Design of A Hands-Free Control Interface For Object Relocation

Final Report

Sponsor: Mike DeMichele, General Dynamics Mission Systems
Faculty Advisor: Dr. Lance Sherry
Kassidy Kenney, Angelo Huan, Kimberly Harrington, Muhammad Sungkar

George Mason University, Department of Systems Engineering and Operations Research
4000 University Drive, Fairfax VA 22030

Contents

List of Tables

List of Figures

1.0 Concept Definition

Approximately 3 million Americans identify as severely paralyzed. These individuals either struggle to perform basics tasks or experience some form of dependence as a result of their physical paralysis.

A multitude of devices have been devised for the physically disabled population, however, a severely limiting factor in the ability of this population to benefit from devices is their ability to have the device brought to them. Currently, in-home aids provide support for physically disabled persons, but at a high cost.

A fetch-and-deliver robot coupled with a system which requires no physical input would provide a degree of independence and improve the quality of living of this population. This opportunity to improve the lives of the paralyzed population has become realistic and affordable as a result of several converging technical advances. Both robotic aids and hands-free control systems have become readily available, creating a market space for the coupling of these technologies. This project will study the available control interface methods to determine the best configuration for a coupling interface.

2.0 Context

2.1 Control Interfacing

Every interaction between a human and a machine is performed through an interface. Almost every interface requires the physical manipulation of the interface. Keyboards, touch screens and steering wheels are all examples of interfaces requiring physical manipulation. In many cases, the user experience, usability or operator safety is impaired by the need for a physical input device. Without the interface, the device is useless - a car with steering mechanism, a computer that does not allow user input. This “one size fits all” approach to human-machine interaction has created over dependency and inefficiency.

At the simplest level, the necessity of a generic hand-operated physical input device prevents and able-bodied operator from performing additional tasks simultaneously. An easy example of this is driving a car. A car has a variety of physical input devices. The steering wheel requires the driver to provide directional instruction via hand movement. Manipulation of pedals controls the speed of the vehicle. Radio and heat controls require manipulation of buttons and knobs. Additional complexity is added if headlights, windshield wipers, or a lane change is required. Adding additional controls requires adding an additional physical input device which in turn continues to increase the driver's workload. This snowball effect of a multitude of physical input devices is seen in a huge array of machines and multiplies when a user is asked to operate multiple devices at the same time. Jets, cars and drones are designed to maximize the workload of a single operator, but adding an additional task often pushes the operator past a safe limit.

In other systems, the physical interface renders the system unusable for a subset of users. An electric wheelchair with a joystick to control its speed and direction can make a huge difference in the independence and quality of life for a large portion of the physically disabled persons, however, the same device would be useless for a paralyzed person but it requires physical input the operator cannot provide.

2.1.1 Human-Machine Interfaces

A Human-Machine Interface (HMI) is any interface facilitating interaction between human and machine. HMIs have greatly advanced since the beginnings of computing. Original HMIs were driven by single-selection menus. Since then, HMI has progressed to mouse - keyboard controlled interfaces, and now to touch screen interfaces.

In many cases, the system will require constant attention and input from the user to provide small corrections as the system performs functions. In these cases, repeated actions can create additional stress on the user.

Examples of human-machine interfaces are plentiful in everyday life. A steering wheel is required for a user to manipulate a motor vehicle; a touch pad on a garage door enables a user to open or close the door. A computer relies on two physical human machine interfaces- a mouse and keyboard- to gather data from a user. All of these systems follow the same process outlined in the case study diagram shown in Figure 1. The figure depicts the HMI of an RC Car. As shown, the user's commands and mental model for the intended movement are essentially useless without the physical interface - in this case a remote control.

Figure 1 HMI of an RC Car

These interfaces are largely successful for users with full control of motor functions. However, in persons without control over motor functions, systems where an operator must quickly switch between control interfaces or complex systems requiring a large amount of user input, these systems are inefficient or unusable.

2.1.2 Alternatives to Current Control Interfaces

A number of alternatives to traditional control systems are in various stages of use. Table 1 gives 6 examples of alternatives to a physical control interface. Alternatives will be analyzed in more detail in Section 7.1.

Table 1 Alternative Control Systems

[1] Voice Control 	[2] Eye Tracking Software 	[3] Muscle Contraction Detection 
[4] Head Movement/ Gyroscope 	[5] Predictive Analytics 	[6] BCI Interface 

Each of the control interfaces in Table 1 has unique benefits and challenges. Voice control is a popular hands-free control method which works well in systems without huge risk, in systems with minimal background noise and in systems without strict time constraints. Voice control is not effective in systems where the operator is under a large amount of stress or where a missed command has serious ramifications [2].

Eye tracking software is a very effective human-computer interface and has been proven to give a user a level of control comparable to using a mouse. Eye tracking software, however, has been much less effective outside of interaction with a stationary computer screen [3]. Muscle contraction detection is a growing technology especially in providing an operator with control

over a prosthetic limb. This technology relies on a functioning nervous system and muscle system in the operator [4].

Gyroscopes have been built into a variety of headset devices to allow a user to control the motion of a device by tilting their head in accordance with mapped signals. This technology provides a limited number of possible commands, and very little fine grain control.

