Lev Tours

Dr. Michael Leverington - Sponsor
David Failing - Mentor

Members
Erik Clark
Kyle Savery
Alexis Smith
David Robb
Ariana Clark-Futrell

Requirements Specification Document

11/24/2020

Version 1.2

Overview

This document lays out the requirements, risks, and timeline for "Thirty Gallon Robot Part III: The Smiling Tour Guide". The requirements are split into 2 categories, functional and non-functional.

Accepted as Baseline Requirements for the Project:

Project Sponsor

Erik Clark

Team Lead

1. Introduction

In the field of computer science, robotics is capable of increasingly complex tasks, and the necessary hardware for these tasks has become far more accessible in recent years. As a result, the costs associated with these materials have also decreased, allowing for a greater number of individuals and organizations to take part in the research and development of mobile robotic platforms. Most significantly, educational organizations can give students the opportunity to interact with robotics either from a mechanical or programmable standpoint. To take full advantage of mobile robotics, one of the main goals of our project, “Thirty Gallon Robot Part III, The Smiling Tour Guide”, designed by our sponsor Dr. Michael Leverington, is to demonstrate that an institution's budget for robotics can be further reduced, and used more effectively, by utilizing inexpensive materials. For Dr. Leverington’s robot, one of the most inexpensive components is the thirty gallon barrel that all of the other components are built around. When this project is complete, the robot will be an example of how instructors can get students involved in the world of robotics, which could lead to an expansion of the field.

Beyond computer science the industry of robotics is involved with numerous fields such as mechanical and electrical engineering, medicine, agriculture, and manufacturing. In 2019, the global robotics industry was valued at $62.45 billion [1]. A subset of this industry that is of particular relevance to our project is known as mobile robotics. This field is responsible for creating robots that can move in 3 dimensional space without the need of human assistance. The market size for mobile robotics was approximately $9.34 billion in 2018 [2].

The Thirty Gallon Robot project is in its third year of development. The first team created the foundation for Robot Assisted Tours (R.A.T) by programming the robot to respond to commands from an Xbox controller. The second team built upon R.A.T by creating a way for the robot to generate its own maps by navigating around the building. A stretch goal for the second team was to develop a framework for Wi-Fi localization capable of directing the robot in a building, but due to the 2020 COVID-19 pandemic they were unable to test their design within the engineering building and so that development was not completed at the time. Therefore, this task now falls upon our team. Wi-Fi localization will allow for the robot to navigate on its own throughout a building without the requirement of the Xbox controller. Requiring user control defeats the robot’s purpose of being an automated tour guide.

To complete this project’s third stage of development, our team consists of 5 members, a mentor and a sponsor:

- Dr. Michael Leverington, Team Sponsor
- David Failing, Team Mentor

- Erik Clark, Team Lead
- Ariana Clark-Futrell, Team Communications/Recorder
- David Robb, Team Release Manager
- Alexis Smith, Team Coder
- Kyle Savery, Team Architect

2. Problem Statement

Dr. Leverington is a lecturer at Northern Arizona University (NAU), and cares deeply about giving students the best education possible. Part of this goal is to open new doors for students to follow their interests or expose them to promising new fields they may know little about. Dr. Leverington's robot is made to show students the possibilities of mobile robotics and demonstrate to other instructors that it can be done affordably. In addition to being low cost the robot aims to act as a tour guide for people exploring buildings. Those buildings will be located on NAU campus, starting with the engineering building and then expanding to other buildings in the future.

Currently, our sponsor's robot is not able to navigate buildings without human assistance, which is a critical aspect of an autonomous tour guide. Dr. Leverington did not want our team to use navigational tools such as cameras or GPS for his design and instructed us to only use nearby routers for navigation. The need for router based localization came from the fact that the robot is able to create its own building map, but it is unable to locate itself on the map.

To solve this problem there are a few key points that our software must cover:

- The software must be modular and independent of the robot.
- The software must only rely on the signal strength of nearby routers.
- The product must include a Graphical User Interface (GUI) to allow the users to request destinations within the building.

3. Solution Vision

In order to fulfil Dr. Leverington’s project vision, we must create a software product that will tell the robot where it is and where it needs to go in a building. This software takes R.A.T. a necessary step closer to becoming a fully autonomous tour guide robot.

To solve the previously mentioned problem points, our solution must incorporate the following:

- A system that can be run and tested independently of the robot on a laptop.
- A system that takes in Wi-Fi signals from available routers.
- A system that can navigate between any two accessible points in the building.
- A GUI to display the status of the map and accept user requests.

