Shop cleaning vacuum, Navigation: Project Closeout

Isaac Marquez: marquezi@oregonstate.edu

Jeremy Duke: dukeje@oregonstate.edu

Abdallah Nawras: nawrasa@oregonstate.edu

Oregon State University

May 2021

Contents

1 Design Impact Statement

Our team's project for Oregon State's senior design class is to create an autonomous shop cleaning vacuum for Professor Don Heer's personal shop. Below we have assessed some of our potential design impacts.

1.1 Public Health, Safety, and Welfare Impacts

One of our main responsibilities for our team is to select a method/algorithm that navigates the vacuum around at least 80% of the shop floor. If we pick an algorithm that successfully performs this task, then we have successfully cleaned the shop floor. This will have significant health and safety benefits to the stakeholders and people interacting with the robot. This will keep the shop floor clear of debris that could pose a hazard to the inhabitants of the shop, which at times could be both people and pets. Furthermore it can improve the air quality and reduce the chances of developing unwanted infestations(article by thrive global on health benefits of vacuuming)[1].

1.2 Cultural and Social Impacts

There are many cultural impacts that should be considered; for example, mistrust that people have in robotic technology. Cameras might make some users feel uncomfortable around a robot that is able to view them while a lidar solution would not have this issue. These types of bias are important because the people around the robots should not feel uncomfortable. It would be great if there were a way to create trust in the systems that we create, but corporations have a history of abusing data collection for monetary gain [2]. It does seem very likely that being weary of cameras can be a positive reaction to this technology. As engineers, the reduction in sensing that could cause harm seems like a great solution to these issues.

1.3 Environmental Impacts

The use of materials found in microprocessors relies upon the mining of rare materials. These are not commonly ethically sourced and using these processors further promotes the exploitation of this unsafe or regulated labor as well as draining a finite resource of our planet. The creation of computers produces thousands of tons of CO_2 emissions every year. Nvidia, the creators of the Jetson nano (the processor we are using) report creating 2,356 tons of CO_2 in 2018 as a result of the manufacturing of their chips [3]. This is a reduction in CO_2 emissions compared to past years, but this does harm the environment.

