vertical board directly in front of the robot arm distanced about 90 centimeters away from the robot frame. Each of the 3 pre-measured positions was tested a total of 6 times. The robot arm was commanded to go to these goal positions from the compact/home position. The robot arm was allowed to pass through two intermediate positions between the compact/home positions and the goal position. When the robot arm reached the goal position, a tape measure was used to measure the error distance between the ground truth pre-measured point and the landed position of the end effector on the board. The end effector has to be firmly pressed onto the board. The robot arm was when commanded to go back to the compact/home positions. The robot arm passed through the same two intermediate positions before reaching the compact/home position. Once the robot arm reached the position, a measurement was taken with a tape measure between the base motor and the end effector tip of the robot arm.

Figure 14 below shows the 3 pre-measured goal positions that were tested on the COBORG during SVD. Position 02 was set orthogonally to the right from Position 01 by 15 centimeters. Position 03 was set orthogonally downward from Position 02 by 15 centimeters.

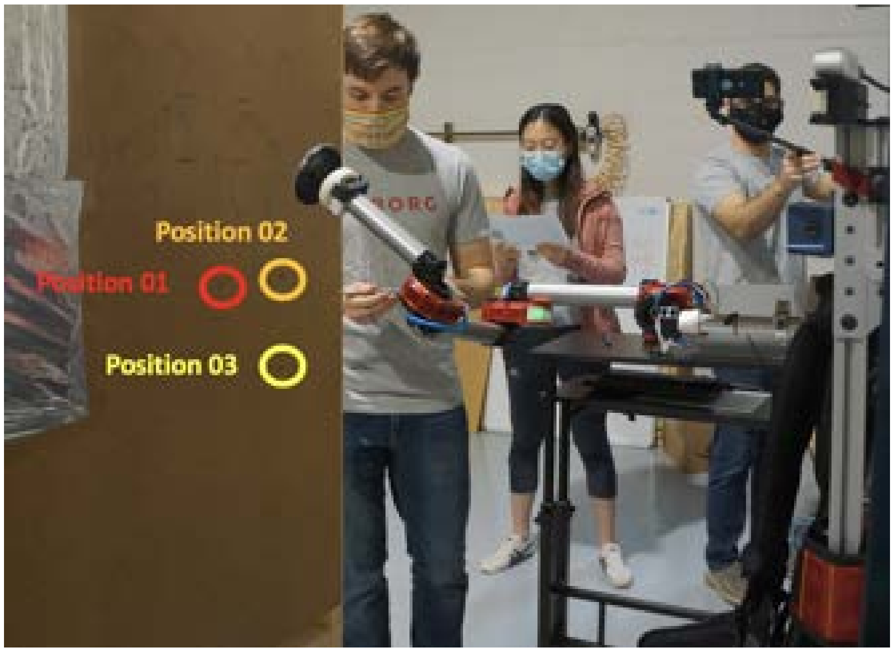


Figure 14 - Actuated Manipulation System Validation Setup

To summarize the results shown in Table 12 below, the actuated manipulation subsystem performed well to a relatively high precision when working around the set distance and workspace area set by the 3 pre-measured goal points. Distance error measurements and statistics determined in pre-SVD testing confirmed the precision of the actuated manipulation system meets the precision requirements set forth on this project. Compact distance error measurements and statistics determined in pre-SVD testing confirmed the consistency and compact size requirements set forth on this project.

Table 12 -Actuated Manipulation System Validation Results Summary

Distance Error		Compact Distance	
Sample Size	18	Sample Size	18
Sample Mean (in)	0.354	Sample Mean (in)	13.56
12”?	<1e-30	20”?	<1e-30
6”?	<1e-30	14”?	0.00299
0.8”?	0.0099	13.9”?	0.0166
0.7”?	0.0454		