Partly due to COVID-19, our team will develop the location software module independently of the robot in order to support modularity, and allow for efficient testing that does not rely on the physical presence of the robot. The software will be built on a laptop at first to demonstrate its functionality. If there is sufficient time remaining after we complete this task then we may be able to move on to a stretch goal where we will port the software to a smartphone or tablet.

An important piece of our software will use routers and their respective signal qualities to calculate the direction the device should move to reach its destination. It is important that when this data is retrieved we base our calculations on average signal strength along with tolerances since the incoming data will not always have the same strength even when retrieved from the same location. Using these stable data points we will create a navigational system that works within the building.

Finally, a GUI must be created that can display all useful information. Useful information in this context would be the building map, device's location and possible destination points with respect to the map.

4. Project Requirements

The following sections will cover our project's functional and non-functional requirements in detail. Laying out these requirements is a crucial step to creating a strong foundation from which we can develop our solution. Since the beginning of this project, our team has been meeting with Dr. Leverington to precisely determine what he expects our system to accomplish which guides us in being as specific as possible for our requirements.

4.1 Functional Requirements

The functional requirements of a project are related to fundamental behaviors of the system. These behaviors define what a system does or does not do. Our two main functional requirements are that our software must navigate itself through a building and communicate with a user via a GUI.

The navigation component of our product will be focused on three aspects of movement through 3-dimensional space, data capture, localization, and building traversal. Furthermore, obtaining data from nearby routers, calculating position, and then determining which path will lead to the destination each have their own functional requirements which are broken down as follows.

- Data Capture
    - Get Wi-Fi Signals
        - Our software must read in sufficient amounts of relevant network data and subsequently parse through that information before it can begin calculating its position. This stage will serve as the setup or training for the software since each building's routers will have different signals as well as locations. On setup, the software will be taken throughout a building to each destination point and then read in samples of the signal data to learn what nearby routers' signals should be when it returns to this position. After the training phase, the software will not need to capture network data as frequently and only when it has to update its position.

- Storage of Past Data Points
  - When a building's router signals are captured through Relative Signal Strength Indicator (RSSI), our software must also store these values in an accessible data structure which will be used on subsequent calculations.

- Get Router Addresses
  - Part of parsing through our software's acquired signal data is ensuring that each router has its own identification to be distinguished from adjacent devices. The Python library, RSSI, attaches each router's Media Access Control (MAC) address, which is a unique identifier assigned to network interfaces, to its signal output. As such these addresses, in addition to signal quality, will be used to determine the software's location.

- Get Signal Quality
  - Since RSSI already brings in the data our software requires, another important aspect of capturing signal quality is that we must determine which signals to use. Some signals may be too unstable, either due to physical distance between our software and the device or possibly the device itself, to be used in consistent calculations. Hence, our software will ignore these signals and/or put tolerances on them to negate any minor fluctuations.

- Localization
  - Comparison of Current Router Signals
    - After reliable router signals have been acquired and stored, the process for calculating position will involve comparing current captures of signal data to these stored values. It is important this process is efficient since our software must be fast enough to match a human's walking speed, which will be discussed further in non-functional requirements.

- Building Traversal
  - Grid System
    - To prevent our software from considering physical obstacles as valid paths, it will need to implement a grid as its view of the building layout. This grid will overlay onto the map of the building and its function will be to keep track of the signal ranges that our software must be within to not hit an object. The grid 'lines' will be centered along each hallway the robot is allowed to traverse. This grid will also reflect distances between points, or nodes, in the building which will be recorded during the training phase.

  - Point to Point Navigation
    - Once our software is able to know what destination point it is at, it needs to be able to calculate the necessary movements to reach any other destination point. Since the previously mentioned grid system includes

nodes throughout the building, their signal qualities, and respective distances our software must calculate the most efficient path between these nodes.

With the fulfilment of our project's navigation component, the next functional requirement concerns our software's GUI. The purpose of this interface is to streamline the software's interactions with both user's simply taking advantage of the tour capability or user's performing setup operations in a new building. There are two main components of our planned GUI, image manipulation, and the ability to communicate with a user which are outlined below.

- Image Manipulation
    - Accept an Image File
        - The robot is already able to generate its own maps, but lacks the ability to make calculations based on this and router signals. Therefore, our software must be able to accept this image file of the building map to allow for manipulation.

