Mobile Robot Package Delivery

Project Closeout
Team 47

Team Members
Ashley Pettibone: pettiboa@oregonstate.edu
Wyatt Deck: deckw@oregonstate.edu
Leon Tran: tranleo@oregonstate.edu

Table Of Contents

Design Impact Statement

Public Health, Safety, and Welfare Impacts

In the past, Starship Technologies robots have collided with vehicles. In one incident, a car was waiting at an intersection when the robot, "traveling at a low speed of 2 mph and scuffed the bumper of a car" [1]. The author's team is using a lidar sensor for detecting close objects. The requirement involves avoiding a 25x25in object placed in front of the robot. This will confirm the lidar can detect and avoid objects such as cars and people. In addition, bump sensors will be installed that will cut power to the robot if a collision is detected.

Cultural and Social Impacts

The primary social impacts related to this team's domains all have to do with the camera. Since we will be streaming the live camera feed to the website, there could be concerns over privacy. To remedy this, we have developed a system to detect and blur any face the camera sees. However, this creates a new problem related to training biases. There have been many machine learning systems in the past related to human detection that have exhibited bias towards functioning better on white individuals and being less effective on people with brown or black skin [2]. We are using a third party trained network, so we will need to do thorough research and testing with the system to ensure it does not exhibit these biases.