1.4 Economic Factors

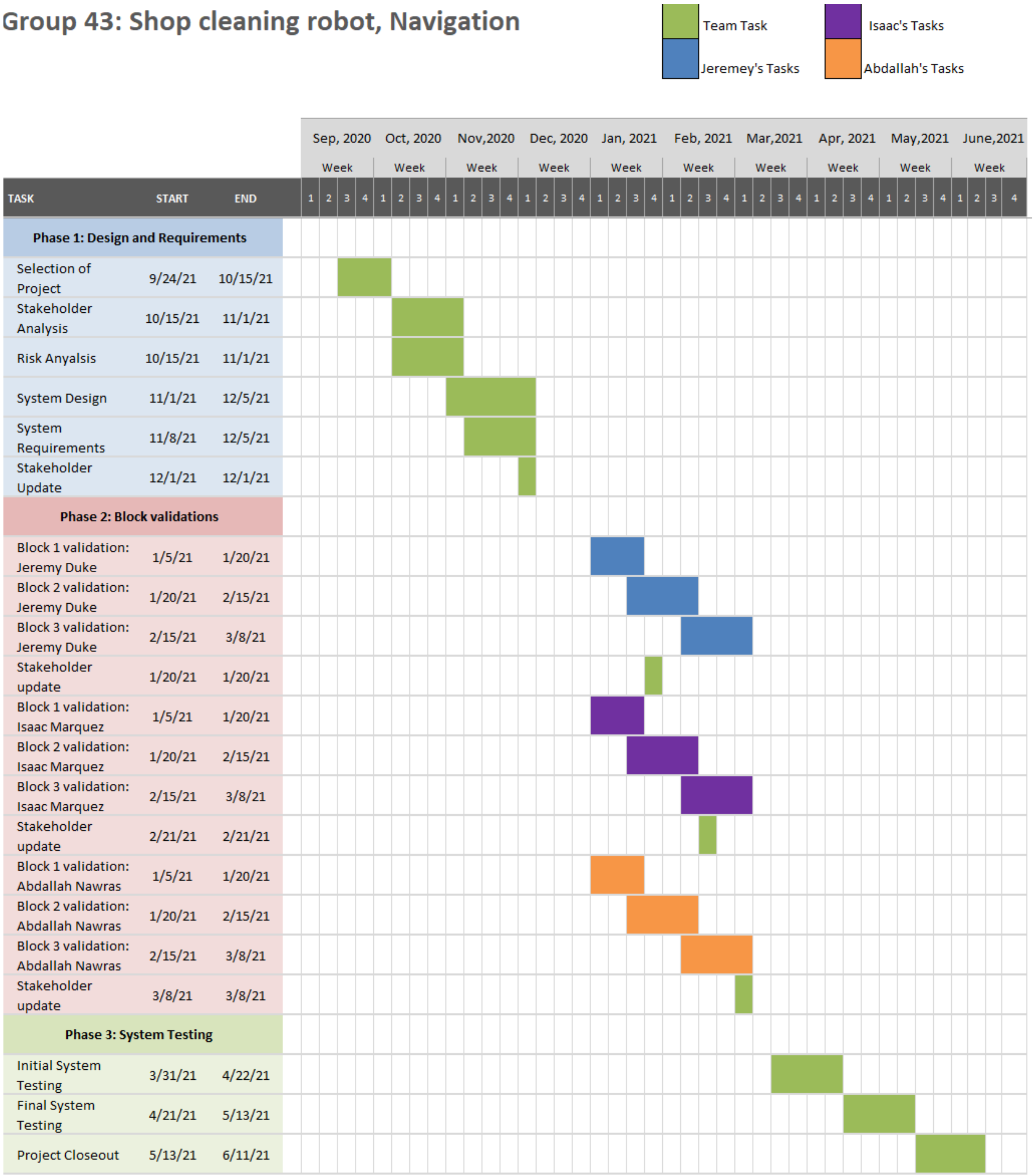
Economically, using an automated cleaning system seems beneficial, but a reduction in cleaning jobs might greatly impact communities that rely upon them for work. Large populations all around the world get paid and support their family by doing what is becoming a very automatable task. More than forty percent of tasks that cleaners do could be automated away by 2030 [4].

1.4 Conclusion

This project operates in a single household and interacts with a limited number of individuals. This may lead some to assume that we do not have to assess the impacts of our design. But as this section has shown there are many impacts to be considered, and if neglected could lead to unwanted negative outcomes.

2 Project Timeline

Below is our timeline of major events over the last 9 months of research and development into our senior design project.



TASK	START	END
Phase 1: Design and Requirements		
Selection of Project	9/24/21	10/15/21
Stakeholder Analysis	10/15/21	11/1/21
Risk Anyalsis	10/15/21	11/1/21
System Design	11/1/21	12/5/21
System Requirements	11/8/21	12/5/21
Stakeholder Update	12/1/21	12/1/21
Phase 2: Block validations		
Block 1 validation: Jeremy Duke	1/5/21	1/20/21
Block 2 validation: Jeremy Duke	1/20/21	2/15/21
Block 3 validation: Jeremy Duke	2/15/21	3/8/21
Stakeholder update	1/20/21	1/20/21
Block 1 validation: Isaac Marquez	1/5/21	1/20/21
Block 2 validation: Isaac Marquez	1/20/21	2/15/21
Block 3 validation: Isaac Marquez	2/15/21	3/8/21
Stakeholder update	2/21/21	2/21/21
Block 1 validation: Abdallah Nawras	1/5/21	1/20/21
Block 2 validation: Abdallah Nawras	1/20/21	2/15/21
Block 3 validation: Abdallah Nawras	2/15/21	3/8/21
Stakeholder update	3/8/21	3/8/21
Phase 3: System Testing		
Initial System Testing	3/31/21	4/22/21
Final System Testing	4/21/21	5/13/21
Project Closeout	5/13/21	6/11/21

3 Scope and Engineering Requirements Summary

3.1 Requirements Description

For our team's project we are responsible for the navigation of the autonomous vacuum robot. This task can be achieved in many different ways, but we have chosen to use a camera and central microprocessor, specifically the NIVDIA Jetson. With these decisions in mind we were able to create the requirements listed below. Our requirements have three main objectives: vacuum the shop floor as efficiently and effectively as possible, avoid hazards and obstacles, and navigate to and from the charging base.