    - Edit Image File
        - Our product must overlay a grid onto the generated map. Once this is completed the software will be able to translate user requested destinations based on the grid's nodes to actual locations on the map. This image of the requested destination can then be displayed to the user. Additionally, this GUI must scale the map's image depending on the level of zoom the user decides to view the building layout.

- Communicate with the User (Backend and Frontend)
    - Backend communication
        - Backend communication is primarily related with the software's setup in a new area. Interacting with the GUI, the backend user would input all destination points and turns within a building as the grid's nodes. Then the distances between each node and its closest neighbors would also be added. In this case, closest would define the nodes that the software can reach without having to make any turns. The backend user can flag specific destination points as having significant value and to be included in a list of tour spots.

    - Frontend communication
        - Frontend users, or the ones who have no knowledge of the inner workings of our software and simply want to use the service, must be able to request destinations within the building. These destinations can be either a single destination the user wants to be directed to or the request can be made for the software to take the user to all points within the list of tour spots.

4.2 Non-Functional Requirements

The non-functional requirements of a project are requirements that specify a tangible set of criteria which performance can be objectively tested against. Similarly to functional requirements our software's non-functional requirements can be split up into two categories, navigation and the GUI.

- Navigation
  - Speed
    - An important aspect of Dr. Leverington's robot is that it is able to move quickly enough to match the average walking speed of a human. Clearly, the robot would not be an effective tour guide if it moved too quickly or slowly. In relation to our software, this means it must calculate its position and where it must move fast enough so that the device the software is running on can be moved between 3 and 4 mph and still produce accurate data [3].
  - Definition of Success
    - When a destination point is requested the software will attempt to navigate the building and reach this point. We consider it a success if the software can get within 2 meters of the target. For example, if the user requests to be taken to room 102, then this task will be a success if the software is no more than 2 meters from the doorway after it traverses the building.
  - Success Rate
    - Furthermore, as long as the software can achieve a success in at least 95% of trial runs the overall product's navigation component will be considered a success.
  - Shortest Path Between Points
    - We will likely use a simple process such as Dijkstra's Shortest Path First algorithm, which as the name implies finds the shortest path between two nodes in a graph. In this context, the 'graph' is our grid.

- GUI
  - When a user is interacting with the software via the GUI, we require that a destination request can be made within 2 clicks and any necessary typing to input the destination's title. Specifically, the user should be able to click on the

search bar, input their desired destination, and then click enter for the software to begin directing them around the building.

5. Potential Risks

As with any navigation system the project has some inherent risks to it. The main risk we have is that our software may guide the robot off course and have it run it to objects such as walls, ledges, doors, or people. In terms of substantial damages the system could cause to itself or the object it encounters, the risk is somewhat low. This is due to the fact the system will only be moving between 3 and 4 mph at any given time. Still, the one instance where the robot running off course could prove truly dangerous is if it directs itself into a stairwell, at which point it would easily cause damage to its hardware and any individual it may contact. However, in terms of effective functionality this is our highest risk since an autonomous tour guide can not have its followers constantly having to pick it up and place it on the correct path. Another consequence of running off course is that the individual it was guiding may now also be lost in the building, which further undermines the tour functionality.

The building the software is operating in may occasionally have temporary hazards blocking various hallways. This risk is quite low because while these kinds of hazards may appear often (i.e. wet floors signs), the remedy is quite simple. Anytime a new hazard of this type pops up, a backend user could temporarily update the robot's map to consider this hazard as a wall and calculate alternate routes. Since our software solely relies on Wi-Fi signals and user inputs there is no other possible remedy as our system is not allowed to incorporate cameras to actively avoid hazards.

Another moderate risk we must take into account is how our software will function if a router changes position, is removed entirely, or whenever all Wi-Fi signals in the building go down. For the latter, there is nothing our software can do to continue operating since its entire system is based on functional Wi-Fi signals. For routers that are removed the software's signal data storage would be updated to reflect the absence of the router. Lastly, if a router happens to be shifted in a building (which should rarely happen) then the backend user will need to take the software through the training phase again to maintain accurate point to point traversal.

6. Project Plan

Once our functional and non-functional requirements were laid out we were able to generate a definitive and ordered plan for the remainder of this project to ensure we have ample time to meet all requirement criteria (see Figure 6.1 for a Gantt chart).