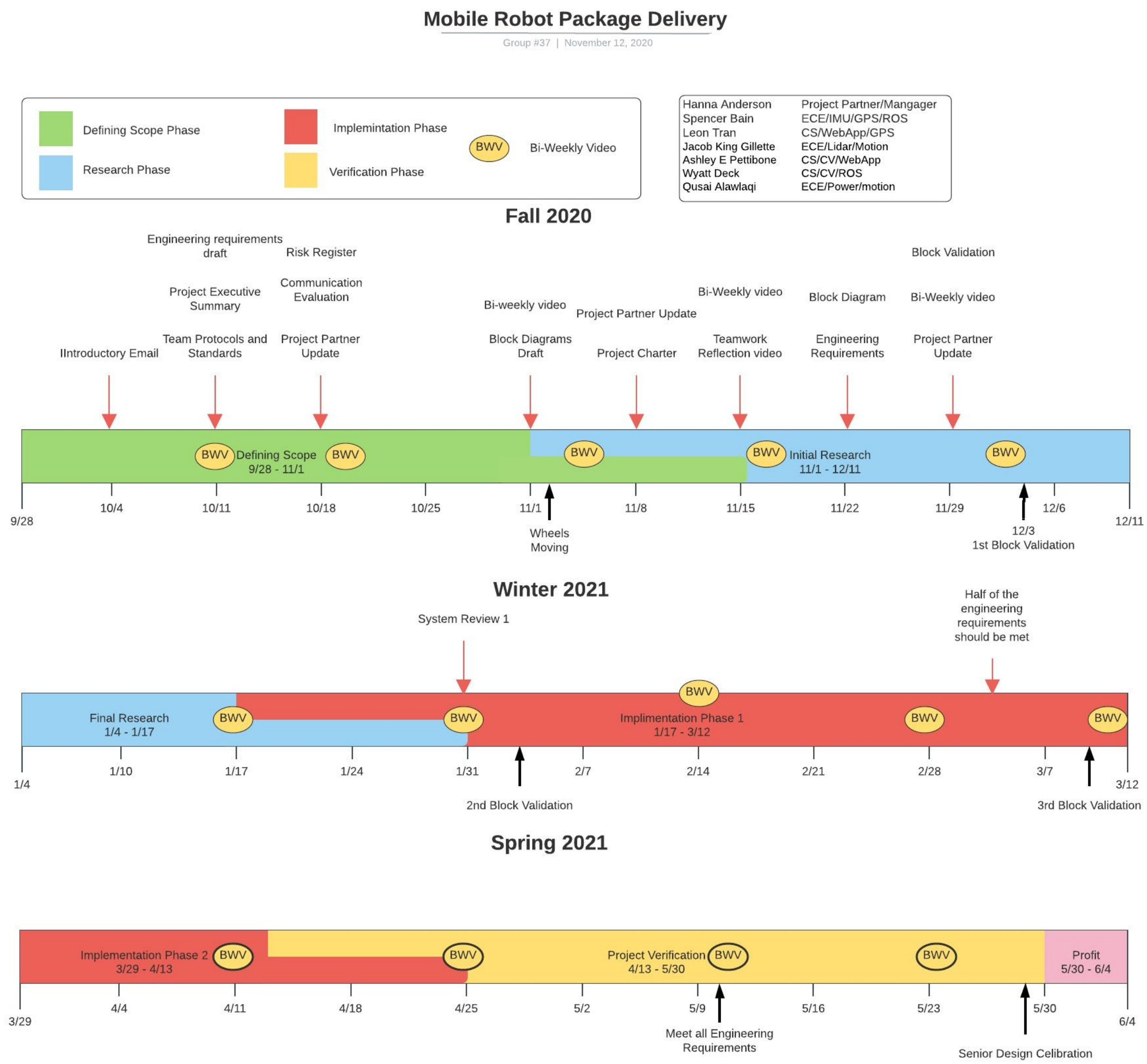
Environmental Impacts

As our robot uses many different materials to make it, and operates indoors and outdoors, there are a few impacts on the environment, some positive and some negative. The robot is made of metal and other materials and uses a raspberry pi for the computer, so as mentioned before it will eventually have some e-waste when parts go outdated or are damaged beyond repair. As parts will have to be replaced and updated from time to time "unsustainable exploitation of natural resources could be increased, especially if our dependency on rare metals for production of electronic equipment further deepens"[3]. We also used a 3D printer to print some of the pieces on our robot. As our robot will also need to be recharged, it will be consistently using electricity as well, which in turn can raise greenhouse gas emissions. One thing to note is that our robot base is made from recyclable wheelchair materials, so it is also in some way reusing materials instead of buying brand new.

Economic Factors

While our robot is unlikely to be significant economically, the users of the robot will be primarily concerned with how well the system accomplishes its task, mainly does it deliver packages quickly and safely. The safety component has already been considered with regards to pedestrian safety. As for quickness, again the robot travels at a relatively slow speed (no faster than five mph) for safety reasons, so to improve speed, the robot needs to take faster routes. Because of this, it is important that the robot takes the optimal path to its destination, which is a requirement of the global planner that has been accomplished using the algorithm A*, which is both sound and optimal for path finding problems [4].

Project Timeline



Scope and Engineering requirements Summary

The scope of this project is to take the wheelchair base already constructed with a motor driver and add autonomous and manual controls to the robot to navigate simple outdoor environments, such as the multiple quads on campus. The Computer Science team will handle the code-heavy parts such as the web app, object detection, and image processing.

The Computer Science team is responsible for eight engineering requirements (ER). The first requirement is sidewalk edge detection, the system will stay within the edges of the sidewalk while moving. The second ER is the live camera feed, the webpage will display a live image taken from the robot's camera every 15 seconds. The third ER is notification, the system will update the webpage and inform users with the current status of the system. The fourth ER is web app user input, where the webpage will allow users to input the sender and receiver for a delivery. The fifth ER is path creation, the system will create a path from each starting location to each ending location. The sixth ER is object detection, where the system will detect faces from the camera feed. The seventh ER is blurring faces in live-camera feed, the system will blur all human faces from object detection. The last ER is live map update, there will be a map on the webpage that will display the robot's GPS location.

Engineering Requirements

Name	Client Requirement	Engineering Requirement	Verification Method	Test Process	Pass Condition
Blurring facing in live-camera feed	The website live feed will display blurred faces of people to respect their privacy	The system will blur all human faces that are within 3 meters of the system's forward field of view with at least 90% accuracy	Demonstration	1. Open the website (camera will start streaming) 2. Have a human face be at a maximum of 3 meters in front of camera 3. visually inspect blurred image on the website 4. Repeat for at least 10 faces 5. Repeat steps 1-3, but with 3 human faces in front of the camera	9 out of 10 human faces will be blurred. Multiple faces can be blurred at once. Evidence Link- https://drive.google.com/file/d/12jlcR2ew0v96zAzBxNyNbx_Nf9FavmiD/view?usp=sharing
Cost Map	The robot will avoid trying to path through major static obstacles.	The system will display multiple available paths for the robot to take between two points that on inspection only include areas that are sidewalk or crosswalks only for a set of at least 10 locations on campus.	Demonstration	1. Randomly select a valid start and end GPS destination in the MU quad (valid = on a sidewalk/footpath, non-grassy area) 2. Run the path finding algorithm altered to output a visual display of paths overlaid onto a map of the operational area 3. Inspect the output to determine correctness Correctness is defined as staying on sidewalks/footpaths 4. Repeat 1 - 3 for 10 total origin and destination pairs	Of all paths generated, no more than 1 is invalid Evidence Link- https://drive.google.com/file/d/1i75KtNP-cATADL-0iyhAzXUs5alq-iBQ/view?usp=sharing