3.2 Requirements table

Name	Project Partner Requirement	Engineering Requirement	Verification Method	Testing Process	Test Pass Condition	Evidence Link
Avoid prohibited areas	Avoid areas that should not be cleaned.	The system shall stop or redirect the system to avoid entering prohibited areas specified by the project partner.	Demonstration	1. Initialize the systems camera 2. simulate the shop vacuum navigating towards a prohibited area by moving the camera in the direction of a qr code 3. demonstrate that the system sends navigation commands to avoid the prohibited area 4. repeat steps 2 and 3 with the camera coming from the front, the right, and the left	If the system sends navigation commands to avoid the prohibited area, then the test has passed.	https://media.oregonstate.edu/media/t/1_00x1mesr
Clean majority of the floor space	The robot should be able to clean the majority of the shop floor each time that it cleans.	The system shall navigate over 80% of a 10'x10' square in under an hour	Analysis	1.Run the test program I. Start the Sensor Node (with simulation flag set on to simulate the environment) II. Start the Motor Node III. Save the graph of	If the graph of the path covers more than 80% of the available space, then the test passed.	https://kaltura.oregonstate.edu/media/t/1_rx40qff0

				pathing locations 2. Verify that the path covers the whole 10*10 square visually.		
Localize the system	Make the system return to the charger and clean the shop floor.	The system shall be able to localize its own relative positions within the environment to within 1 foot of actual location, 80% of the time it is running.	Analysis	1.Run the test program I. Start the slam node II. Start the bag (this replays prerecorded lidar data to ROS) III. Wait 30 seconds IV. Save the map as a .tif 2. Verify that the map contains the path through the environment.	If the .tif contains the localized path through the environment with less than 80% missing, then the test passed.	https://kaltura.oregonstate.edu/media/t/1_pz5yz1li
Map our pathable environment	Map the shop floor to efficiently and effectively clean it.	The system shall map 80% of the system's path through the environment.	Analysis	1.Run the test program I. Start the slam node II. Start the bag (this replays prerecorded lidar data to ROS) III. Wait 30 seconds IV. Save the map as a .tif 2. Verify that the map contains the pathed environment	If the tif contains a map of the environment with more than 80% of the pathed environment, then the test passed.	https://kaltura.oregonstate.edu/media/t/1_32ft3ooo
Navigate back to charger	The robot will return to the charger after completing a cleaning cycle.	The system shall navigate back to the charging station.	Demonstration	1. Initialize the systems camera 2. simulate the shop vacuum navigating towards the charging station by moving the camera in the direction of	If the system sends navigation commands to the low-level microcontrol	https://media.oregonstate.edu/media/t/1_8cvdd7z2

				the charging station 3. demonstrate that the system shall send navigation commands to successfully get the robot back to the charging station	ler that results in the robot navgiating back to the charging station, this ER has been met.	
React to environment hazards	The robot will stop or change directions if it encounters a hazard such as a shop cord or a fluffy cat.	The system shall stop or redirect the system to avoid hazards reported by the low level microcontroller	Analysis	1. Connect Arduino to the Jetson Nano 2. Run test program I. Start up the system II. Instruct the system to read motor status from Arduino III. Read and log the Arduinos motor status 3. Verify that the system recognizes the hazard, and stops or redirects the path of the robot.	The test passes if the system receives the alert for the hazard from the microcontroller, recognizes the hazard, and stops or redirects the path of the robot.	https://media.oregonstate.edu/media/t/1_o1pcjd2p
Receive sensor data	There must be a high level navigation CPU that communicates with the low level sensor control processor.	The system will support the status updates listed in the 'Command Interface' Google Doc: https://docs.google.com/document/d/1mbS1jySUtclh21ppn_JF7uFOdZLku_NinsdbgMOD7xY/edit	Test	1. Connect Arduino to the Jetson Nano 2. Run test program I. Start up the system II. Instruct the system to read all the sensor data from the Arduino III. Log the Arduinos sensor data and the Jetson Nanos Sensor data 4. Verify that the Arduino received the navigation	If the sensor data was transmitted and stored on the Jetson Nano and the sensor data is identical, then the test passed.	https://media.oregonstate.edu/media/t/1_e7q9ahc4

