Technology Feasibility Report

11/7/2018

Keystone Robotics
Robot Assisted Tours

Project Sponsor:
Michael Leverington

Faculty Mentors:
Austin Sanders, Jun Rao

Team Members:
Hailey Ginther, Shannon Washburn, Gabrielle Halopka, Falon Ortega

Table of Contents

1.0 - Introduction

1.1 - Robot Assisted Tours at Northern Arizona University

Northern Arizona University's Engineering building is the most important stop of campus tours for future engineering students. The labs, project rooms, and lecture halls of the building are where they will be studying for the next 4+ years. This means the impression touring groups get of the facilities during their time on the tour is an essential contributing factor to attracting and retaining new students. Attracting the attention of these students with physical evidence of the work accomplished by seniors of the department will help convince these new individuals to enroll.

The goal of this project is to create a robot capable of autonomously giving tours of the engineering building, fulfilling this need for a captivating introduction to the projects NAU students can accomplish thanks to their coursework.

1.2 - Client's Vision

Our client, Dr. Michael Leverington, is a professor at Northern Arizona University's School of Informatics and Computing. The professor has a Ph.D. in Education, Masters of Computer Science and Psychology, a Bachelors in Physics. Additionally, he is an avid robotics fan eager to bring the multidisciplinary topic to his department. He has tasked our team with the planning, assembly, and programming of a robot capable of giving tours of NAU's engineering building with some level of autonomy. Our client sees the robot as a solution to two main problems- the first being the need for the automation of tours that will simultaneously free up faculty's time and impress visitors, and the second being the need for a robotic framework that future student teams could use as a foundation for other projects.

This robot is intended to be:

- 100 pounds at minimum
- 3-4.5 feet in height
- 2 feet in width
- Able to run on a single charge for 4-6 hours
- Capable of autonomous movement
- Able to give automated tours of a set building

1.3 - The Problem

Our group is expected to take the initial steps of planning and building this robot. With less than a year for this project, we do not have the time to implement the full capabilities a tour-giving robot would need to work competently. Therefore, our group has worked with our client to identify the key features he most wanted us, as the first of several planned teams, to complete. These challenges are:

- Planning and implementing a navigation system capable of basic obstacle avoidance
- Planning and implementing safety features that ensure the robot and those around it remain unharmed
- Determining most suitable hardware, with emphasis on how parts integrate with one another to determine the most efficient and modular build
- Creating a form of manual control to move the robot once constructed, for testing and early navigation purposes

1.4 - Solution Approach

Our solution is to choose and then assemble the hardware components required to meet the specifications of our robot by our client. We will simultaneously be researching software frameworks our robot will use to interpret sensor data. Specifically we will be making decisions on:

- Microcontrollers
- Robotic Navigation Frameworks
- Sensors
- Component Housing
- Bases
- Batteries
- Motors & Wheels
- Motor Controllers
- Manual Controls

1.5 - The Team

Keystone Robotics is a multidisciplinary team comprised of two electrical engineers, Gabrielle Halopka and Falon Ortega and two computer science students, Hailey Ginther and Shannon Washburn. Our team is mentored primarily under Austin Sanders with the assistance of Jun Rao.

1.6 - Document Summary

This document is a formal representation of the process we have undergone in choosing the ideal hardware and software needed to solve our problem. We begin the analyzation of our requirements in section 2 with an overview of the technical challenges we expect to face during our research and development. Each challenge in section 2 is followed by a brief summary of the hardware and software we identified as necessary for solving the problem. In section 3 we cover the specifics of each component of the required hardware and software identified in section 2. For each, we analyze our options carefully before presenting our final choice along with our reasoning. Section 4 presents our plan for the integration of the discrete parts chosen in section 3. Finally, we restate and confirm our stated plans in section 5, our conclusion.

2.0 - Technological Challenges & Solution Overview

2.1 Introduction

Keystone Robotics has broken down the challenges of the Robot Assisted Tours project into 4 discrete parts - Navigation, Safety & Obstacle Avoidance, Runtime & Efficiency, and User Interface. For each of these four challenges, we give a brief description of what the problem entails. Then we follow up these descriptions by addressing the generalized types of hardware and software that we will utilize to solve these problems. An in-depth analysis of each component mentioned in this section (2) can be found in the next section (3).

2.2 Navigation

2.2.1 Introduction

In order to give tours of the building, the robot must be able to navigate. The criteria for the navigation aspect is that the robot must be able to move autonomously and accurately give tours of the building. For this project, autonomy is defined as the robots ability to move around independently without human intervention. Accuracy for our device would be defined as the ability to move with an exactness fairly close to the instructions given to it (Specifically, within ~2-4" of a desired path, with a speed of approximately 55 inches/second, and with arrival at the correct destination).

