Max Brodeur-Urbas, Carl Elkhoury, Zakaria Essadaoui, Ayden Malik, Ashish Paul, Shaodi Sun

Student ID: 260806240, 260806273,260776919, 260627064, 260706034, 260712632

Final Project Report

Group: 20

Assignment presented to

Frank P. Ferrie

David Alister Lowther

ECSE 211: Design Principles and Methods

Friday, November 30th 2018

The undersigned members of team *20* agree that the contents of both this report and the information handed in on cd, dvd or memory key, provide an accurate representation of the work done on this course and the contributions of each team member.

Name	Signature
Max Brodeur-Urbas	______________________________
Carl Elkhoury	______________________________
Shaodi Sun	______________________________
Ayden Malik	______________________________
Ashish Paul	______________________________
Zakaria Essadaoui	______________________________

TABLE OF CONTENTS

1. Introduction

The main objective of this project is to design and construct a machine that can autonomously navigate a closed course in search of a set of colored rings, identify color, retrieve as many rings as possible, and come back to unload them at starting position. This project is designed to be accomplished by a six-member team in a period of six weeks total. Beyond that, the design should respect multiple constraints such as traversing a tunnel of fixed height and avoiding an area labelled as a “River”. Moreover, the teams have to use limited hardware components with limited software capabilities to tackle the challenge at hand.

There are two main reasons for doing this project. The first one is to have a practical experience on a design concepts by introducing us to a challenge that incorporates both hardware and software integration. Additionally, and more importantly, this project is intended to prepare us for a work environment by simulating an industry-like design process. The team were being provided with limited amount of resources and had to solve a design problem with a restrictive time budget. Learning to work with different team member’s skill sets, changing project requirements and conflicting availability. Those constraints also helped in creating a real-time design environment. During the design process, in order to achieve the goals, the team needed to explore various techniques and tools for designing, analyzing and evolving a design prototype. The design was discussed with our professor on a weekly basis imitating a real-life interaction with a client. At the end of the design process, a competition was being held to evaluate the product’s abilities.

Generally speaking, this project is intended for the team to apply the process of design methodologies on a relatively large-scale. These concepts are best learned through practice, and this has proven to be very helpful since none of the team has previous experience with those ideas.

2. Team organization – the startup of the project

A preliminary Gantt chart was created to outline the tasks that needed to be completed before each weekly meeting.
Each task performed followed a managemental chain to ensure every hour was documented and every team member had a justification for their hours spent. First each member contacted the project manager on a near daily basis to register the hours they contributed and the reason for having worked. At the end of the week the project manager would sum the hours and write a justification for each task a member participated in.

To facilitate task allocation, the Gantt chart was color coded (legend can be seen in the first tasks of the chart) to help identify which category each task fell into. The strengths and weaknesses of each member were put forward and so software tasks were primarily allocated to software leads, hardware to hardware lead and so on. The preliminary Gantt chart required a significant amount of foresight and estimation but as the project progressed, we were more experienced with estimating the length tasks causing the tasks to be reallocated dynamically as they approached. Often, when reviewing the Gantt chart for an upcoming week, it was considered too large for the designated team member to finish alone so other team members had to lend a hand. In this situation, the project manager referenced the resource allocation chart to consider the availability of members and assigned someone with free time to lend a hand. As mentioned in our capabilities document, our team had a very wide variety of skills so the project manager helping with coding or software lead documenting was not uncommon.

Tasks creations were initially inspired by the provided document titled “ECSE211-Meetings-Fall 2018 v3.1”. In this document, the expectations for each weekly design meeting were outlined provided a framework of milestones to follow. This created a rough structure of the project but was heavily documentation based and neglected much of the testing and integration of software. These tasks were created off a rough estimation how the project would progress and what was required for the beta demo. Additional team brainstorming and reference to past semester Gantt charts were used to add additional more specific tasks.

After the creation of the tasks it was important to set their estimated lengths based on how many days it was expected to take. All of our team members attended lectures primarily in Trottier so the majority of work was done between classes. As a result, our teams’ approach to work was to do a small amount of work every day rather than cram it into a single day. The most any team member would work in a day was estimated to be two hours so the “standard payment rate” in the resource allocation chart was set to a value of two. This meant that if “100 units” of a team members resource were being spent on a task, they would be working on this task for two hours that day. If a task were to stretch over the span of four days, the same team member would be spending 2 hours every day causing a cumulative eight hours worked on the task. Team members were assigned to tasks and the resource chart was used to ensure no member was being overworked on any given day.

Once the length of tasks has been created and resources assigned, dependencies were set based on the order that tasks had to be completed. If it was observed that a given weeks work took longer than a week due to the dependencies, member resources were restructured to expediate the critical path and the task was shortened. This process was followed until all weeks had been roughly outlined.