Predictive analytics, as used in the Google Self-Driving car, work to distance or remove the human operator from the system. While this technology is not specifically a control interface, it is important to consider that the human operator may not be necessary for more than a start command [5].

The final grid section shows a brain-computer interface (BCI). Brain-computer interfaces are largely considered to have the highest potential of the interface options. However, this technology is largely still developing [6].

2.2 Developed Aids for Paraplegics

There are many types of generic handheld assistive technologies available to paralyzed individuals. These technologies include but are not limited to: smartphone applications that operate using head-tracking sensors, switches that activate via breath-detection, and voice-activated IR remotes. At some point, most paralyzed individuals opt for custom-designed assistive products that better suit their personal, ergonomic needs. If devices such as these are out of a paralyzed person's reach, the hands-free control system presents a convenient solution, while offering more independence to the user.

2.2.1 Robotic Aids

With modern advances in robotics, the notion of integrating robotic aid systems in place of in-home care assistants is gradually becoming a reality. Examples of contemporary robotic aids include Robo-nurse, which is an automated robot used to lift and transport patients from one bed to another, and Care-O-Bot - still currently still in research - which helps users cook and clean as well as perform other house tasks. Among this spectrum of robotic aids, the hands-free control

interface is specifically designed to fetch and deliver items for a user who is disabled. Robotic aids in development will eventually alleviate the shortage of in-home care nurses.

2.2.2 End Use Applications

Due to its versatility, the end-use applications of this system are virtually limitless. With extra funding and more research in machine-learning algorithms, the number of commands a user could input is potentially unlimited. The applications of this product could expand to a wider market of users, including individuals who do not display any difficulty moving, yet seek a robotic aid for help around the house.

2.0 Motivation

According to a 2012 study performed by the Christopher and Donna Reeve Foundation, 1.9% of the American population identifies as paralyzed. Of these 5,596,000 people, 2,909,920 people identify as “unable to move” or “a lot of difficulty” performing any motion [7]. For these persons, even devices specifically created to help persons with physical disabilities are rendered useless as long as the user is required to perform physical motion to use the device.

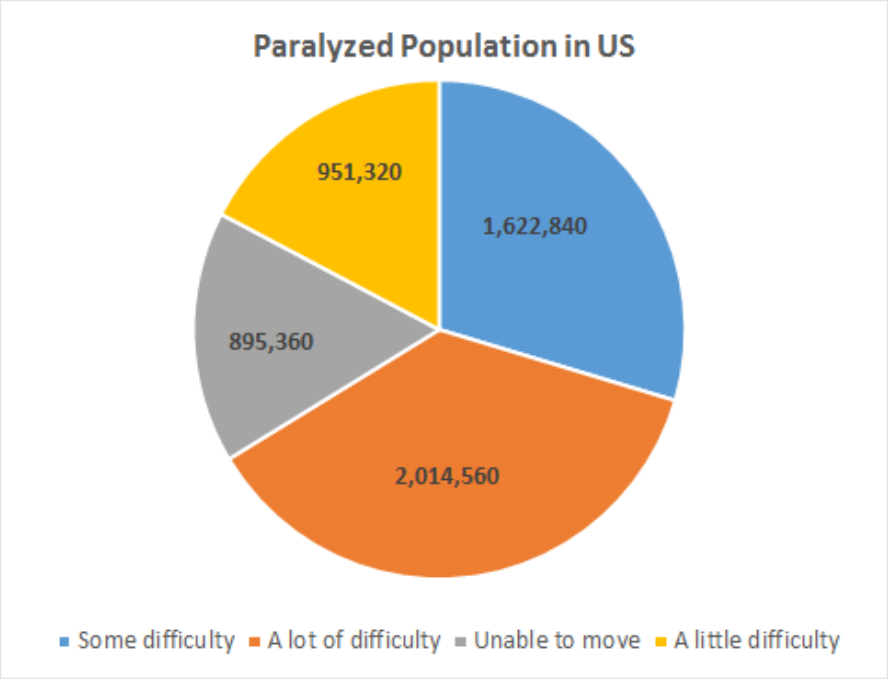


Figure 2 Paralyzed Population by Degree of Movement

In addition, New York Times article written in 2014 cited a shortage of 1.3 million in-home caregivers in the United States in the next decade. The New York Times article continues by explaining that the probability that this shortage is filled is low- whereas in most markets supply and demand would enable “drive up prices in the labor market,” it is not expected that the average hourly wage would rise above the $20 an hour national average. Almost 75% of the

current American caregivers are paid for by the Medicare or Medicaid programs making it unlikely that a significant raise in salary would be feasible.

3.0 PROBLEM OVERVIEW

There are almost 3 million Americans who identify as severely paralyzed and are unable to perform basic movement. A multitude of devices, as detailed in Section 2.2.2 have been devised to mitigate the impact of these persons disability. However, the paralyzed population still has no way to perform fetch and deliver tasks which would enable them to operate these devices without the aid of a caregiver or assistant.