2.2.2 Hardware Solutions

The hardware item that would satisfy the autonomy component of the navigation challenge would be the microcontroller(s). A microcontroller would allow the robot to be pre-programmed in a way to allow the robot to move on its own. Two board brands that we are considering are the Arduino Uno and the Raspberry Pi. The approach for this project was to use one microcontroller such as the Raspberry Pi as a 'brain' for the robot that would delegate to a more basic microcontroller like the Arduino to control the motors. The supported programming language of Arduino models are not object-oriented while the Pi models are, leading us to the consideration of combination of at least 2 microcontrollers. To see more in depth analysis of which microcontrollers we chose and why see section 3.1.4.

2.2.3 Software Solutions

The software solution would be to either use a pre-existing robotic navigation framework or to implement navigation and mapping algorithms from scratch. Both solutions would require some sort of sensor data such as velocity and rotation to determine the robot's location in respect to its starting position. Frameworks such as Robot Operating System (ROS) and the Mobile Robot Programming Toolkit (MRPT) provide implementations of these algorithms and tools to collect sensor data which can be used with our robot.

2.3 Safety and Obstacle Avoidance

2.3.1 Introduction

In terms of obstacle avoidance, the hardware solutions are composed of the sensors we have chosen to fulfill this requirement. The robot needs to be able to sense when it is going to come in contact with a person or object as well as be able to sense when it is coming close to a drop-off like a staircase to avoid falling.

2.3.2 Hardware Solutions

The criteria for choosing sensors for obstacle avoidance comes down to usability and cost. The sensors we considered for this project include the Microsoft Kinect, laser scanners, infrared sensors, and sonar. Both the Microsoft Kinect and laser scanners can be used for both mapping and obstacle avoidance as they can create a 3D map of the area of navigation and sense people and objects using depth data. Infrared sensors and sonar both measure distance and can be used to determine whether or not the robot is near a drop-off and would work well for sensing drop-offs.

2.4 Runtime and Efficiency

2.4.1 Introduction

The client requires the robot to be able run on batteries that can sustain the robot for eight hours. There is no time limit for recharge time. The battery must able to power all needed components for at least 4-6 hours.

2.4.2 Hardware Solutions

There are two components for the runtime and efficiency: the batteries themselves, and the components that would use the batteries. For the batteries we've considered Lithium Ion batteries instead of other heavier options, such as Lithium Polymer batteries. We also have to consider rechargeable smaller batteries, such as double AA's vs. power packs used to charge mobile USB devices that we can use to charge the Arduino and the Raspberry Pi. For the wheels and motors, the size and shape of wheels also affect the draw on power. If the wheels are too wide or with tread, they will have higher traction and will draw more power. For our solution, the battery or batteries must be able to handle the power draw from the components. The wheels should be at most 2-3” in width in order to draw less power.

2.5 User Interface

2.5.1 Introduction

For testing and maintenance on our path towards an autonomous robot, this project needs some manner of interface that allows for manual control. Furthermore, once autonomous, our robot will need some way to receive commands that will allow it to perform functions beyond obstacle avoidance, such as leading a user to a location of their choosing.

2.5.2 Hardware Solutions

The initial hardware considered for the solution was a simple touch screen with a GUI that has a select number of commands for the robot. Another solution considered was using the Microsoft Kinect for gesture commands. In the end, we decided additional hardware and programming solutions such as these are better suited as stretch goals or future projects for other teams. The most suitable solution for our project would be to use a remote desktop connection to access our microcontroller via a laptop computer. This approach is simpler and low cost, not requiring the purchase of any dedicated components beyond a computer with a wireless internet connection.

2.5.3 Software Solutions

The software needed for this solution are the IDEs and operating systems our chosen microcontrollers will be operating on.

3.0 - Technology Analysis

3.1 - Introduction

This section provides an in-depth analysis for each component used in the building of the robot. We have sorted these components into 1 of 4 technological challenges identified in the previous section. Sorting is based on their relevance to the given problem's solution. Components that are used in the solution for multiple challenges are placed in the first section they apply towards.

For each piece of technology identified in this section we introduce the component by explaining the function it is intended to serve and identifying parameters that options must meet. Then we present our options (following identified parameters) and identify pros and cons of each choice, including a table of summarized options if more than 2 alternatives are considered. After that we provide the step(s) to be taken that can be used to prove the feasibility of a chosen component. Once our options are established and analyzed we state the approach we decided on. Finally, we conclude section with reasoning on the final choice

3.2 - Navigation Components

<u>3.2.1 - Microcontrollers</u>

- Introduction: A microcontroller is needed in order to control the functionality of the robot so that it is able to move autonomously. Size and physical space these microcontrollers take up are not a concern of the group, so we have to use other compare other statistics to make our choice. The metrics used by the team to choose between boards were:
    - Of a price less than $50
    - Supported by free/ open-source libraries & documentation
    - Allows for expansion of memory
    - Has compatibility with largest possible range of sensors under our consideration
    - Capable of communication with other types of microcontrollers
- Alternatives: The two board brands that considered were varieties of Arduino and Raspberry Pi models, which are both supported by large open-source libraries and are compatible with other microcontrollers.
    - Arduino Options:
        - Arduino Uno