				commands		
Send navigation data	There must be a high level navigation CPU that communicates with the low level sensor control processor.	The system will support the commands listed in the 'Command Interface' Google Doc: https://docs.google.com/document/d/1mbS1jySUtclh21ppn_JF7uFOdZLku_NinsdbgMOD7xY/edit	Test	1. Connect Arduino to the Jetson Nano 2. Run test program I. Start up the system II. Instruct the system to send all the navigation commands to the Arduino III. Read and log the Arduinos motor status 3. Verify that the Arduino received the navigation commands	If the Arduino received all the navigation commands, then the test passed.	https://media.oregonstate.edu/media/t/1_w0uvg00 l

4 Risk register

4.1 Risk Register Description

This project had a lot of risks that we expected because of the limitations that the pandemic put on us. The restrictions put in place for our safety did harm our ability to assemble the project as one system. We had to quickly adapt our expectations from working collaboratively in person on a single system to simultaneous independent work. To assist in working and developing remotely we utilized a variety of hardware including virtual machines, Raspberry pi's, and Jetson Nano.

4.2 Risk Register Table

Risk ID	Risk Description	Risk category	Risk probability	Risk impact	Performance indicator	Responsible party	Action Plan
R1	Electrical shock	Safety	5%	L	Injury reported by student	Jeremy Duke	Retain
R2	Theft of System	cost	2%	H	Our robot is missing	Isaac Marquez	Retain
R3	Decoupling of connections	Technical	20%	L	Unexpected behavior	Jeremy Duke	reduce
R4	Battery failure	Technical	5%	H	Stops holding a charge and possibly the battery stops charging	Jeremy Duke	retain
R4	**Inter-team communication delay:** Due to our large team communication between our subgroups may be a risk factor.	Process	30%	M	On-time completion of assignments, and prompt answers to questions	Isaac Marquez	Reduce
R5	**Remote constraints:** Due to working remotely, we may not be able to easily test our system.	Timeline	80%	M	Experience roadblocks due to remote constraint	Abdallah Nawras	Reduce
R6	**Incompatible interface with microcontroller:** It will take a lot of coordinating to manage interfacing between subsystems.	Technical	15%	L	Change of microcontroller or interfacing protocol.	Isaac Marquez	Transfer
R7	**Equipment availability:** We may not have access to equipment for our subsystem.	Technical/Cost	10%	L	Expensive test equipment, or no access or test equipment.	Jeremy Duke	Retain
R8	**Delay of microcontroller:** Our other subteams may have a substantial delay in their design/choice of embedded microcontroller.	Timeline	40%	H	Subteam does not inform us of their choice until weeks into the quarter.	Abdallah Nawras	Transfer
R9	Team delay	Timeline	20%	H	Due dates	Jermey Duke	Retain

R10	Program performance	Operating system	40%	H	Testing oftenly	Isaac Marquez	Reduce
R11	Adding new requirements	Technical	40%	H	Emails	Isaac Marquez	Reduce
R6	Removing current requirements	Technical	30%	M	Emails	Abdallah Nawras	Reduce
R7	Lack of communication	Technical	20%	M	No able to communicate	Isaac Marquez	Retain