6.5. Performance Evaluation against Spring Validation Demonstration (SVD)

During SVD, four system tests were carried out to evaluate their performance. Starting with the vision subsystem test, 10 predefined positions, same as the settings mentioned in the previous section (6.4), were used to evaluate the vision system performance. These predefined positions covered different board textures (i.e. shiny and matte) and different hand locations. Instead of iterating the whole test 4 times, we only went through the test once in SVD. By taking the average of the errors for all of the cases, we compared the final results with the vision subsystem's targeted requirements. The summary of the SVD results is shown in Table 13. The vision system performed well in all of the testing cases and met our requirements for the vision system. The averaged euclidean distance error was about 3.74 cm compared to the ground truth value. As our requirement mentioned, the euclidean distance error shall be within 6 inches (see more details in section 3). Our SVD performance met the requirement and even left a large buffer for our system. Also, the surface normal angle error was about 4.17 degrees compared to the true angle, which is also much less than 45 degrees mentioned in the requirement. In SVD, we also time the system execution to evaluate the system performance on runtime. For the vision system, the average execution time is 1.3 seconds over 10 test cases, whereas the success criterion was to finish execution within 5 seconds.

The voice subsystem was tested by running through 2 trials of the full validation test described in 6.4. First, the user put on the backpack strap that held the microphone and read off the two commands for COBORG action, the command to move to the target and subsequent the command to compact back to the home position. Then, a random phrase was read from a random sentence generator to test for false-positive command triggers. Next, the two emergency stop commands were tested, the one within the keyword trigger structure and then the independent command. To end the trial, another randomly generated sentence was read by the user to test for false-positive triggers. This process was repeated to showcase 2 trials for SVD. Based on the qualifications that a command can be repeated once and still pass as a success if correctly interpreted the second time, the SVD had 100% accuracy in voice recognition. Unfortunately, many of the commands took two attempts so the demo was not as cleanly successful as it seemed. To remedy this, the voice subsystem was overhauled to optimize the command dictionary for improved recognition and the commands were changed to be simpler and more intuitive. During SVD Encore, one aforementioned trial was performed by Gerald for proof of concept, and one trial was performed by Dr. Dolan for proof of robustness to various users. In both trials for both users, every command was recognized on the first attempt for a cleaner 100% voice recognition accuracy.

To test the actuated manipulation subsystem during SVD, the robot arm was commanded to move to each of the 3 aforementioned pre-measured actuated manipulation goal points at the pre-measured set board distance of 90cm. The number of iterations performed on the same point was cut down to only one iteration during SVD. The results of SVD testing resulted in the actuated manipulation subsystem meeting its accuracy and precision requirements, as shown in Table 13 below. The average surface distance of the end effector between the ground truth of a commanded goal point and the actual landing position of the end-effector was averaged out to be about 1.46cm between all 3 goal points, which was well below the about 15 cm max error requirement set. The robot arm was compacted back to its home position after every run to a goal point, and the average base-to-end distance between the base motor and the end effector at the compact/home position was about 34.5 cm, which was well below the max base-to-end distance of 50.8 cm requirement set

Two non-functional requirements were tested for the hardware subsystem during SVD: ergonomics and weight requirements. Ergonomics was shown during SVD encore, when a user from the team wore the COBORG to perform a functional test of the fully integrated COBORG with voice, vision, and actuated manipulation. The user demonstrated very few readjustments during use and the robot did not pinch, scratch, or harm the user during the full integration testing. Further testing of this non-functional requirement will be expanded upon in the Fall semester to validate the ergonomics of the COBORG. Weighing the COBORG hardware frame consisted of weighing a user on a bathroom scale then comparing the difference with the user wearing the COBORG frame and weighing themselves. A digital bathroom scale was used for this weight measurement. One testing iteration was performed for the weight and the weight came out to be about 17.5 lbs (shown in Table 13 below), which was well below the weight requirement for the project at a max weight of 40 lbs.

Table 13 - SVD Results Summary

Subsystem	Metrics	Success criteria	Performance on SVD
Perception: Vision	Euclidean distance error, Surface normal angle error, Execution time	Distance error < 6 in (~15cm) Angle error < 45 deg Execution time < 5s	Avg. distance error =3.74cm Avg. angle error = 4.17deg Avg. execution time = 1.3s (Avg. on 10 testings)
Perception: Vision	Binary Recognition Accuracy/Command	60% Recognition Accuracy	100% recognition accuracy
Actuated Manipulation	Euclidean distance error Euclidean distance error, base-to-end	Distance error < 6 in (~15cm) Compact distance < 20 in (50.8 cm)	Avg. distance error = 1.46cm Avg. compact distance = 34.5 cm
Non-functional	Digital bathroom scale	Weight < 40 lbs	Weight ~= 17.5 lbs

6.6. SVD Strong/Weak Points

We finished strong in our SVD, and we learned quite a bit in the development of our subsystems. Our vision system performed well in detecting hands; however, it doesn't tell us the hand's pose, which we need for the robot arm to push effectively into the part. We plan on post-processing the depth map next semester to give us the surface normal of the part we are pushing into. Our voice subsystem also performed robustly, and next semester we will work on making it invariant to loud background noises that can be found in factories. Our actuated manipulation system also performed effectively, however it wasn't as fast as we thought it would be, and we want to add another degree of freedom to allow our arm to reach more places. Below is the detailed breakdown of each subsystem with the pros and cons listed in a digestible format.