End-point devices (Section 2.2.2), robotic aids (Section 2.2.1) and interface alternatives (Section 2.1.2) have been previously developed in order to aid this population. A comprehensive solution is needed to bind these technologies in order to provide a safe, reliable object relocation system which does not require tactile input from the user.

3.1 Gap Definition

The necessity of physical movement of the control interface is a highly limiting factor is the system usability and user workload. For the 2,900,000 severely paralyzed persons in America, the necessity of physical motion to operate a device interface renders the system unusable. A "win-win" scenario is to create a hands-free control system interface which can match the quality and cost of a traditional interface.

3.2 Problem Statement

The 2.9 million severely paralyzed individuals in the US require a technology to enable them to relocate objects without requiring physical interaction. This design has the potential to assist users in other daily activities that ordinarily rely on physical input, such as navigating a wheelchair.

3.3 Need Statement

Paralyzed persons need an alternative interface device which will allow them to manipulate a robotic aid device without requiring physical motion from the operator.

4.0 Scope

Figure 3 shows the convergence of technology and need which has allowed for a feasible, timely solution to the problems faced by the severely paralyzed population.

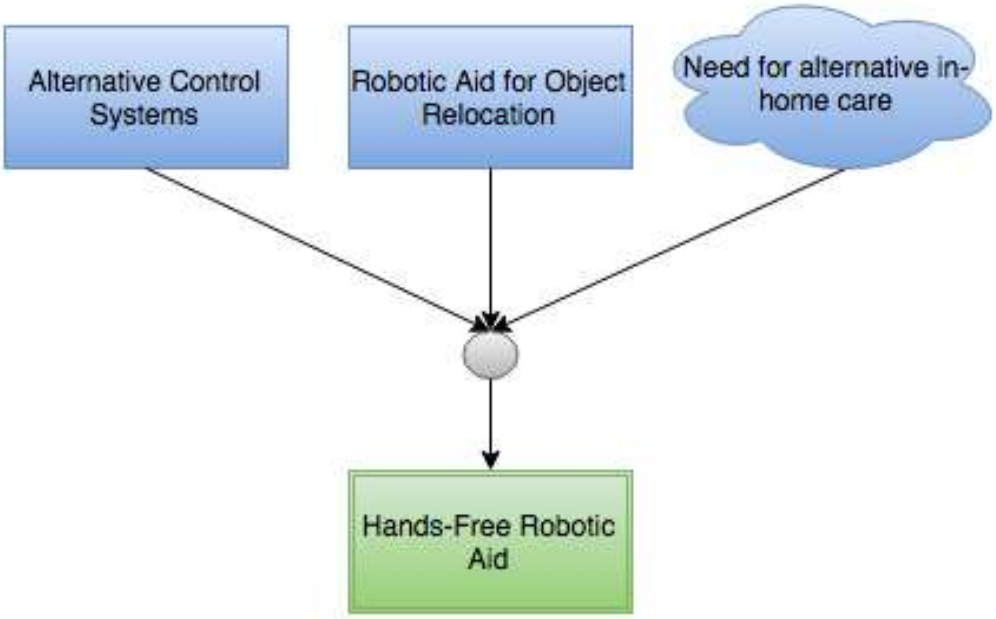


Figure 3 Converging Technologies

Our projects focus is to determine the best way to join these existing technologies with the needs of the paralyzed population. Before a new control interface can be selected for the robotic aid, we must examine the maneuvers which the robot will need to perform. This project will perform a simulation to collect and analyze the data crucial to creating a hands-free control interface. Once the requirements for a hands-free interface for object relocation system have been determined and quantified, a trade-off analysis will be performed to determine the best available interface technology.

5.0 Stakeholder Analysis

The end-point users of the system and primary stakeholders are the paralyzed population of Americans. Other stakeholders include In-home Care Agencies, Government Organizations responsible for approving and overseeing medical care and insurance companies. Table 2 summarizes the system's' stakeholders and the tensions between stakeholders. Each stakeholder will be analyzed in more detail following the table.

Table 2 Stakeholder Analysis

Stakeholder	**Significance**	**Tensions**
Severely Paralyzed	- Benefit from Hands-Free Control System	- Learning time - May initially distrust system
Interface Technology Companies	- Lead on current development work for alternatives - Potential to modify control technologies to best fit need	- Competition between companies - Tension with government approvers
Insurance Companies	- Create new policies to include HFCS	- Cost of insurance - Additional costs to insurance companies
HHS and FDA	- Under Medical Device Act, aid would require approval	- Tension if design is not accepted
In-home Care Organizations	- Leading care for severely paralyzed - Potential to integrate new robotic care device	- Concerns of being replaced/ losing market dominance

3.5.1 Physically Disabled Persons

According to the Christopher and Donna Reeve Foundation, a leading researcher in curing spinal paralysis and support provider for those faced with managing long term paralysis, 5,596,000 people in the United States currently report some form of paralysis. The degree of paralysis is defined by one of four descriptors ranging from “A little difficulty moving” to “Unable to move.” The primary focus of this project is the 2,900,000 Americans who identify as “A lot of difficulty moving” or “unable to move.”