- Pros: Free of cost (4+ units already in team possession) & well-suited to basic motor/wheel controls
- Cons: less than 32 KB of non-expandable memory

■ Arduino Mega

- Pros: More memory and pins than UNO model
- Cons: Pricier and still less than 1 GB non-expandable memory

○ Raspberry Pi options:

■ Raspberry Pi 3, Model B

- Pros: 1 GB memory, expandable memory via micro-SD card
- Cons: Extra parts like the SD card must be purchased separately, adding to the price

■ Raspberry Pi 3, Model B+

- Pros: Faster processor speed
- Cons: Pricier than the regular model B Pi, added features not essential for any project requirements

○ Option summary:

Criteria	Arduino Uno	Arduino Mega	Raspberry Pi 3, B	Raspberry Pi 3, B+
Price	22$	39$	35$	40$
Memory	32 KB	256 KB	1 GB	1 GB
Expandable Memory Possible?	No	No	Yes, via MicroSD	Yes, via MicroSD

Table 1: Microcontrollers Summary

- Proving Feasibility: By testing and building our microcontroller(s) in three stages we will ensure our choices are a viable solution for this project at each step in assembly. First, the microcontroller must be able to interface with the sensors of choice and report the data back into a accessible file. Next, the microcontroller must exhibit proof of cross-compatibility with any other microcontrollers chosen by successfully passing a set of test instructions back and forth between them. Finally, the microcontrollers must successfully pass instructions to external hardware i.e. a motor and perform basic back and forth movement. These tests will be performed first on a small-scale test robot in the team's possession, then repeated at each step of the project's assembly.

- Chosen Approach: Raspberry Pi model 3 B and Arduino Uno
- Conclusion: The Arduino Uno is easy to obtain as it is inexpensive and several are already at our disposal and is relatively easy and simple to use. However, the IDE that would be used with the Arduino does not have object-oriented capabilities. OOP capabilities are a necessity in the project to make the coding more simple and easy to implement. For the purposes of navigation It should also be noted that Arduinos are designed for hardware control and not processing, so they are better suited to control the movement of the wheels. When considering the Pi microcontrollers, we looked at several models. The Pis had better memory than the Arduino Uno and were capable of OOP programming. One of our considerations was to have something with internet connection in the event using Google Maps for navigation was a viable solution on the software size. For this reason, we opted to use the Raspberry Pi 3 model B.

<u>3.2.2 - Robotic Navigation Framework</u>

- Introduction: When choosing the navigation framework for our robot we looked at a few different options including Robot Operating System (ROS), Mobile Robot Programming Toolkit (MRPT), and the possibility of writing the navigation algorithms ourselves. The parameters the navigation framework need to meet for this project are:
    - Open source/free to use
    - Ability to interface with the microcontroller and sensors of choice
    - Provide flexibility for future projects
    - Provide ability for multiple process to run simultaneously and communicate
- Alternatives: We decided using a pre-existing toolkit would be best considering the complexity of data processing required for the autonomous navigation of a robot. The two we considered using for our project are MRPT and ROS. Both are robust libraries which would provide functionality to interface with our chosen sensors and microprocessors.
    - Options:
        - MRPT
            - Pros: Compatible with multiple operating systems including Windows, Linux, and Mac.

        - Cons: Not as well documented with few readily available tutorials to facilitate quick learning.
    - ROS
        - Pros: Highly documented usage on other robotic projects with detailed examples and tutorials to allow us to learn the concepts relatively quickly.
        - Cons: Only fully supported on certain Linux operating systems.
- Proving Feasibility: By installing the navigational framework and successfully interfacing with our microcontroller to control motors on a smaller test robot and grabbing data from our sensors we can be sure that it is a compatible framework for our project.
- Chosen approach: ROS
- Conclusion: We have decided to use ROS as our navigational framework due to its extensive libraries and developer tools including those that will help us send messages to the microcontroller and gather data from the sensors. Specifically we will be using the rosserial_arduino package to interface with our microcontroller, the libfreenect package to obtain Kinect sensor data, the joy package to get controller input, and the navigation stack to convert our sensor data into a map that our robot will use to navigate. As we continue our development, we will likely encounter other ROS packages which will help us reach our end goal. Due to ROS's highly documented usage in other projects of a similar nature to ours we feel that it would be quite feasible to use with our robot.

3.3 - Safety and Obstacle Avoidance Components

<u>3.3.1 - Sensors</u>

- Introduction: The sensors for our robot will be responsible for collecting data about the robot's environment which will be essential for navigation, mapping, and obstacle avoidance. Data collected via sensors can be distance, point cloud data, RGB image data etc. Sensors suitable for our needs must be:
    - Less than 25$ per unit
    - Compatible with microcontroller brands under consideration (Raspberry Pi, Arduino)
    - Simple to install on body of robot
    - Accurate in terms of location and spatial data