6.6.1. Vision Subsystem:

- **Strong points:** High detection accuracy, real-time detection (30 FPS)
- **Weak points:** Some false-positives, color-dependent, hand pose variant
- **Future work:** Post-processing the point cloud to extract surface normal, filter out visual noise

6.6.2. Voice Subsystem:

- **Strong points:** High recognition accuracy, flexible command structure, low/no false-positives, no false commands sent to main_state_machine
- **Weak points:** not robust to excessive variable background noise
- **Future work:** Expand command dictionary, simplify existing commands, optimize dictionary for improved recognition, add command timeout function

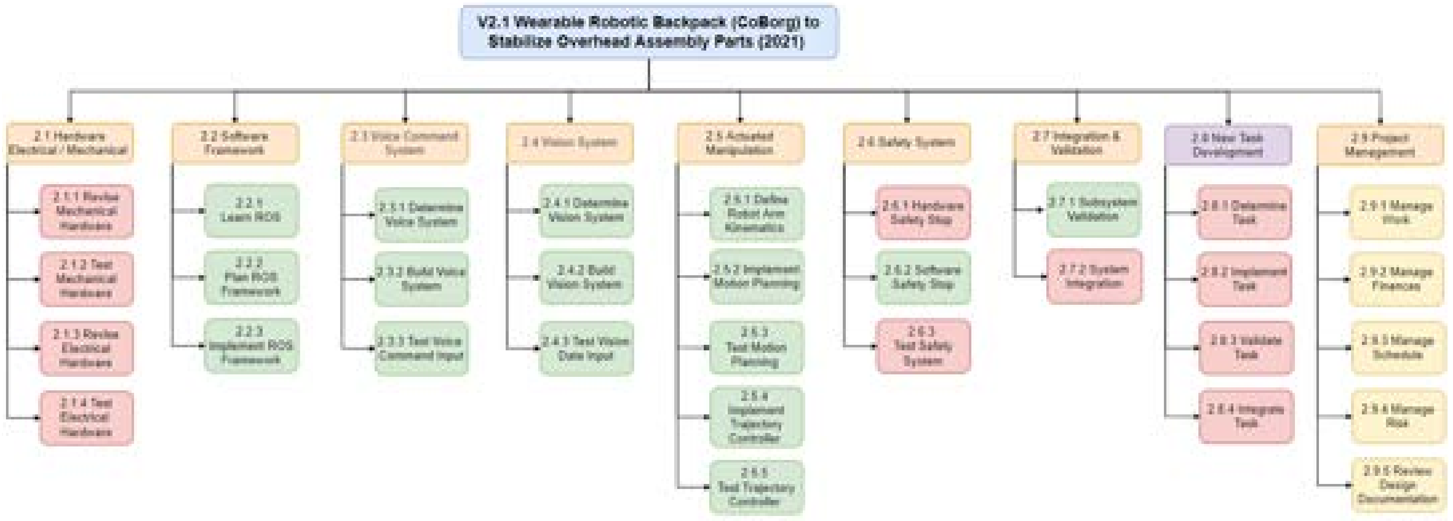
6.6.3. Actuated Manipulation Subsystem:

- **Strong points:** High position accuracy, no hard collision with user
- **Weak points:** Inconsistent performance, slow performance, position-only actuation
- **Future work:** Global-frame stabilization, increase DoF to solve full pose, collision avoidance, perform multiple simple tasks

7. Project Management

7.1. Work Breakdown Structure

Figure 15 below depicts the updated high-level Work Breakdown Structure (WBS) required to execute the COBORG system. From the high-level breakdown, we were able to derive the tasks required to accomplish each of the work packets detailed in the WBS:



V2.1 Wearable Robotic Backpack (CoBorg) to Stabilize Overhead Assembly Parts (2021)
2.1 Hardware Electrical / Mechanical
2.1.1 Revise Mechanical Hardware
2.1.2 Test Mechanical Hardware
2.1.3 Revise Electrical Hardware
2.1.4 Test Electrical Hardware
2.2 Software Framework
2.2.1 Learn ROS
2.2.2 Plan ROS Framework
2.2.3 Implement ROS Framework
2.3 Voice Command System
2.3.1 Determine Voice System
2.3.2 Build Voice System
2.3.3 Test Voice Command Input
2.4 Vision System
2.4.1 Determine Vision System
2.4.2 Build Vision System
2.4.3 Test Vision Data Input
2.5 Actuated Manipulation
2.5.1 Define Robot Arm Kinematics
2.5.2 Implement Motion Planning
2.5.3 Test Motion Planning
2.5.4 Implement Trajectory Controller
2.5.5 Test Trajectory Controller
2.6 Safety System
2.6.1 Hardware Safety Stop
2.6.2 Software Safety Stop
2.6.3 Test Safety System
2.7 Integration & Validation
2.7.1 Subsystem Validation
2.7.2 System Integration
2.8 New Task Development
2.8.1 Determine Task
2.8.2 Implement Task
2.8.3 Validate Task
2.8.4 Integrate Task
2.9 Project Management
2.9.1 Manage Work
2.9.2 Manage Finances
2.9.3 Manage Schedule
2.9.4 Manage Risk
2.9.5 Review Design Documentation

Figure 15 - Updated WBS Structure Level 3

A major difference between our original WBS in the Conceptual Design Report and our current WBS is the addition of "2.8 New Task Development". With the overwhelming success of the Spring semester progress, we feel that we can add another task to develop in the Fall on top of what was originally planned. With these updated work packets we were able to understand our timeline in regards to our schedule and when key milestones should take place in the Fall semester.

7.2. Schedule

We are currently scheduled to begin right on track with the start of the semester. There are three main phases in the Fall development of the COBORG platform:

- Concept Development
- Subsystem Development
- System Integration and Validation

"Concept Development" is a short sprint at the beginning of the semester that focuses on creating the design foundation for the new task we are incorporating into the COBORG. After the short sprint, we will commence with the development of the new task, the development of enhancements to the core subsystems, and the improvement of the COBORG hardware framework. Finally, we will have six weeks of integration and validation to ensure that all components of the COBORG work and are ready for FVD. The "Fall Work Breakdown Schedule", generated from our WBS, can be found below in Figure 16. Below it, in Table 14 are the major system development milestones that will be achieved during the fall semester to achieve our operational goals:

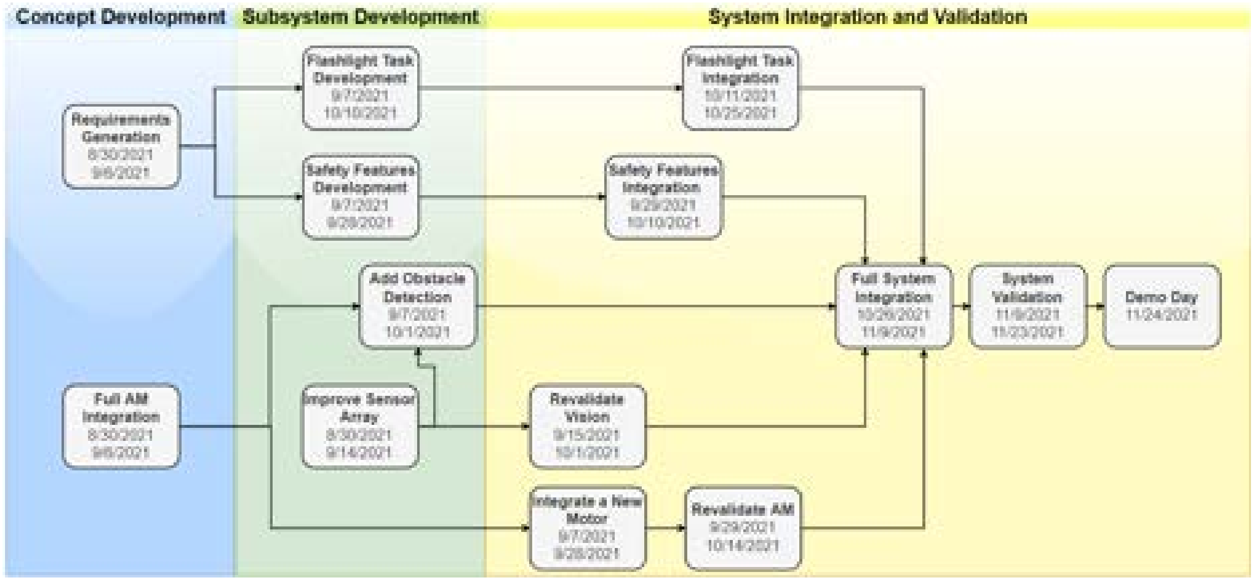

Concept Development
Subsystem Development
System Integration and Validation
Requirements Generation 8/30/2021 9/6/2021
Full AM Integration 8/30/2021 9/6/2021
Flashlight Task Development 9/7/2021 10/10/2021
Safety Features Development 9/7/2021 9/28/2021
Add Obstacle Detection 9/7/2021 10/1/2021
Improve Sensor Array 8/30/2021 9/14/2021
Revalidate Vision 9/15/2021 10/1/2021
Integrate a New Motor 9/7/2021 9/28/2021
Revalidate AM 9/29/2021 10/14/2021
Flashlight Task Integration 10/11/2021 10/25/2021
Safety Features Integration 9/29/2021 10/10/2021
Full System Integration 10/26/2021 11/9/2021
System Validation 11/9/2021 11/23/2021
Demo Day 11/24/2021

Figure 16 - Fall Work Breakdown Schedule

Table 14 - Major System Development Milestones for Fall

Date	Milestones
06-Sep-2021	Full Actuated Manipulation Integration
28-Sep-2021	New Motor Integrated
10-Oct-2021	Safety Features Integrated
25-Oct-2021	Flashlight Task Integrated
09-Nov-2021	Full System Integrated
23-Nov-2021	Full System Validated

7.3. Test plan

7.3.1. Capability Milestones

The following table, Table 15 contains the key milestones that will be achieved for each progress review during the fall semester. These coincide with our Fall schedule and will act as checkpoints to gauge our progress relative to the plan we created:

Table 15 - Key Milestones for Fall

Date	Milestones
PR 7: Early September	Requirements Generation Full Actuated Manipulation Integration
PR 8: Mid September	Improved Sensor Array
PR 9: Early October	Revalidate Vision System Safety Features Development Add Obstacle Detection Integrate New Motor
PR 10: Mid October	Safety Features Integration Flashlight Task Development Revalidate Actuated Manipulation
PR 11: Mid November	Flashlight Task Integration Full System Integration
PR 12: Late November	System Validation

7.3.2. Fall Validation Demonstration

The Fall Validation Demonstration will be held in the basement of Newell-Simon Hall in Lab B512. The equipment needed for this demonstration includes the COBORG backpack, the test structure used in the spring validation tests, the testing board used in the spring validation tests, a stopwatch, a measuring tape, a laser pointer, a power drill, 8 screws, record sheets, and a plate with shiny and matte finishes on either side. An area of roughly 10 feet in diameter will be required for the test itself.

The first test will be a validation of the new flashlight function added to the robot. The accuracy and usefulness of the system will be tested in a method that will be determined after requirements generation is completed at the start of the fall semester. The initial assumption is that it will function similarly to the vision and actuated manipulation demonstrations completed during the spring validation demonstration.

The second test will be a full use case scenario. A user will put on the COBORG backpack, hold a panel above their head (against the testing structure) and command the COBORG to move to the target. Once the COBORG has secured the part, the operator will release their hands and use a power drill and screws to attach the panel to the testing structure. The operator will then command the COBORG to return home. Once it has reached home, the user will put their hands back on the panel and command the COBORG to move to the target. After the COBORG has once again secured the part, the user will remove the screws with the drill, hold the part, and then command the COBORG to return home. This process will be repeated several times, using either a matte or shiny surface finish (two different sides of the panel) each time. The COBORG's range may be demonstrated by a variety of different testing structure poses. Almost all of the requirements will be validated by a successful use case. Other requirements will be achieved by timing the process and making sure that it takes less than the allotted time and that the COBORG can operate for the allotted minimum battery life (20 minutes).