1. We must scan and analyze Wi-Fi data from several points in the engineering building at NAU as a first step toward our initial prototype.
2. Once we are able to capture these Wi-Fi signals we must begin capturing numerous sets of data to get an accurate picture of how the Wi-Fi signals operate.
3. The initial prototype will be constructed which uses the stored data on Wi-Fi signals to determine if it is at one of two points in the building.

4. After the initial iteration of our design is functional, we will continue collecting data on the first floor so our software can determine if it is at one of any possible points.
5. At this point, we would begin creating our GUI and overlaying a grid onto the building's map.
6. Once the grid overlay is completed, our team is ready to begin solving the problem of navigation.
7. The first solution for navigation we plan to derive is calculating the shortest path between two points.
8. After our software is operational on the first floor of the engineering building, we would then expand its functionality to include all of the floors.
9. Finally, we would take our software to a new and untested building to prove the validity of our design.

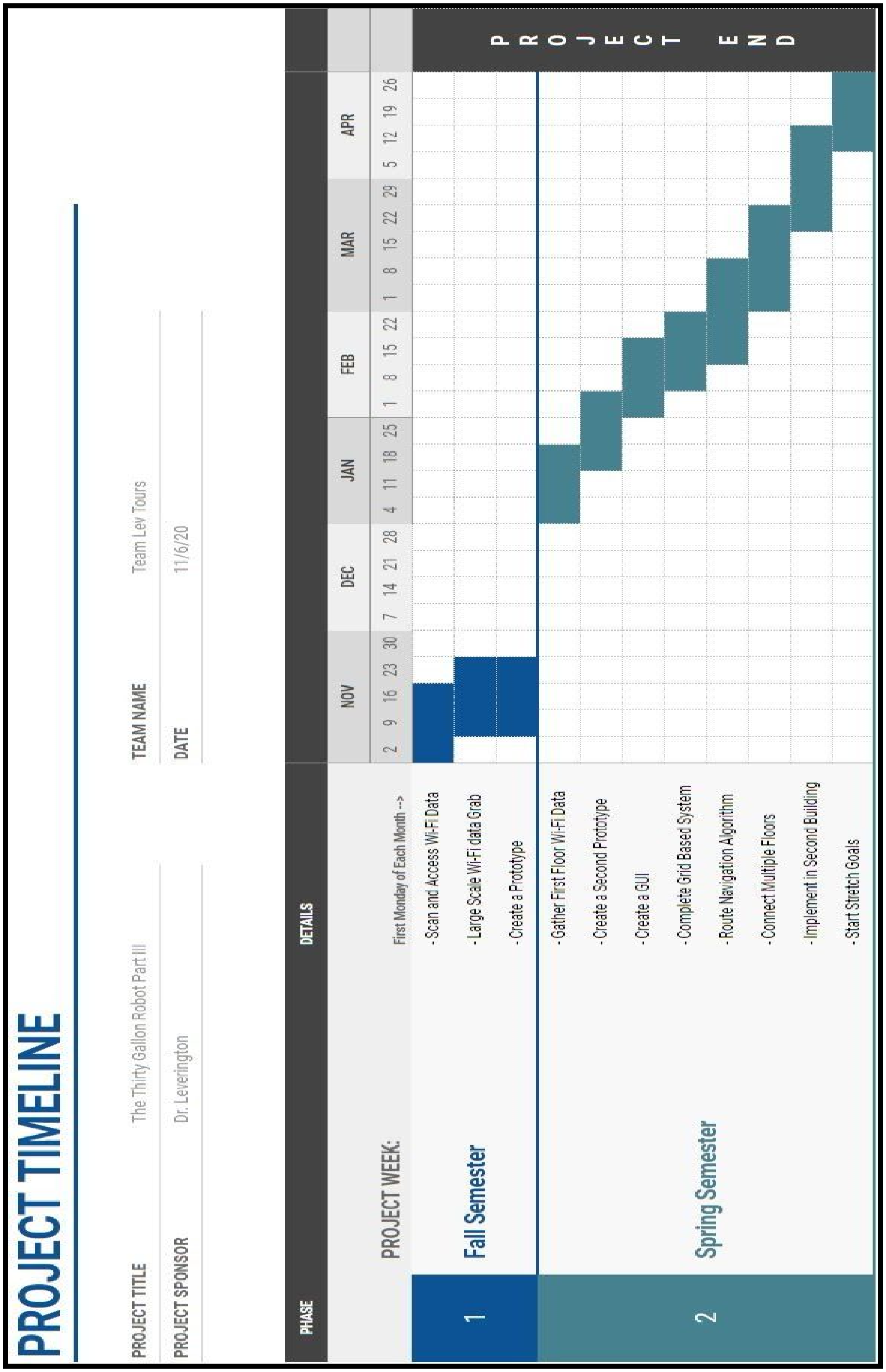

Figure 6.1

A Gantt chart depicting our plan for the remaining project duration.