The paralyzed population is quantified in several ways. Persons can be qualified as paraplegic or quadriplegic. Paraplegic injuries occur below the first thoracic vertebrae in the spine and typically result in paralysis below the waist. Quadriplegic injuries typically occur above the first thoracic vertebrae and results in loss of movement in all limbs. A form of quadriplegia is Locked-In syndrome, which leaves the person cognizant but with no means to communicate with the outside world. Persons with Locked-In syndrome are incapable of any movement, including that necessary to speak.

Quadriplegics are at a disadvantage, even when compared to other paralyzed groups. A wide array of technology developed for paraplegic persons is not suitable for quadriplegics. Examples include joystick-controlled wheelchairs and mouse-and-keyboard operated robotic systems.

3.5.2 Alternative Control Interface Manufacturers

Alternative control system manufacturers consists of any companies who may produce the technology necessary to assemble the hands-free interface. This includes companies researching and developing improved voice-command algorithms, those creating muscle contraction devices to operate prosthetic limbs, and those beginning to delve into the capabilities of BCI interfaces. These companies will be interested in the outcome of the simulation of this project and the insight which will drive the design of future products.

3.5.3 Insurance Companies

Health insurance companies would be required to modify policies to include funding decisions for hands-free robotic interfaces. In the case that the interface is successful and less expensive than in-home care aids, insurance companies may push to replace some care aide hours with robotic aids.

A second impact on insurance companies is the option to offer insurance on the robotic aid-interface system. Since the system is targeted towards the physically disabled population, the system can have a huge impact on health insurance and their coverage on control methods for the physically disabled.

3.5.4 Health and Human Services and Food and Drug Administration

The government is interested in protecting the wellbeing of individuals. Specifically the United States Department of Health and Human Services and the Food and Drug Administration, are stakeholders of EEG due to safety concerns. EEG is labeled as a very safe procedure according to the United States Library of Medicine and is used to diagnose seizures, epilepsy, tumors, and other head injuries. However, this only applies to EEG testing, which works by monitoring the brain activity. Our system differs in the sense that a user input is required and is designed to be used in a dynamic environment.

If the system is successful, an Investigational Device Exemption needs to be submitted to the Food and Drug Administration under a nonsignificant risk device to test its safety. Once approved, a clinical study can be conducted.