    - If needed, compatible with any other sensors that give supporting data
- Alternatives: Sensors under consideration need to report data that allow the robot to both orient itself in a constant environment and avoid variable obstacles. We focused on sensors capable of providing either one or both of these capabilities.
    - Options:
        - Indoor Positions System (IPS)
            - Pros: Accurate localisation data in a building
            - Cons: Expensive, requires beacon installation throughout environment
        - Laser Scanners
            - Pros: Mapping ability very accurate, as proven by use in self-driven cars
            - Cons: Expensive
        - Infrared Scanners
            - Pros: Cheap models available
            - Cons: Small in size, robot would require multiple units
        - Microsoft Kinect
            - Pros: Well-documented as sensor system for navigating robots, cheap - available used
            - Cons: Recently discontinued from manufacturer
    - Option summary:

Sensor	IPS	Laser Scanner	Infrared	Kinect
Within Price Range?	No	No	Yes	Yes
Environment data reported	Exact location in set environment	360-degree point-cloud data	Proximity, motion	Proximity, motion, Limited range point-cloud data, RGB-image data

Table 2: Sensors Summary

- Proving feasibility: First, any chosen sensor must exhibit compatibility with any microcontrollers chosen, to be tested by temporary installation on team's small-scale test robot. Then, the sensor must be able to mount securely onto an appropriate space on the body of the final robot where it can report data without hindrance.

- Chosen approach: Microsoft Kinect & Infrared sensors
- Conclusion: We have chosen to use the Microsoft Kinect sensor to get 3D map data with the chosen navigation framework to implement the autonomous navigation of our robot. This sensor would also be able to detect people and other obstacles to avoid. The Kinect sensor is highly used among robot hobbyists as well as academic robotic projects as a cheap alternative to laser scanners for providing 3D mapping data. The Kinect can be mounted on the top portion of the robot to function primarily for navigation and large obstacle avoidance.
  In addition, we decided on using infrared (IR) sensors to determine the distance our robot is from the ground in order to avoid drop-offs. IR sensors are cheap and easily obtainable and can quickly detect change in distance to determine if the robot has encountered a drop-off. These sensors can be placed in low areas of the robot outside of the Kinect's line of sight, sensing for small obstacles and drop-offs such as staircases.

3.4 - Runtime and Efficiency Components

3.4.1 - Component Housing

- Introduction: Some sort of housing is needed to cover the inner hardware components of the robot- namely the microcontrollers, batteries, and wiring. The parameters desired for this housing are:
    - Comprised of a material that can be cut/ altered with household power tools
    - Of a size height-wise in the range of 2.8' to 3.5', to be at a height where the top is roughly at the height of an average adult's waist
    - Of a size width-wise less than 2', to properly fit through doorways of 2' or larger
    - Less than 30$ in cost
- Alternatives: Our client strongly desires a 30 gallon barrel as the housing for this project, which fit all the parameters for this component and was cheap to obtain. Therefore, our options were limited to varieties of these barrels. We considered metal versus plastic barrels and new versus used barrels.
    - Options:
        - 30 Gallon Plastic
            - Pros:

        - Meets all size and price criteria
        - A donated barrel would be the least expensive route
      - Cons:
        - Plastic material may not be strong enough to handle modifications
        - The diameter is just wide enough that addition may prevent it from getting in and out
    - 32 Gallon
      - Pros:
        - Smaller diameter would make adding parts to the outside easier without having to worry if it will fit or not
      - Cons:
        - Plastic material may not be strong enough to handle modification
        - Most expensive option
    - 31 Gallon
      - Pros:
        - The metal material is more durable
      - Cons
        - It has the largest diameter and may not be able to fit any modifications
        - It may be more painful to bump in to
  - Option summary:

Criteria	30 Gallon	32 Gallon (Uline)	31 Gallon
Material	Plastic	Plastic	Metal
Size(HxD)	30’’ x19’’	27’’x 22’’	20.5’’x27’’
Price	Free (donated)	27.00	19.99

Table 3: Housings Summary

- Proving feasibility: The barrel selected must be within the size parameters of plus or minus 2’’. The barrel should also be durable enough to be modified as needed when parts are added (i.e. drilling holes and adding mounts). Finally, the barrel material must exhibit heat resistance up to the expected temperature the internal hardware is expected to reach- tested by heating samples from barrel taken from the lid.

- Chosen approach: 30 Gallon barrel - Plastic, used
  The 30 gallon barrel is 30" high and ~19" in diameter, meeting our size limitations. The plastic can be easily cut by household power tools such as saws and drills, unlike the metal varieties we considered. Finally, new barrels ranged in price from 18 to 30+ dollars, while used barrels were made available to us free of charge. In short, our choice fit the parameters we identified most suitably and was the lowest cost option as well.