5 Future Recommendations

Recommendation	Reason	Starting Point
Implement a number of cleaning modes	After creating the navigation node, it became clear that using a bug algorithm to cover the floor was effective, but inefficient because most of the floor was clean more than one time. By utilizing a series of pathing methods that target specific areas of the floor (ie, walls, open spaces, under furniture) and limits the time the robot recleans an area, the cleaning time and power consumption can be reduced.	Create a ROS msg that holds the current cleaning mode so you can implement separate logic for each cleaning mode as well as navigating to the charger.
Identify hazards remotely using the a camera and object detection	Our initial design goal included identification of wires to reduce the likelihood of harming the system but this was reduced in winter term to responding to hazards reported from the brush sensors and bump detectors. At the time, we had assumed the system would have a GPU bottleneck if it had to go monocular SLAM as well as object detection. By switching to LIDAR SLAM via hector SLAM we use our CPU for SLAM instead of GPU.	1. Use a pre trained classification model like Alexnet to update other ROS nodes when classifying specific images 2. Work to train a model that classifies just the hazardous items that might exist in a woodworking shop like cables, tools, etc.
Create or attain sensor reading from the sensors that will be used on the robot	Being a team that works closely with two other teams, it is important to be able to use the data created by those other teams. By asking for data samples early in the process, you can verify that data will work well with your system.	3. Attach the sensors to a ROS system 4. Record a bag that contains all the data sources you would expect on the final system
Utilize a 3D simulation for testing of the navigation in a realistic environment	Creating a navigation system without access to the final robot is exceptionally difficult. By working with the other teams to define their system in a simulated environment, you can reduce assumption and make implementation easier.	1. Create a Gazebo simulation that utilizes the turtle bot as the system. 2. Work with your other teams to create a better representation of the robot in the gazebo environment 3. Work with your project partner to create a representation of the woodworking shop within the Gazebo environment
Work with the other teams to set a completion data far before the final tests	By completing the multi team system early, you can test the navigation code on the real robot and fix the expected issues. This will also help all teams to stay on track and be able to fix any errors or issues before the initial checkoffs.	Start by using a discord channel where all the teams can communicate.Then make channels for individual teams, this way teams will be able to check the chats for any other team and check their process and ask questions.Then setup weekly multi team updates in the first term and agree upon deadlines:

		- Sensor decisions - Example sensor data collection - Simulation design and implementation - Final hardware assembly - Testing hardware in the woodworking shop
Create a unit test for verification of ongoing functionality of the system between version	This can help reduce compatibility issues early on in development that are created by merging multiple branches together.	After creating unit tests for your python functions you can use github actions to automate the running of these upon merge request.
Verify that team members have an understanding of the tools used to create the system	The tools used to create a control system for a robot can be complex and require understanding predefined functionality of the overarching framework. We used ROS for our development, but we only learned enough for each team member to complete their individual blocks. If we had a more diverse understanding of the framework, we would have been able to work together to fix issue and implement feature.	By each going through the ROS tutorials, you can verify that each team member is able to understand and help with completion of the system.
Share code with other teams working on the robot using version control such as GitHub or GitLab.	Since the other teams don’t communicate with the navigation team as often, it is more important that they are able to reference the current well documented code as well as reference key definitions in the wiki.	For sharing code, create a GitHub repo where you can update and share the current version. Then create a wiki that stores all the shared information between each team like sensor and motor documentation, decisions about sensor placement, and shared team deadlines.

Reference List

[1] Developer, D. (2020, February 10). The health benefits of using the vacuum cleaner. Retrieved April 16, 2021, from https://thriveglobal.com/stories/the-health-benefits-of-using-the-vacuum-cleaner/

[2] B. X. Chen, “Your Doorbell Camera Spied on You. Now What?,” The New York Times, 19-Feb-2020. [Online]. Available: https://www.nytimes.com/2020/02/19/technology/personaltech/ring-doorbell-camera-spying.html. [Accessed: 17-Apr-2021].

[3] “2018 NVIDIA SOCIAL RESPONSIBILITY REPORT,” Nvidia.com, 2018.

[4] M. Muro, R. Maxim, and J. Whiton, “Automation and Artificial Intelligence: How machines are affecting people and places,” Jan-2019