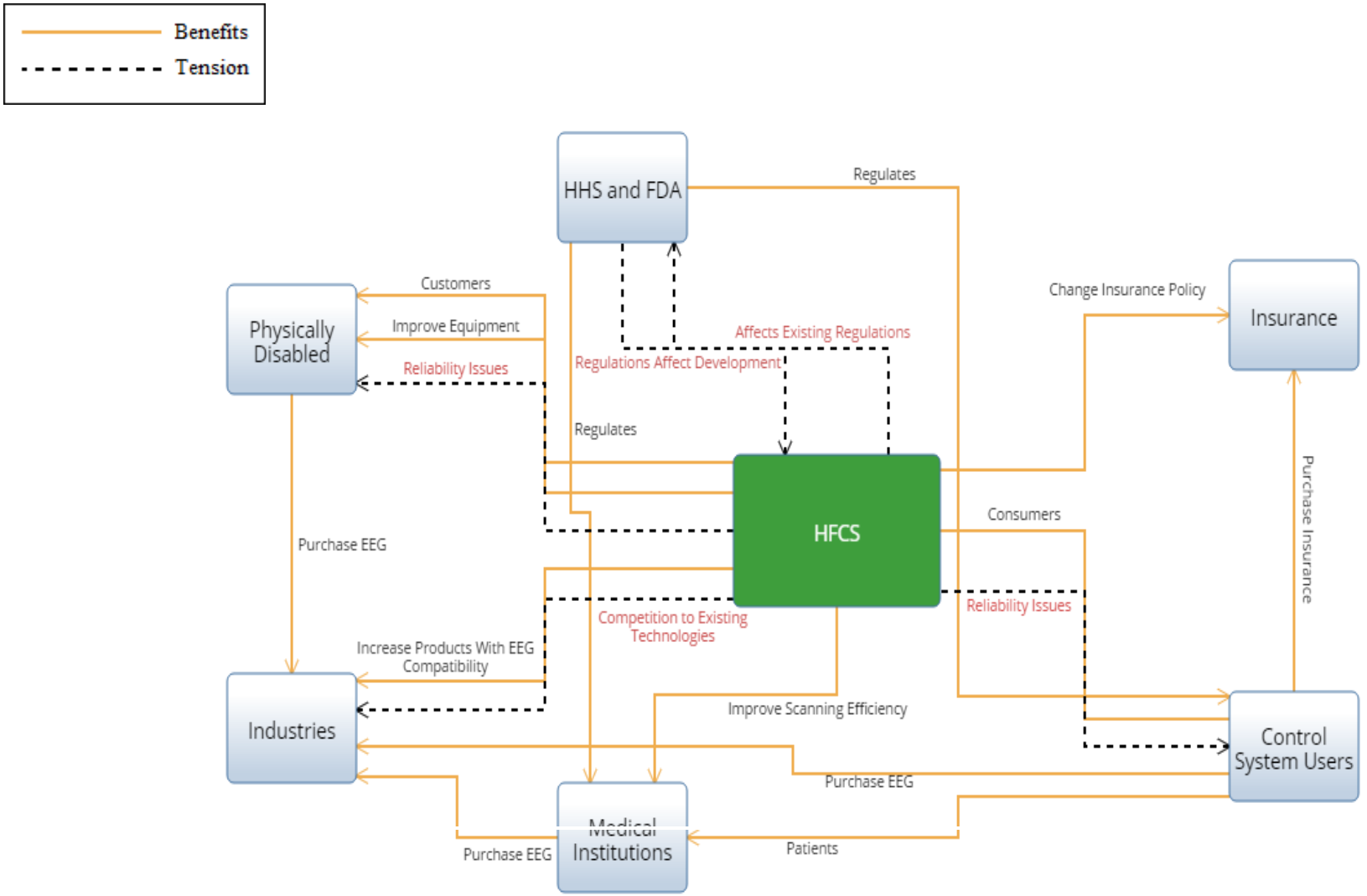
3.5.5 In-Home Care Organizations

In-home care organizations are currently the primary care provider for the paralyzed population. The expected shortage of caregivers, as described in Section 2.0 will impact the ability of care organizations to meet market demand for quality care. In-home care organizations may feel threatened by the introduction to an alternate care provider into the market, however, these organizations may choose to integrate the hands-free robotic aid with currently available care

options. In either case, in-home care organizations will be forced to define procedures relating to providing care with other options available.

3.5.6 Stakeholder Tensions

Figure 8 shows the effects of our system to the stakeholders. The main tension that can arise from the system to the control system users and the physically disabled is a reliability issue. Having a control system operated by solely the brain causes a considerable risk. As previously stated, introducing an integrable Hands-Free Interface to the market can cause competition amongst industries.


Benefits
Tension
HHS and FDA
Regulates
Physically Disabled
Customers
Improve Equipment
Reliability Issues
Affects Existing Regulations
Regulations Affect Development
Regulates
Change Insurance Policy
Insurance
HFCS
Purchase EEG
Consumers
Purchase Insurance
Competition to Existing Technologies
Reliability Issues
Increase Products With EEG Compatibility
Industries
Improve Scanning Efficiency
Control System Users
Purchase EEG
Medical Institutions
Purchase EEG
Patients

6.0 Concept of Operations

A paralyzed person who cannot perform basic motor tasks would be benefited by the assistance of a robot in order to pick up an item. The person will use a hands free control system in order to fluidly navigate the robot in a domestic environment. The robot is composed of a mobile platform, an arm, and a gripper. The platform can perform functions such as move forward, pivot to the correct direction, and stop. The arm can adjust its elevation and rotation. The gripper can clamp and open.

The robotic aid will operate primarily in a domestic setting and therefore should be able navigate tight spaces and be capable of avoiding obstacles.

6.1 Design Independent Concept of Operations

The operator will provide navigational commands to the robot via a hands-free human-machine interface. The robot will execute each command. The robot will execute automated collision avoidance to reduce operational risks.