3.4.2 - Base

- Introduction: A frame or base is needed to both support our housing and to give our team additional, more stable surface on which wheels and motors can be mounted. The parameters for this are:
  - The frame must be no wider than 2' so it is able to roll through doorways
  - The frame must be no taller than 6''
  - The frame must be able to hold the weight of the barrel and all other components
  - Preference for prefabricated frame
  - Team is able to make changes to the base without worry of compromising it
  - Price less than $60
- Alternatives: Due to the time and experience limitations of the team in terms of construction, we have decided our options should be limited to prefabricated options. First, we considered the material of the base - wood, plastic, or metal. We also had to consider the shape- round, triangular, trapezoidal, or rectangular.
  - Wood Frame Dolly (Solid)
    - Pros:
      - The Dolly is made of solid wood and would be very durable
      - The rectangular board would be stable
    - Cons:
      - The dolly is so simple that it would be less expensive to build one ourselves.
    - The rectangular board may be more difficult to navigate and turn corners
  - Metal Triangular Drum Dolly
    - Pros:

        - The robot would have a very good turning radius
        - small enough to be flush with the barrel
        - Can hold up to 700 lbs
    - Cons:
        - The base is very small, essentially just a metal triangle with wheels, it would be difficult to make changes to. It would be difficult to mount the motors
- Wooden Rectangular Dolly
    - Pros:
        - The wood dolly will be durable but still easy to modify
        - The center is hollow so boards can be added for support or left open to place wires and components
    - Cons:
        - The rectangular design my have issues with turning
        - The Frame may not be able to support the barrel and components properly
- Trapezoid Wood Dolly
    - Pros:
        - The trapezoidal design would allow us to have benefits of both stability and turning radius
        - The design would be made the team so it would be easy to make needed modifications or plan out where they would be
    - Cons:
        - The team has little experience with building, so construction our own may be difficult
        - The center of gravity would have to be calculated in order to keep the barrel steady. As more pieces are added, this may create problems in the future
- Option summary

Criteria	Wood Rectangle (Solid)	Metal Triangle	Wood Rectangle (Hollow)	Trapezoid Wood Dolly
Shape	Rectangular (Solid)	Triangular	Rectangular	Trapezoidal

Material	Wood	Metal	Wood	Wood
Size	30''x18''	Not listed	30''x18''	15''x12''x 20''
Price	$55	$75.55	$50	~$30

Table 4: Bases Summary

- Proving feasibility: To test if the base is stable, it should be able to carry the weight of the barrel and all components with minimal rocking or other movement. The whole robot should also be able to turn reasonably well; it should not bump into anyone who would be walking down the hall when it turns.
- Chosen Approach: Hollow Wood Frame Moving Dolly, 30" x 18"
- Conclusion: Wood was the aspect we settled on first when considering the base. It was the most easy-to-alter of the options available. The size was a fairly simple decision as well, as we were limited to something equal or larger in width than our housing that remained smaller in width than the doorframes of the building it will navigate.
  Shape was our most researched factor, one that led us to consult with professionals at AZ Power and Lawn, a local company in Flagstaff, Arizona. Discussion there led us to the decision that a four-sided, rectangular base would suit us best. Finally, we elected to purchase the "frame" type dolly rather than the solid board dolly to both cut down on weight and give us room to make easier frame adjustments should the need arise.

3.4.3 - Battery

- Introduction: A battery is needed to power the motors through a motor controller and the microcontrollers so that the robot can move autonomously and run on its own without having to be plugged into a wall. Ideally the client would either be able to recharge the batteries daily or once every few days with little effort. Rechargeable batteries are cheaper and decrease the likelihood of backwards current flow affecting the battery's performance and safety. The batteries chosen need to fit the following parameters:
  - Easily accessible/rechargeable
  - Price less than $300
  - No less than 1' wide and 1' across
  - 10-14Ah minimum for desired run time

  - Battery(ies) chosen must have at least 48V when in series or 48V alone
  - Able to have leads that easily connect to rest of circuit
- Alternatives: We’ve considered Lithium Ion batteries vs. other heavier options, such as Lithium Polymer batteries. We also have to consider rechargeable smaller batteries, such as double AA’s vs. power packs used to charge mobile USB devices that we can use to charge the Arduino and the Raspberry Pi.
  - Lithium Ion Battery
    - Pros:
      - These batteries are rechargeable, making sure that the client can often access them.
      - Lithium Ion options are cheaper than lithium polymer.
      - Lithium Ion batteries can stores more power than other battery types.
      - These kinds of batteries do not suffer from “memory loss”, i.e. their storage capability doesn’t decrease with time.
    - Cons:
      - These batteries will have to recharge often once a day or every few days.
      - Lithium Ion batteries are lighter, and the client wanted a heavy battery for a bigger center of gravity.
      - These kinds of batteries can physically decay faster than Lithium Polymer.
  - Lithium Polymer Batteries
    - Pros:
      - These batteries are commonly used because they are considered to be robust for various robotics applications.
      - These batteries can be rechargeable.
      - The chance of leaking from these batteries is very small.
    - Cons:
      - Lithium Polymer batteries tend to be very expensive.