Edge Detection	The system should stay on the sidewalk	While moving, the system will stay within the edges of the sidewalk traveling for 30m.	Test	1. Turn the robot on 2. The robot will be at one side of the MU quad, we will tell the robot to start traveling 3. Robot will travel at least 35 meters to the other side of the MU quad, mostly in a straight line, avoiding sidewalk edges during it's journey 4. The robot should not encounter a sidewalk edge as the robot will avoid them.	If while moving the system stays within the edges of the sidewalk traveling for at least the first 30m, this ER is met. Evidence Link:
Live Camera Feed	There should be a semi-live camera feed viewable on a publically viewable web page	The system must update the publicly viewable web page with an image from the on-board camera at least every 15 seconds	Test	1. Setup a clock in front of the camera 2. Turn on the robot 3. Access the web page 4. Note of the time the clock displays 5. After 15 secs, the clock in the image should show 15 sec have passed	If the system updates the publicly viewable web page with an image from the on-board camera at least every 15 seconds, this ER is met. Evidence Link: https://drive.google.com/file/d/1tRYuQFcRYJsWsMngmw7yn5y_KW-74VTr/view?usp=sharing
Live Map Update	The website will show a map and where the system is on that map.	The system's GPS location will be visually indicated on a map, updated every 30 seconds or less. There will be a visual trail on the map showing the path the robot has taken.	Demonstration	1. Turn on the robot 2. Access the map on the webpage 3. Take note of the current location and start a stopwatch 4. Move the robot forward using the manual control 5. Stop the stopwatch when the map updates 6. Observe that the time will be less than or equal to 30 seconds 7. Observe that the robot's previous location and updated location are linked with a visual line on the map.	Observe that the time on the stopwatch is less than or equal to 30 seconds when the map is updated. There is a visual line linking the robot's previous location with its current location. Evidence Link: https://drive.google.com/file/d/1AfZunYf-Lce_A4jWuFwFe_2I06DnKhXD/view?usp=sharing
Notification	The system will notify users/webpage on the status of the package delivery	The system will update the webpage and the users with the current system status upon system status change	Demonstration	1. Open webpage 2. Run script to simulate inputs (GPS coordinates) from ECE group. 3. Check that the status on the webpage is Available" 4. Enter a sender and receiver. 5. Check the status is "Creating Route". 6. Have the script simulate robot movement. 7. Check the status is "En Route" 8. Send simulated end coordinate. 9. Check the status is "Ready for	The robot updates the status of the robot correctly for each case Evidence Link: https://drive.google.com/file/d/172Vv-Atps80ZVdSIQuJyV8aPqxmaD-VH/view?usp=sharing

				Pickup"	
Path Creation	A path should be created from each starting location to each ending location	The system will follow a valid path from its start location to its destination	Demonstration	1. Decide a starting point in operational area 2. Decide an ending point in operational area so that there are multiple possible valid paths to it 3. Query the system for a path between the two points	Chosen path stays on the walkway and is the shortest route. Evidence Link: https://drive.google.com/file/d/1Wp5eZ6P_Ua8t6K3rIXRxaHogDMEqmRSb/view?usp=sharing
Web App User Input	The web app should allow users to select the sender and receiver of packages, and delivery destination.	The system UI allows users to input sender and receiver information, and select destination from a list. The sender and receiver must be authenticated with OSU emails.	Demonstration	1. Turn on the robot 2. Access the webpage 3. Attempt to input sender and receiver information without emails. 4. Check if the system sends an error message and blocks the input. 5. Attempt to input sender and receiver information with unauthorized OSU emails. 6. Check if the system sends an error message and blocks the input. 7. Input sender and receiver information with authorized OSU emails. 8. Check if the system allows the user to select a destination from a list. 7. Select the destination.	If the system UI blocks all unauthorized inputs and allows authorized inputs to select a destination from a list, the ER has been met. Evidence Link: https://drive.google.com/file/d/1-gQzumOJikkE_sRjZbN6ns9pz-yTtZ02/view?usp=sharing

Risk Register

Over the course of this year, our team encountered many roadblocks, both anticipated and not. Among the hurdles we found that caught us off guard, the fact that the Twitch api does not allow for websites that don't have a domain to host streams and the edge detection block is now invalidated by other project features (mostly from our partner team) while still being a requirement were the most troublesome. Our team never found a remedy to these issues. Therefore, the largest risk we did not anticipate was that we would encounter problems we didn't have the time to solve.