6.1.1 Use Case Diagram

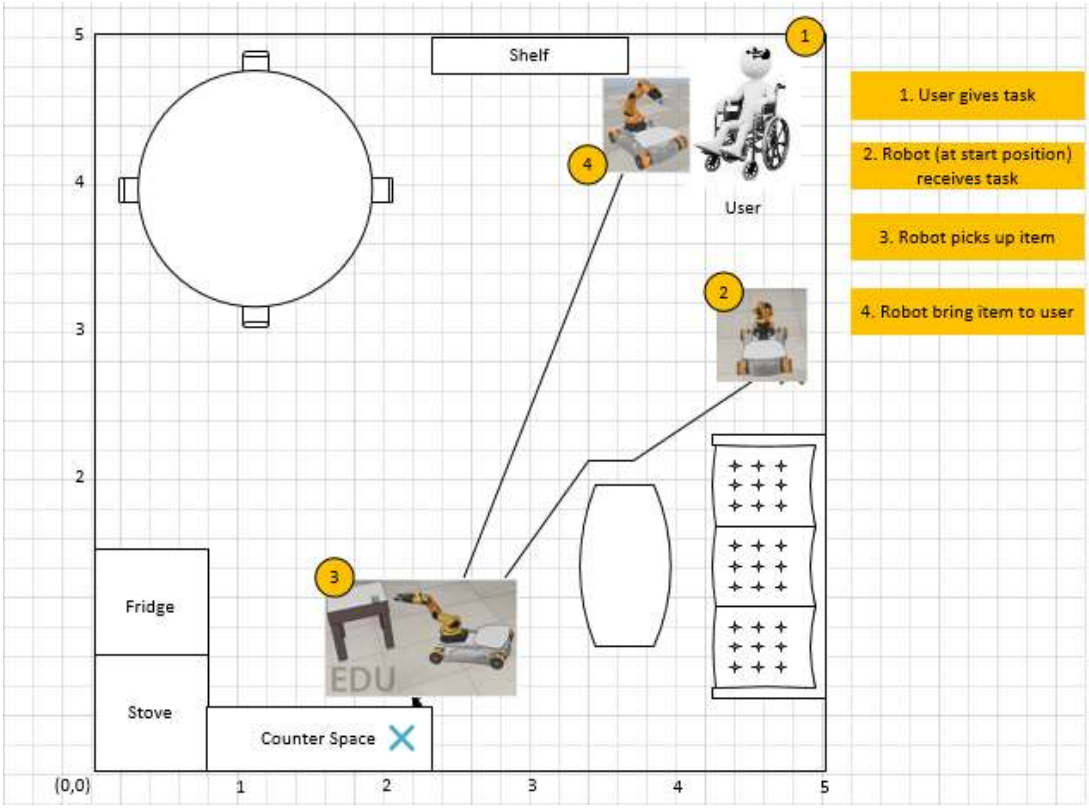


Figure 4 Use Case Diagram

The above diagram shows the process to be completed by the system. The robot will start in a random location in the room, travel to a fetch object, then return to the user.

6.1.2 Basic Operational Steps

The below set of steps is required in order for the robotic aid to fulfill its task:

1. Pivot to face object
2. Move forward
3. Stop at the table
4. Raise arm above table

5. Open hand
6. Move forward
7. Grasp object
8. Raise arm
9. Pivot to face user
10. Move forward
11. Stop in front of use
12. Pivot to face user
13. Adjust arm height
14. Open hand/ release object

6.2 Mission Requirements

R.0 The HFCS shall be operable without tactile user input.
R.1 The HFCS shall not harm the user in any way.
R.2 The HFCS shall provide flexibility for use for a variety of functions.
R.3 The HFCS shall operate in real-time.
R.4 *The HFCS shall be capable of differentiating between [X] commands.
R.5 *The HFCS shall be capable of executing [Y] commands.

While defining mission requirements, the project team determined that additional data was required before a trade-off analysis could be performed. A simulation was created to determine the number of commands necessary for the interface to perform domestic fetch and return tasks. [X] and [Y] in the above mission requirements will come from the simulation.

6.3 Requirements Decomposition (Mission and Functional)

The mission requirements listed in Section 6.2 have been decomposed into functional requirements. This decomposition is reflected below. These quantitative values in these requirements may be revised after the simulation is run and analyzed.

R.0 The HFCS shall be operable without physical user input.

R.1 The HFCS shall direct the motion of a robot in 2 dimensions.

R.1.1 The HFCS shall direct the robot to move forward.

R.1.2 The HFCS shall direct the robot to pivot left.

R.1.3 The HFCS shall direct the robot to pivot right.

R.2 The robot shall perform motion directed by HFCS.

R.2.1 The robot shall move forward upon command.

R.2.2 The robot shall pivot left upon command.

R.2.3 The robot shall pivot right upon command.

R.3 The HFCS shall direct the movement of the robot arm.

R.3.1 The HFCS shall direct the arm to raise.

R.3.2 The HFCS shall direct the arm to lower.

R.3.3 The HFCS shall direct the hand to open.

R.3.4 The HFCS shall direct the hand to close.

R.4 The robot arm shall perform movement directed by HFCS

R.4.1 The robot arm shall raise upon command

R.4.2 The robot arm shall lower upon command.

R.4.3 The robot arm shall open upon command.

R.4.4 The robot arm shall close upon command.

R.5 The HFCS shall not harm the user in any way.

R.5.1 The HFCS shall employ only non-invasive technology.

R.5.2 The HFCS shall weigh less than 25 lbs.

R.5.3 The HFCS shall be worn by an operator with head size min = 19 inches, max = 25 inches.

R.5.4 The HFCS shall not impair the operator's ability to move their limbs or neck.

R.5.5 The HFCS shall not impair the operator's ability move from one location to the next.

R.5.6 The robot shall employ automatic collision avoidance.

R.5.7 The robot shall employ an automatic stop function if signal is lost.

R.6 The HFCS shall provide flexibility for use for a variety of functions.

R.6.1 The robot shall have a total radius of 8 inches.

R.6.2 The robot shall be able to grasp an object up to 4 feet off the ground.

R.6.3 The robot shall be able to grasp an object up to 3 inches away from the edge of a table.

R.6.4 The robot shall be able to lift an object up to diameter 5 inches.

R.6.5 The robot shall be able to lift an object up to 5 lbs.

R.6.7 The robot shall be able to transport up to 25 lbs.

R.6.8 The HFCS shall operate for a minimum of 3 hours before recharging is required.

R.6.9 The robot shall operate for a minimum of 3 hours before recharging is required.

R.6.10 The HFCS shall broadcast signal to 20 ft radius of system.

R.7 The HFCS system shall operate in real-time.

R.7.1 The HFCS shall direct the robot within 2 seconds of user command.

R.7.2 The robot shall perform motion within 1 second of HFCS direction.

R.7.3 The HFCS shall correctly identify 75% of user commands.