            - The amount of power that can be stored in them over time decreases.
            - The overall amount of power in lithium polymer batteries is lower than Lithium Ion.
- AA Batteries
    - Pros:
        - AA batteries are very cheap and don't have to be ordered online.
        - AA batteries are extremely common and are easier for the client to replace.
    - Cons:
        - These kinds of cheap batteries would have to be replaced fairly often for a circuit that requires the kind of power we have.
        - Rechargeable options are available but they are more expensive.
        - They would have to be put in series to get the power we need, which isn't a very good practice when it comes to circuit design.
- Battery Pack (aka Power Bank)
    - Pros:
        - Power banks are very common and can be bought at most department stores.
        - These devices can hold around 12V which would be perfect for a microcontroller.
        - They are rechargeable but don't need to be recharged as often as other alternatives.
        - Various voltage options are available, rather than the set AA, A, etc. voltages.
    - Cons:
        - Power banks are more expensive that typical A,AA, D, etc. batteries.
        - They are very light, and don't contribute very much to the center of gravity.
- Option Summary

Criteria	Lithium Ion (motors)	Lithium Polymer (motors)	AA (micro-controllers)	Battery Pack (micro-controllers)
Size	13.25" x 4.75" x 1.2"	6.1" x 4.0" x 2.4"	1.94" length, 0.53" diameter	4.9" x 3" x 0.9"
Rechargeable	Yes	Yes	Possible but more expensive	Yes
Power Availability	High	Low	Low	High (for micro-controller applications)
Possible Ah (in price range)	4-10Ah	4-8Ah	NA	NA
Price	~$200	~$250	~$20	$59

Table 5: Batteries Summary

- Proving Feasibility: To test if the batteries would work for this circuit, we need to make sure both motors receive a similar amount of voltage and current went wired up.
- Chosen Approach: Lithium Ion Batteries & Power Pack
- Conclusion: The Lithium Ion battery would be rechargeable, and it less expensive that a Lithium Polymer battery. Lithium Ion batteries also don’t suffer from what is known as memory effect, where batteries get harder to charge over time. Lithium Ion batteries also have overall higher power than other battery types, meaning it would be easier to get more current to the motors. The only issue with this type of battery is that it is lighter than other types of batteries, meaning the center of gravity for the robot will be slightly less stable. We chose a battery power pack for the Arduino and Pi because not only are battery packs more reliable and have more voltage output than regular AA and AAA or even D batteries,they also have a much longer battery life. This means they need to be charged less. Battery packs are more expensive and heavier than regular rechargeable batteries though, but it will help adding weight to the center of gravity for the robot.

3.4.4 - Motors & Wheels

- Introduction: Wheels and motors are needed in order to move robot, but without drawing too much power. The wheels and motor must be able to support the weight of at least 100 pounds, or what is considered to be the weight of all equipment and still be able to move at a walking pace. Price and stability of the motors and wheels were also considered.
- Alternatives: Several motor and wheel designs were considered as well as the motors and wheels themselves. The motors and wheels were selected in pairs because a standalone motor may not work for some wheels.
    - Four Driver Wheels
        - Pros:
            - The most control over the turning of the robot
        - Cons:
            - More hookups and power draw
            - Programing for navigation may be more difficult
    - Two Driver Wheels, Two Caster Wheels
        - Pros:
            - Having four wheels would make the robot more stable
            - Less programing for navigation
            - Less power draw
        - Cons:
            - The turning could be more awkward and that would have to be compensated for
    - Two Driver Wheels, One Caster Wheels
        - Pros:
            - Triangular design would have better turning radius
            - Less programing for navigation
            - Less power draw
        - Cons:
            - May be less stable, or more difficult to balance other parts
    - Roombas
        - Pros:
            - The robot would be prebuilt
            - The navigation system already in the robot could be utilized

        - Cons:
            - The Roomba is more expensive
            - The pre-programmed navigation might be difficult to access and modify
            - The robot may not be able to hold the weight of all components
    - Used 2007 Permobil Electric Wheelchair Parts:
        - Pros:
            - The whole thing is a set and comes with a motor control
        - Cons:
            - The set can hold only 60 lbs, it may not be enough
            - It is one of the more expensive sets
    - Parallax Motor Mount and Wheel Kit - Aluminum
        - Pros:
            - It is a wheel and motor set
            - It is designed to hold a person and move at a walking speed
        - Cons:
            - We will need to build a motor controller
            - Very heavy
    - Option summary
        - Designs

Criteria	Four Driver wheels	Two Drivers, Two casters	Two drivers, One Caster
Stability	Good	Good	Poor
Level of difficulty for team to modify	Difficult, more hookups and programming	Least difficult, only two motors	Moderate, there may be issues with balance
Turning Radius	Fair	Fair - Poor	Good
Power consumption	Higher	Lower	Lower

Table 6: Wheel Positions Summary

        - Wheel/ Motor Options

Criteria	Roomba	Used Wheelchair Parts	Parallax Kit

Ability to carry weight	Virtually no ability	High Ability	Low ability
Price	$200	$160	$299
Programmability	Difficult to access on robot programs	None	Easy to access

Table 7: Wheels and Motors Summary

- Proving feasibility: The ability of the wheels and motor will be tested by usage. Sample programs and sample power sources will be used for testing the parts. Stability can be tested to see if the cart wobbles when moving. All parts can be set on the robot (without being plugged in) to see if the robot can move with weight on it. Turning radius can be tested by how quickly it rounds corners and if it hits anyone or anything while turning. Power consumption can be calculated by amps per hour.
- Chosen approach: 4 Wheel Design (2 motor wheels, 2 caster) with the Used Wheelchair wheels and motors
- Conclusion: The four wheel design is the most stable and using only two drive motors will draw the least amount of power, meaning we could use less or smaller batteries. The wheels come with the motors, so there is no compatibility issues between the motor and wheels as opposed to other applications. With this design, no motors would need to be added to the caster wheels, and programming two motors would be easier than four. The wheels and motors are also made out of a stronger metal and are designed for rigorous and repeated use, making it a better choice for a robot meant to be used for several hours day.