7. Conclusion

The capabilities of mobile robotics continue to expand and the societal problems that can be solved by them are expectedly growing. As a result, mobile robotics is an increasingly valuable field and the more people who can get involved early on expands the subject area both further and faster. Our project, “The Thirty Gallon Robot”, seeks to demonstrate affordable ways that educational institutions can get students involved with robotics. By contributing our software to a robot that is as inexpensive as possible, our team (and previous teams) believe that we can show other instructors that getting their students involved with robotics is not only possible, but truly rewarding.

The specific problem our team is working to solve is designing, developing and delivering a functioning software product that utilizes the signal strengths of nearby routers to navigate a building. This problem exists as our team sponsor, Dr. Leverington, has a robot in its third year of development that does not currently have a way to autonomously navigate the engineering building at Northern Arizona University. Our solution to this problem involves fulfilling several functional and non-functional requirements that were specified through discussions with Dr. Leverington.

The functional requirements of our software product begins with the ability to capture information regarding Wi-Fi. This information must include the current Wi-Fi signal strength from nearby routers which will be used to determine the software’s location (localization) by comparing current signal strengths with previously captured samples which represent points throughout the building. Once the software is able to distinguish itself between any two listed points in the building, it must be able to calculate which direction to move in order to reach a point from any other point. After the software is capable of traversing the building, it is important that relevant information is displayed to the user. This relevant information will be the software’s current location and possible destination points which will be communicated to the user via a map shown on a Graphical User Interface (GUI). This GUI must be able to then accept user inputs from either backend or frontend users. The respective inputs expected from these users would be commands to set up the robot in a new building or requests for the software to direct them to a new location.

The non-functional requirements for our software are centered around how quantitatively efficient our software is expected to be. The touring robot is expected to move at the average walking speed of a human, so our software must produce expected output even when moving around a building at 3 - 4 mph. The expected output will be rated on the fulfilment of a destination request. A request is defined to be fulfilled if the software can guide the user to within 2 meters of the requested destination. Furthermore, this operation must be successful for at least 95% of requests. Our software must also calculate the shortest path between any two points in the building to be as fast as possible to not waste the user’s time. In relation to our software’s GUI, we require that a user must be able to input a destination request within 2 clicks and any accompanying typing to spell out the destination’s title.

When developing our solution we must also take into account potential risks. For our project, there are a few issues that must be taken into account.

- The software guiding the robot off course which could result in damage to itself and other objects it comes into contact with.
- The software being unaware of temporary hazards within the building.
- The software's functionality being reduced due to router outages.

Laying out definitive requirements is absolutely critical to the timely success of any software project. After constructing this document, our team has a well defined schedule for the duration of this project and every member knows exactly when tasks must be completed. As a result we are confident we can produce a software product to the satisfaction of our sponsor, Dr. Leverington.

8. REFERENCES

[1] Pramod, B., & Shadaab, K. (2020, October). Robotics Technology Market Size, Share and Analysis: Forecast - 2027. Retrieved November 04, 2020, from https://www.alliedmarketresearch.com/robotics-technology-market

[2] Rahul, K. (2019, January). Mobile Robotics Market Size, Industry Analysis and Applications by 2026. Retrieved November 04, 2020, from https://www.alliedmarketresearch.com/mobile-robotics-market

[3] Cronkleton, Emily (2019, March). What Is the Average Walking Speed of an Adult? Retrieved November 06, 2020, from https://www.healthline.com/health/exercise-fitness/average-walking-speed

9. Original Signed Cover Sheet

Lev Tours

Dr. Michael Leverington - Sponsor
David Failing - Mentor

Members
Erik Clark
Kyle Savery
Alexis Smith
David Robb
Ariana Clark-Futrell

Requirements Specification Document

11/23/2020

Version 1.2

Overview

This document lays out the requirements, risks, and timeline for "Thirty Gallon Robot Part III: The Smiling Tour Guide". The requirements are split into 2 categories, functional and non-functional.

Accepted as Baseline Requirements for the Project:

Project Sponsor

Erik Clark
Team Lead