3. Issues encountered in the progress of the project

The main issue that the team encountered during the design process varies with specific design phases. During the first two weeks, the main issue that the team have was regard to the prioritization of research and development. That is, the research took a back seat to development. The majority of the team effort was put into preliminary hardware proposals and software designs based on initial project specifications. Therefore, the critical path for the first two weeks was governed by discussing and analyzing all design possibilities during weekly team meetings.

During the third and fourth week of design process, the critical path was composed of developing hardware prototypes and software architecture, and unit testing. The main issue the team were having during this phase was regard to documentation. Due to lack of clear consensus on how the documentation should be approached, the team spent certain amount of time on unnecessary correctional work. This lags the project progress. That issue was solved by setting clear goals for each document and solidifying the structures.

During the last two weeks of design process, the critical path was composed of finalizing design solution and integration test for beta demo and final competition respectively. The team had three initial hardware prototypes: Arm and Basket Mechanism, Claw Mechanism, Long Arm Mechanism, but that integration tests proven those were all flawed. Those designs were proven to be either demanding or non-stable for the set of hardware blocks available to team. New designs were being proposed and improved based on test results. The main issue that the team faced in this phase was regard to testing. Initially tests were being performed based on multiple changed variables, then the team decided to change the plan by testing change of only one variable to get more reliable result.

4. The budget

Having a fixed budget forced us to manage our time more efficiently in order to not waste any of it. This put pressure on all of us and especially on our program manager that kept track of the budget through the project development, leading us to plan in advance in order to organize our tasks efficiently. Another consequence of this was the dynamic allocation of our tasks at hand. If an important task went over budget, we would have to reallocate our individual schedule for the next days to get pack on the right track.

Initial planning lead us to divide the tasks at hand between us, thus we each choose a role that matched our expertise and stuck mostly with them through the project. This helped us plan our initial and future resource allocation. Our respective roles affected the timeline of the project and allowed us to finish our respective task faster.

The team was successfully allocated all project tasks as organized in Gantt chart. But if there were no constraints on budget, the budget would be allocated to extensive testing and detailed documentation. In an ideal project process, it would be optimal if most of all the possible situations would be able to be tested to reveal potential bugs and optimize the solution. And testing documentation would record all the details associated with each test, and provide reference for future development phase.

If more budgetary constraints were imposed, one solution would be spending less time on optimization, more on functionality. That is, most of the available budget would be allocated to investigate what the robot is capable of instead of improving precision on performed tasks. That would introduce some negative impacts on robot’s performance, but optimization can be prioritized as least, if time constraints.

5. How the process contributed to the success (or failure) of the project

The process was useful in achieving the goals and contributed to the success of project. The process was organized and provided step-by-step set of project milestones that helped the team to continue make progress and keep everything on track. It also helped the team to break the project into series of tasks that were being organized by Gantt chart. Although the process greatly contributed to the success of project, there were certain aspects of the process that could be optimized to increase the possibility of success. One improvement would be extensive research on expected content and standard of detail on documentation. That would save time on unnecessary weekly correction. Another aspect would be extensive testing. That is, designing smaller scale tests for only one change of variables would help producing more reliable result. Also, repeating tests with as many different situations as possible would increase the chances of detecting potential and runtime bugs, and thus increase the overall performance of the product. Detailed testing documentation would also help to make improvements and optimize performance.

The hardest part of the process was related to software for multiple reasons. First of all, an optimal algorithm to traverse a tunnel is hard to implement because the objective of the design is to achieve accurate tunnel traversals no matter what is the size and orientation of the tunnel. Another aspect would be in color detection. Because the design would need to filter out the false detections, like the color of the tree and error generated by the color sensor.

Each week 8 to 12 hours was devoted to either hardware testing, software testing or, for the final 2 and a half weeks, full prototype(integration) testing. A fair amount of testing was included under the umbrella of software development as the robot was tested between incremental improvements of the code. This testing took more of a trial and error approach and was run with the primary goal of producing a software product, so it was considered as such in the managemental tasks.

We tested at the subcomponent level whenever changes were made to either the hardware or software that would produce different results.

Sufficient tests have been designed. The tests that we designed were testing for not only normal cases but also for a lot of edge cases that were not in the final presentation. What we did not properly account for was another robot on the field. This caused us to not localize properly in the first run of the final presentation.

We estimated that integration testing would take around 2 weeks. However, we actually spent two and a half weeks on integration testing. This was because we were done with unit testing and decided that our hardware and software were sufficiently successful to continue to integration testing half a week before scheduled.
To make testing more effective, we would spend more time on the integration testing process. The small errors we found during the subcomponent level testing added up in the integration testing. These issues were only realized in the integration testing process and therefore needed more time to fix.

During the beta demo, the robot received wrong coordinates for the corners for the first time, so it ended up with a failure after localization. But the second run with correct coordinates was a successful beta demo. The software developed for beta demo was being thoroughly tested and validated. Therefore, the software developed for the beta demo was proven to be robust and was used as a base for final competition. And the result of beta demo was helpful in optimization of the design.