The remaining requirements will be tested in a nonfunctional miscellaneous test which will involve measuring various features of the system, such as the total weight of the backpack. In addition, the various safety features added will be tested for reliability and ease of use. Specific requirements, and the method(s) to validate them, will be determined at the start of the Fall semester during the "requirements generation" phase.

7.4. Parts List and Budget

The parts list detailed in Table 16 is actual and covers our expenditures in the Spring semester. Our goal was to retain a majority of the project funds ($4,000) to allow for hardware expansion in the Fall. We were able to achieve relatively close to our goal budget, at $3,300. The major difference in our Concept Design Review budget and the actual spending was the addition of a $699 computer replacement which was not originally factored as a high likelihood in the risk assessment. However, we did have funds allocated for that risk and took the measures necessary to ensure success in our project:

Table 16 - Spring Budget Spent

No.	Part Name	Cost	Quantity	Total Cost
1	Voice Subsystem Parts	$34.98	1	$34.98
2	Robot Hardware	$119.71	1	$119.71
3	Supplementary Items	$501.61	1	$501.61
4	Intel Realsense T265	$199.99	1	$199.99
5	Computer Parts	$152.46	1	$152.46
6	Nvidia Jetson Xavier AGX	$699.99	1	$699.99
Total				**$1708.74**

We anticipate that our expenses in the Fall semester will be roughly $1,500 to update the hardware components of the COBORG platform. Included costs are mainly related to the fabrication of a new housing and updates to the safety of the system. From the remaining $3,300, a $1,500 deficit leaves us with a buffer of $1,800 to be used for risk mitigation and unexpected expenditures. Table 17 below shows the estimated breakdown of the Fall semester budget:

Table 17 - Fall Budget Expenditure

No.	Part Name	Cost	Quantity	Total Cost
1	Fiberglass Shell Assembly	$400.00	1	$400.00
2	Laser Cut Acrylic Base	$200.00	1	$200.00
3	Carbon Fiber Tubing	$125.00	1	$125.00
4	Electrical Components	$300.00	1	$300.00
5	T-Slotted Aluminium Assembly	$500.00	1	$500.00
Total				**$1525.00**

7.5. Risk Management

An important facet of every project is taking into account potential risks that may arise as the project progresses forward. These risks span from technical to programmatic, and from inconsequential to “show-stopping”. To ensure that our project progresses forward successfully, we have taken into account project risks and detailed tasks taken to mitigate the severity or likelihood of these risks as shown in Table 18 below:

Table 18 - Top 11 Risks Mitigation Plans

Risk	Label	Mitigation Plans
Hebi motor module dies	RT1	Ensure at least one spare is secured.
		Reduce budget to $2,000 to afford spare
Main computer dies or does not perform to our standards	RT2	Work out of Cloud, Github Repository
		Ensure spare workstation is available
		Budget $1300 to purchase spare (Zotac)
Estop devices malfunction	RT3	Inquire Biorobotics lab for spare component
		Inquire John's inventory for spare device(s)
		Ensure Estop device is fail-open
TCP/IP connectivity is lost	RT4	Inquire Biorobotics lab for spare
		Inquire John's inventory for spare device(s)
		If no spare, allocate funds to purchase spare
Water damage to COBORG System	RT5	Seal backpack and perform liquid spill test
		If operating outside, ensure weather is favorable
		Ensure all liquids are 6ft from robot exoskeleton
End Effector Breaks	RT6	3D print files downloaded into our Google Drive
		3D printed spares on hand
Team lacks ROS fundamentals by start of spring semester	RR1	Execute plan to learn ROS over winter break
		Enforce one week boot camp before Spring
		Hire ROS SME to build framework for project
Unable to work 10hrs/week/member on MRSD project	RR2	Offload work amongst team in certain situations
		Treat MRSD project deadlines as HW deadlines
Hazard occurs on user while wearing robot	RR3	Be prepared to act when the issue arises
		Brief and prep the user on proper procedure
		Perform on-the-rack run through before testing
		Set torque limits and vision obstacle avoidance
Member contracts COVID	RP1	Follow CMU pandemic safety procedures
		Create simulation of system in ROS
MRSD program gets disrupted due to COVID pandemic	RP2	House robot outside of lab environment
		Create simulation of system in ROS
Our sponsor graduates in the spring of 2020	RP3	Confirm Julian can commit time after graduation
		Acquire non-CMU contact information