Due to the roadblocks alluded to above, our team has gained additional experience to better tackle future problems. The most valuable lessons we have learned are to integrate sooner and be in good communication with other group members to be knowledgeable of the specifics of what they have done, be clear about what the client is asking for and be prepared when this changes, and to get help quickly after a road block is encountered. Integrating later meant that despite our blocks being well designed, they were largely stand alone and weren't made with connecting them in mind. When we found that the edge detection block was difficult and unnecessary to the success of the project, we left it alone to focus in other areas. However, the client still wanted this feature to be complete, and it never was. Lastly, we encountered many obstacles that caused significant delays when asking others could have solved the problem much quicker, wasting valuable time.

Risk ID	**Risk Description**	**Risk category**	**Risk probability**	**Risk impact**	**Performance indicator**	**Responsible party**	**Action Plan**
WD2	Code loss	Logistics	5%	M-H	Code that was there isn't anymore	Leon (tranleo@oregonstate.edu)	Reduce - Use Github
AP9	Incompatible interface	Technical	15%	H	Signals are misinterpreted or system crashes	Wyatt (deckw@oregonstate.edu)	Reduce - Test each part together add manage detailed block diagram
LT1	Sickness	Operational	<1%	H	People say they are ill and their work does not get completed by the deadline, or not checking in with communication platforms	Wyatt (deckw@oregonstate.edu)	Reduce - Try to keep healthy and wash hands often
AP8	Errors with image detection	Technical	20%	H	Can't detect or misidentifies an object in front if it	Ashley (pettiboa@oregonstate.edu)	Reduce - Hardcore testing/ training
WD5	Part damaged while operating	Cost	<5%	M-H	Something malfunctions or starts making bad sounds	Hanna	Retain

Future Recommendations

	Recommendation	Reason for recommendation	Starting point for solution
1	Integrate Early	The most complicated spaghetti code appears when trying to combine two or more written pieces of code together	Work together early to establish how pieces will connect in a specific and concrete way. This includes defining data types, ros topic names and message types, and file formats.
2	Test Each Others Stuff Early	This helps other group members understand what you're doing. Also, if someone is struggling with part of a problem, it helps if more than one person can help solve that problem. Additionally, knowing how all the part work makes presenting	Use Github. Maybe schedule a time once a week to walk through everyone's code for clarity and help.
3	Be Mindful of the Processing Capabilities of the Hardware	The system uses a Raspberry Pi as its main controller. It's a good computer, but it has its limits. It struggles to do face blurring or image processing in real time.	We've limited the frame rate, but changing the face blurring code to use C++ and/or using the multi-threaded version that's on Github would like to be able to improve that. Another option is to stream the live feed to an external computer which can process the images faster and stream to the website.
4	Computationally Expensive Tasks Should be Done in C/C++	Our face blurring was done in Python and it's slow.	The face blurring code already uses OpenCV which has a C++ version if changing this is deemed necessary.
5	Ensure You've taken all needs of a feature into account	The face blurring code does detect fast moving or not front facing faces particularly well.	Face blurring code could add object tracking to track faces and blur them, which would likely make face

			blurring in the video much more accurate.
6	Document and comment thoroughly	This helps other group members understand what you're doing. Setup documentation is the most important. It allows other people to test your code and makes integration much easier.	On Github, Readme.md files are made specifically for this job. Use them and put as much information there as possible, even if you feel like you don't need to.
7	Make a explicit web server	The current system launches the web server using php of localhost and the web page connects to ROS directly. This makes viewing the site from another device, even on the same network.	Try making a server using something like Flask (in Python) on the Pi. This can easily talk to ROS while being able to serve remote clients.
8	Find a better way to stream video	The current system streams to twitch on its own account which works, but Twitch's api has changed to not allow sites that don't have SSL certificates to host a stream.	Either figure out how to authenticate the site or use a different streaming method entirely.

References

[1] D. Z. Zoga and E. Parks, “NBC 5 Responds: What Happens if You're in a Crash With a Robot?,” *NBC 5 Dallas-Fort Worth*, 11-Sep-2020. [Online]. Available: https://www.nbcdfw.com/news/nbc-5-responds/nbc-5-responds-what-happens-if-youre-in-a-crash-with-a-robot/2442345/. [Accessed: 17-Apr-2021].

[2] L. Hardesty, “Study finds gender and skin-type bias in commercial artificial-intelligence systems,” MIT News | Massachusetts Institute of Technology, 11-Feb-2018. [Online]. Available: https://news.mit.edu/2018/study-finds-gender-skin-type-bias-artificial-intelligence-systems-0212. [Accessed: 15-Apr-2021].

[3] J. Dusik and B. Sadler, “What Effect Will Automation Have on the Environment?,” International Institute for Sustainable Development, 22-Jan-2019. [Online]. Available: https://www.iisd.org/articles/automation-environment.

[4] S. J. Russell, *Artificial intelligence: a modern approach*. Harlow: Pearson, 2016.