6. The success Design (Robot) in meeting the original specifications and the performance requirements

We were hoping that the robot would perform better because in all the tests that we ran we had a high success rate. However, the robot did not perform as expected overall. The robot performed as expected in two of the runs (run 2 and 4). In run 2, it did the full run, it detected all the rings accessible in the tree and attempted to grab them (it wasn't able to grab one of them even though it detected it which a common problem we had). In run 4, we navigated to the ring set, but we grabbed only the rings that we didn't detect and weren't able to grab the one detected (orange one), which is again the same problem we had in run 2 (grabbing small rings on top is a problem that was mentioned in the testing document).

For the other two runs, our robot didn't perform as intended at all. In run 1, we weren't able to perform a proper ultrasonic localization. In run 3, the robot failed while performing light localization (the explanation for the problem is provided in the next paragraph).

For the first run, our robot failed to perform ultrasonic localization because it detected a false positive which maybe be due to interference with the sensor from the other team. This problem is mainly due to the fact that we do not deal with false position; as you can see in figure 4 of the software document (in section 6)) we base the falling edge detection on one positive reading only). This false detection could have been avoided by adding a filter and basing our falling edge detection on more readings. Another thing that caused the robot to fail is that after ultrasonic localization the robot was off by more than 45 degrees, when it tried to perform the first line detection, one sensor detected the line but the robot was turning by more than 45 degrees without detecting a line so it thought that it missed the line and tried to back off and detect it again (please see figure 3 in the software document for more explanation of the line detection algorithm).
For the third, when the robot tried to perform the sensor line detection, on sensor missed a line but it turned by less than 45 degrees before detecting the line from the other axis. This caused the robot to be off an angle of 45 degrees approximately. This could have been avoided if the line detection algorithm assume that a line has been missed at angle of less than 45 degrees.

7. Conclusions

First of all, we learned the importance of doing extensive, detailed and complete research before starting development. On more than one occasion the team thought of a new idea and every member spent valuable time valuable time pursing the concept only to realize that the concept was flawed from the start. For example, we created a working prototype only to realize positioning the arm we built in the ring was next to impossible, something that could have been avoided with only a few more minutes of collaborative reflection. This caused unnecessary backtracking and correctional work that ate away at our budget.

Secondly, we learned the importance of being flexible team members. Many times, members would have to work on sections of the project they were not familiar with and have to meet deadlines or milestones. Without team member flexibility, it would have been impossible to complete the project under budget due to the sheer size of software and documentation that the leads had to handle.

Finally, we learned to use our own resources and logic to determine the proper course of action for completing tasks when explicit direction isn't given. On a project of such size it is impossible to have explicit direction every step of the way, so it is essential that we choose our own approach based on what we see fit and follow through. Otherwise, waiting for direction causes unnecessary confusion and wasting of time.

Each team member has their own opinions and perspectives on how the project should be approached. Similar to the Mongolian execution method; if we weren't following such a clear design process, every team member would begin pulling in their own direction and tear the project to pieces. The only way to effectively achieve all team members work effectively together and not against each other is to clearly outline a controlled and detailed design process to be followed closely. By understanding our design process, every member knows what general phase we are working on and can choose which specific task they think will best achieve the tams goals.

ECSE 456, ECSE 457 and ECSE 321 are the most notable approaching courses that our newfound knowledge will be applicable to. Both of these courses require a long-term project with a team of like-minded students. Even though the technology used may differ greatly, all of our time management and collaboration skills will be directly applicable. Our familiarity with meeting weekly deadlines and milestones will also be of great value. Additionally, the mistakes we made along the way during the development of this project and the lessons they taught us would not go to waste as the development processes in our future project courses and careers are guaranteed to bare a resemblance.

We were extremely satisfied with the performance of the robot but as there is always room for improvement, we would change a few things to improve our output efficiency. First of all, we would have spent more time researching what was expected of us and understanding every detail of the project before ever touching Lego or coding. Ensuring that the idea we have come up with is 100% viable on all fronts before ever beginning to develop is essential to success and staying within budget.

Secondly, we would have visited the professor's office hours more frequently to reduce the ambiguity at certain points of the project that caused us unneeded stress. We often chose an approach that we saw fit but began second guessing ourselves after seeing speaking with other groups. Meeting with the professors would have given us more confidence in our decisions and allow us to use the energy we wasted stressing on improving our final documentation.

Finally, we should have coordinated our initial testing better within team members and had more present at the same time to ensure everyone was on the same page. All members worked hard on creating a working prototype but often we found that the needs of the software team weren't being properly communicated to the hardware and vice versa and changes weren't being implemented as efficiently as they could have been. By having more team members working in unison during the critical early stages our working robot would have come to fruition much quicker and with much less of a headache.