By taking into consideration these risks, which span technical, resource, and programmatic concerns, we have ensured that none of the risks found have a critical impact or likelihood as the project is executed. Figure 17 demonstrates the risks before mitigation and after:

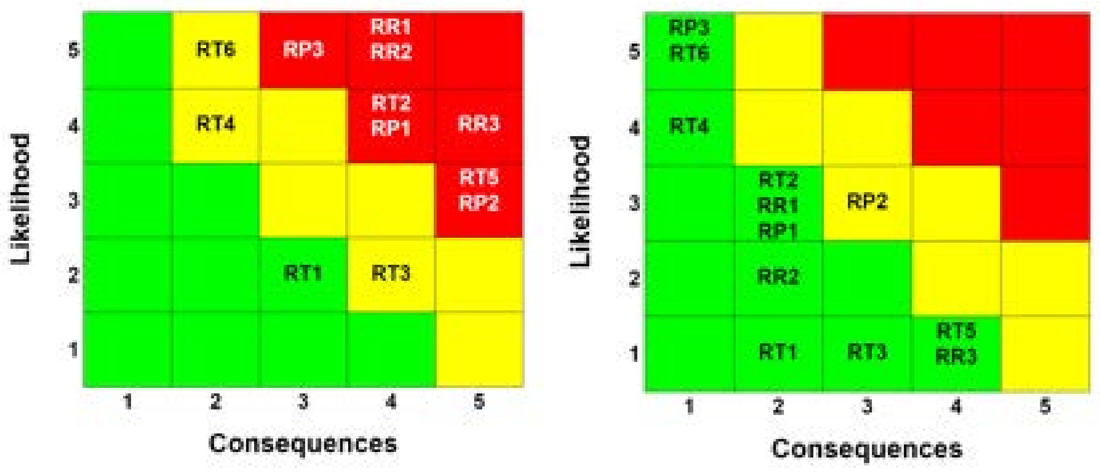


Figure 17 - Risk Mitigation Before and After

8. Conclusions

8.1. Lessons Learned

While overall we were able to execute well on the plans we made in the previous semester, there were still several challenging aspects to this project that we encountered. The biggest lesson learned is that there never was enough time to complete everything we scoped out, as project course assignments and elective assignments quickly piled on. We had to be very thoughtful in how we budgeted time for the project and eventually settled on working continuously on the project during the weekends while leaving course assignments for the weekdays. With this strategy, we were able to have five people working a total of 50-100 hours weekly to stay on track with project objectives. The other biggest lesson we learned was that documentation proved to be helpful in remembering our thought process as we developed the COBORG system. We took meeting minutes weekly and used a versioning control system to keep track of what was edited in the code. We also made sure to create "readme" guides for every subsystem, so that those who were not the subject matter experts could quickly gain expertise and begin supporting other subsystems when necessary.

8.2. Key Fall Activities

Our plans for the Fall will be centered around creating new use cases for the COBORG system. As of now, we want to create a directional flashlight arm that will point a light at the work envelope of the operator. We may change this as we refine this flashlight concept over the summer and at the beginning of the Fall semester; however, our overall goal is to create a new task that is an accessory to the original overhead assistance task. This will allow us to build upon the concepts that we learned in the Spring semester and will be of great benefit to our learning outcomes in the upcoming semester. We also are looking to create a product-ready MVP, so that the COBORG system can coincide with our business course pitch. The Fall semester is where our MRSD project will be tied together, and we look forward to creating a collaborative backpack platform that will change the world.

9. References

[1] Owano, N. (2018, June 21). Ford, exoskeleton company address strain in overhead tasks. Retrieved December 18, 2020, from https://techxplore.com/news/2018-06-ford-exoskeleton-company-strain-overhead.html

[2] Farris, Riley & Pitt, LLP. (n.d.). Retrieved December 18, 2020, from https://www.frplegal.com/industrial-accidents/8-most-common-workplace-injuries-for-factory-workers/

[3] Bruggisser, F. (n.d.). Yolo-hand-detection. Retrieved December 18, 2020, from https://github.com/cansik/yolo-hand-detection