6.0 Simulation of Robotic Aid Movement in Domestic Environment

A simulation of a robot performing the use case will be generated and automatically run with random start point, end-point and fetch points. Data collected from this simulation will feed performance requirements for the hands-free control interface and allow the developed of the hands-free control interface to make the best judgement as to what interface method should be selected. The statistics will also allow the developer to position the control options to that the options most frequently used will be the most accessible. Lastly, this data will allow the developer to draw conclusions about the level of accuracy required in controlling the robot in order to perform the use case.

Terms for Simulation

Fetch item - an item the virtual robot is instructed to pick up

Fetch point - the location of the fetch item

Task - a randomly generated trip from a start point, to the fetch point, picking up the fetch item and carrying it to an assigned finish point

Start location - initial (x,y) coordinates of the center of the robot's platform

Finish location - final location of the center of the robot's platform

6.1 Simulation Set-Up

The simulation utilized several software packages. The educational version of Virtual Robotics Experimentation Platform (VREP) served as the basis for the simulation. VREP offers a variety of pre-made models which can be applied to "scenarios." The model of the Youbot was utilized in this simulation [8]. Statistical and mathematical analysis were performed in Matlab using the Teaching Robotics Simulation (TRS) software plugin. This software package essentially translates the VREP model of the Youbot into Matlab functions for statistical simulation. TRS includes functions such as the D* algorithm which were used on 2 dimensional path planning.

6.1.1 Requirements for Virtual Robot and Simulator

The following requirements were defined in order to select a virtual robot simulator.

S.0.1 Virtual Robot shall accept input from HFCS.

S.0.1.1 Virtual Robot model shall contain a robot with 3D movement capability

S.0.1.2 Virtual Robot model shall contain a robot capable of grasping and relocating

S.0.1.3 Virtual Robot model shall contain an object to be relocated.

S.0.2 The Virtual Robot model shall allow for scripted iterations.

S.0.3 The Virtual Robot model shall allow for start location to be set.

S.0.4 The Virtual Robot model shall include a path finding function.

S.0.5 The Virtual Robot model shall include an obstacle collision function.

S.0.6 The Virtual Robot model shall run on Windows 7 OS.

6.1.2 Simulation Environment Set-Up

The simulation will be run in a 5 meter by 5 meter room, created in the VREP software. The room will contain 10 “fetch objects” located on shelves, tables, chairs and the floor, all located at a variety of heights and distances from the edge of the surface. Fetch objects will be a variety of shapes and sizes. This room will be held constant after its initial generation. Figure 5 shows the simulation environment. Upon creation of the room, 10 “start points” will be created. The Figure 6 below shows 10 such start points.

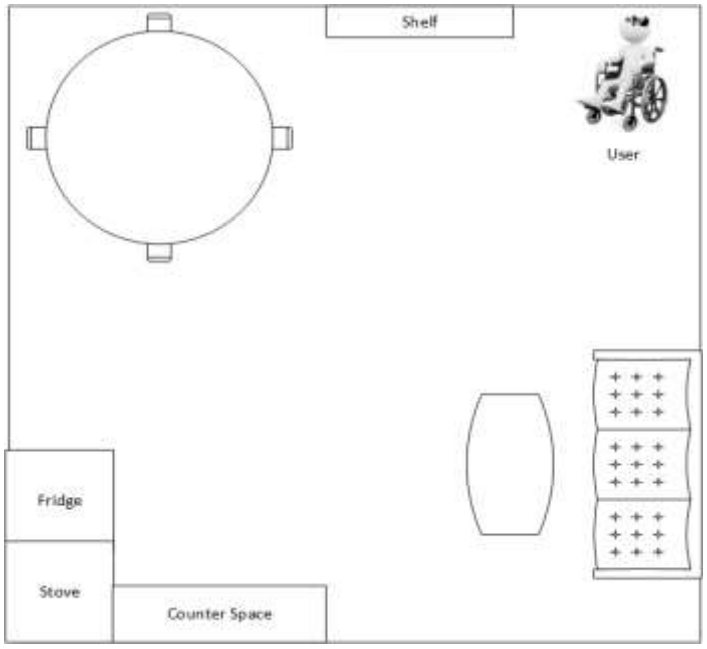


Figure 5 Simulation Environment

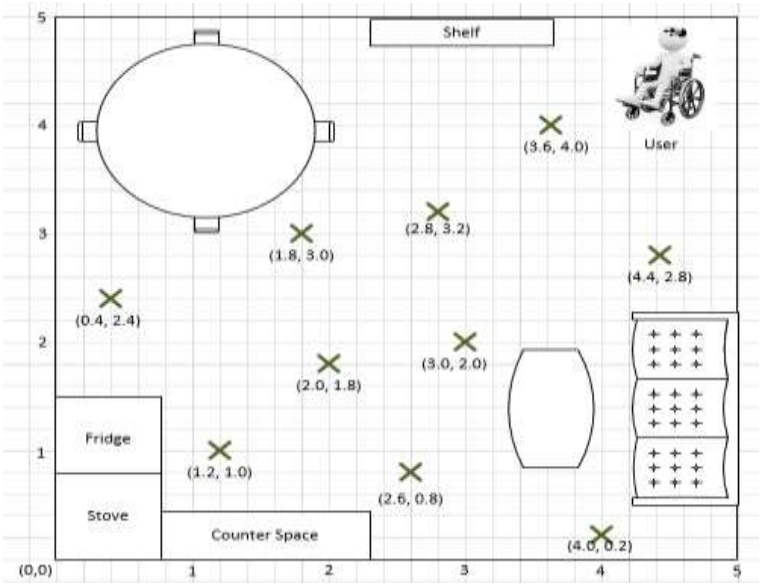


Figure 6 Array of Start Locations

A similar array of finish points was also generated, as shown in Figure 7. Fetch points will be assigned to include a "z" variable (height) in addition to the xy location. For this reason, the majority of the fetch points will be found on the existing furniture in the simulated environment to support this 3rd dimension.