3.4.5 - Motor Controller

- Introduction: In order for the robot to move autonomously, it must be able to stop and turn, and be able to provide a steady current and enough voltage to both motors. This project requires a dual motor controller circuit that can dissipate heat effectively. The controller design chosen needs to fit the following parameters:
    - Able to handle high power applications, i.e. 22V and 3-4A per motor
    - Steady supply of voltage and current to both motors
    - Able to supply different currents/voltages to either motor at the same time
    - Design chosen needs to allow the addition of lots of heat sinks

    - Reasonable difficulty to design and build
- Alternatives: Some motor controller solutions the team considered were h-bridge chips, a new motor controller, or a repurposed one taken out of old tennis ball machine owned by a member of the team.
    - H-Bridge Chip(s)
        - Pros:
            - These chips help create a more modular of a design because the h-bridges could be used for multiple different kinds of motors and circuit designs.
            - These chips come with specification sheets and pinouts that make them easier to use and understand.
        - Cons:
            - Finding a chip with the correct power rating would be hard for out application, and would likely be fairly expensive if even sold.
    - Repurposed Motor Controller
        - Pros:
            - We can be sure that the design works properly with some kinds of motors, so this is a fairly reliable alternative.
            - Repurposing an existing controller would be very inexpensive compared to the other options.
        - Cons:
            - A fully put together circuit removes modularity and customization from our circuit design.
            - The repurposed controller would likely have to be redesigned to handle the specifications of the motors we are using, because they are certainly different from the motor options we considered.
    - New Dual Motor Controller
        - Pros:
            - This would be the easiest design to understand, it will probably come with a detailed specification sheet and a pinout diagram.
        - Cons:
            - A fully put together motor driver board wouldn’t be very modular. It would be hard to add any additional functionality or parts to it.

        - These devices commonly have lower power ratings and so it would be hard to find one with the right current and voltage rating for our application.
        - Motor Driver boards are much more expensive than the other options because they are already put together for you.
  - Custom H-Bridge Circuit
    - Pros:
      - Building this motor driver circuit ourselves would make the design very customizable, there are definitely high power transistors that we could use.
      - Though we would have to buy many of the parts, designing our own would still be cheaper than buying a motor driver board or chip.
    - Cons:
      - H-bridges are known to be tricky to design because of parts matching due the symmetrical nature of the circuit.
      - This would be the most complicated design chosen above all the others, because we'd have to design and build it from scratch.
  - Option summary:

Criteria	Repurposed Motor Controller	New Motor Controller	H-Bridge Chips	Custom H-Bridge Motor Driver
Ease to Implement	Medium Difficulty	Low Difficulty	Medium Difficulty	High Difficulty
Power Capability	Fair	Poor	Fair	High
Customization Ability	Poor	Poor	Fair	High
Price	~$10	$50	~$10	~$15

Table 8: Motor Controllers Summary

- Proving feasibility: The chosen design needs to sufficiently route power to the motors without damaging the batteries or it's own components. The

design must be able to control either motor individually so that the robot is able to successfully turn as well.
- Chosen approach: Team-manufactured H-bridge chip
- Conclusion: We are going to have a very high power circuit in terms of robotics applications and as such, we need very high current and voltage rated components. Most motor controller boards do not have a high enough voltage rating and split the voltage between motors, meaning the most the motors would get at most 12-18V rather than the 22V each we need. There are some h-bridge chips that can handle higher current, be we would get the most flexibility out of a circuit that we build, because we could get very high rated parts and combine, but it would take longer and be somewhat complicated.

3.5 - User Interface

3.5.1 - Manual Controls

- Introduction: Before the robot is capable of autonomous navigation we will need a method of manual control over the robot to facilitate tests. Once autonomy is achieved, manual control will be needed. Therefore, it is essential we have a reliable way for users to input instructions, whether it be from a physical controller or virtual interface. This component must have the following:
    - Ability to interface with microcontroller brands & navigation frameworks under consideration
    - Allow for the robot to have room for motion, i.e. any wires, tethers, or limited-range wireless connections allow the robot at least 2’-3’ of movement beyond the point of the control
- Alternatives: We have taken both wireless and wired solutions under consideration and decided to aim for one of each solution for maximum versatility in testing.
    - Options:
        - USB controller
            - Pros: button inputs remove need for command line control knowledge
            - Cons: robot range limited by length of USB connection
        - Wired laptop connection
            - Pros: Larger variety of input/output information potentially available from command line, debugging software, etc.

            - Cons: laptop has limited battery without connection to power source, laptop bulky and inconvenient to carry/type on while following robot
        - Bluetooth controller, mobile interface
            - Pros: potential for controls to be downloaded onto multitude of devices
            - Cons: Small connection distance, need to build application to interface with
        - Remote desktop connection
            - Pros: Essentially gives control from any distance via wifi signal- limitless range for robot
            - Cons: relies on microcontroller/other hardware capable of receiving wireless signal, hosting connection
    - Option Summary

Criteria	USB Controller	Wired Laptop Connection	Bluetooth Mobile Interface	Remote Desktop Connection
Range	~2”-10”	~2”-10”	328’	∞ (Dependant on wireless internet)
Wireless?	No	No	Yes	Yes
Difficulty of Setup	Low (Framework Libraries available)	Low	High (application development)	Mid

Table 9: Controllers Summary

- Proving feasibility: First we will make use of our small-scale robot to test our chosen control solution. The control solution must be able to make the test robot go forward, backward, and stop, allowing the robot the ability to get at least 1’ of distance between itself and the control. Once this stage concludes we will repeat this step on our full size robot.
- Chosen approach: Remote Desktop connection for wireless solution, USB controller for wired solution.
- Conclusion: Remote desktop can be used with a microcontroller like one of the Raspberry Pi units and USB controllers are compatible with all microcontrollers and frameworks under consideration. Remote desktop fits well because it allows for direct control of a microcontroller, which will

let us to receive visual output in addition to the ability to send input. Additionally, it would be essentially wireless and allow control from any distance that wireless internet can accommodate. The USB controller was ideal for wired controls because of its accessibility to non-programmers and cheap price.

4.0 - Technology Integration

4.1 - Introduction

The hardware and software identified as solutions for our technological challenges in section 2.0 need to work as a cohesive whole, rather than function in four isolated parts. Each of the four challenges identified key hardware and software needed to solve a sub-problem of our project, but in order to solve the big-picture problem these solutions need to cooperate. This section details how the components of each of the four challenges will interact.

4.2 - Integration of Components

The four technology challenges consist of navigation, obstacle avoidance, runtime and efficiency, and user interface. Each of these components have a hardware and software solution used to implement them to best overcome the challenge. These components must integrate as a single entity - a tour giving robot. The biggest integration is between navigation and obstacle avoidance. The robot should not mindlessly follow a predetermined path and must be able to avoid unpredictable collisions with hazards and passing individuals in the hallways. The robot must be able to take input from its surroundings via the sensors and make adjustments to is path using the software solution implemented in the Raspberry Pi. In addition to this, the user interface is integrated to the system by having the robot react to the individuals who are receiving tours. The robot may have to stop to receive and process commands given to it. All of the previous components must be able to function without putting too much strain on the batteries, or the batteries on the components. Runtime and efficiency will directly be affected by how the motor controller circuit and batteries effectively work together. All solutions to the challenges will draw some amount of power and different types of battery systems need to be used in order for the robot to run for the desired 4 -6 hours.

5.0 - Conclusion

5.1 - Problem Review

Our team is working to implement the planning, construction, and programming of a roughly 100-pound tour guide robot. This multidisciplinary project puts our team up against four main challenges: Achieving autonomous navigation, managing safe and timely obstacle avoidance, exhibiting the capability to run for a set amount of time on a single charge in the most efficient way, and possessing a system that allows for direct user control or interaction with the robot.

5.2 - Summary of Solution

Technical Challenge	Proposed Solution	Confidence Level (1 - low, 10 - high)
Navigation	Arduino Uno, Raspberry Pi 3B, ROS	9 Very commonly used in robotics, easy to program
Safety & Obstacle Avoidance	Microsoft Kinect, IR sensors	7 Microsoft Kinect sufficiently accurate, IR sensors should be good for obstacle detection
Runtime & Efficiency	Lithium-ion batteries, Used wheelchair wheels	7 Lithium Ion are rechargeable, wheelchair wheels have very high power ratings but are reliable
User Interface	Laptop, Raspberry Pi	10 The computer science portion of the team is confident in the command line skills needed for rudimentary direct control

Table 10 - Solution Summary

Table 10 above summarizes the proposed solutions for the four main challenges addressed in this document. Each of the solutions is accompanied by a confidence level rating and description to explain said rating.

5.3 Final Remarks

Our team's overall confidence is high - based on the research presented in the analysis section, we are 85% confident in our proposed components and resulting solution. In the eight weeks since Keystone Robotics formed, we have spent dozens of hours researching and discussing each component of the robot and how each piece suggested would best contribute to the challenges we face. Our team is dedicated to leaving the best possible product behind so future students can build even greater projects using our work as a strong foundation. We are firmly dedicated to meeting these identified challenges head-on and overcoming them by making use of the planning summarized in this document.