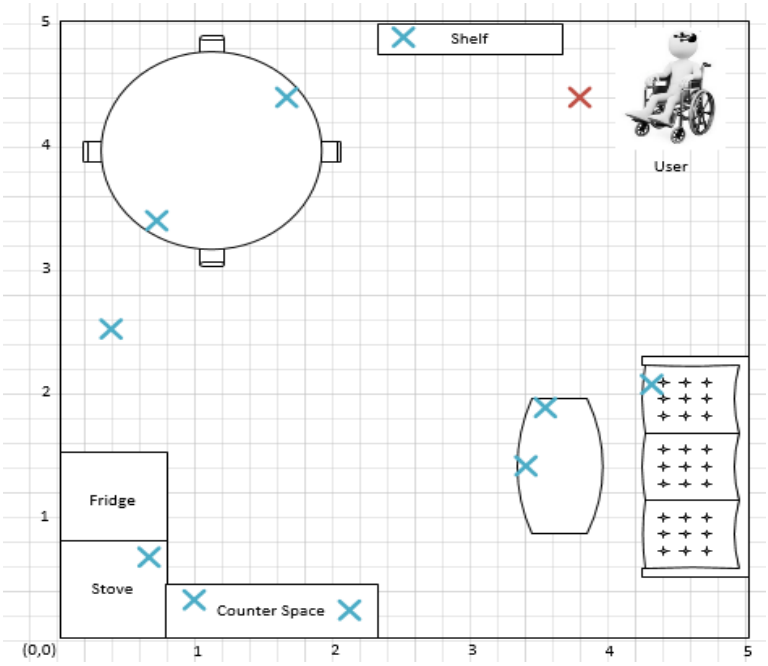


Figure 7 Fetch Points

6.1.3 Virtual Robot Set-up

The robot used for the simulation is the Kuka YouBot. This robot consists of 3 major parts, a platform, an arm, and a gripper at the end of the arm. The platform is capable of omnidirectional movement, zero turn radius turning and has a small surface to place objects. The arm has 5 joints for full 5DoF movement and limited pitch motion. The gripper is a 2 dimensional claw type gripper. The specification for the robot is below [9].

Platform

- Raise distance from ground .7m
- L x W x D: .58m x .38m x 14m
- Payload 20 kg
- Max Velocity .8m/s

Arm (Scale factor x1.4)

- Total reach .917m
- Payload .7kg

Gripper (Scale factor x1.4)

- Range .007m

Work Envelope .7182 m^2

The robot as simulated in VREP was smaller than what was needed for the design, so scale factors were applied to each aspect of the Youbot.

The youBot_gripperPositionTarget function allows the user to set the desired position of the gripper, and the system will adjust the arm accordingly so that the gripper position can be achieved [10].

Summary of Virtual Robot Moving Parts

- Gripper
    - Open
    - Close
- Arm
    - Up / Down
    - Forward / Backward
    - Left / Right
- Platform
    - Forward / Backward
    - Left / Right

Youbot velocity script simulation parameters allow for a minimum and maximum speed to be set for robot movement. These parameters may be utilized to ensure robot travels at the same velocity for each task [11].

6.1.4 Useful VRep Functions

The VREP simulation includes Path Planning module. This module requires the input of a start location, a goal position and obstacles (objects the robot should not collide with while completing the task.) Start and goal positions are identified with "dummies" - markers which appear based on the provided input parameters. Obstacles can be any imported shape or item, which are then identified as "obstacles". All non-fetch objects in the room will be identified as obstacles [12].

VRep includes a variety of sensors. The most important included sensor for this simulation will be the vision sensor and a ray type proximity sensor. The vision sensor detects the direction of objects and a ray type proximity sensor detects how far away the object is. An object can be labeled "renderable" when it is created. A renderable object can be detected by the vision sensor. The Youbot robot contains a function to pick up a renderable object, using its vision sensor, provided that the object is in the line of sight of the robot [13].

6.2 Simulation Design

The simulation was performed in two phases, one for the 2 dimensional aspect of the robot movement and one for the three dimensional movement. The 2 dimensional aspect a captures the robot's path planning and trip from start position to fetch position to end position. The 3 dimensional simulation focused on the movement of the arm and gripper once the robot's platform was in position.

The 2D simulation was performed primarily in Matlab, using the TRS plugin to provide data from the VREP model to the simulation. The 3D simulation was performed in VREP.

6.2.1 Input-Output Diagram

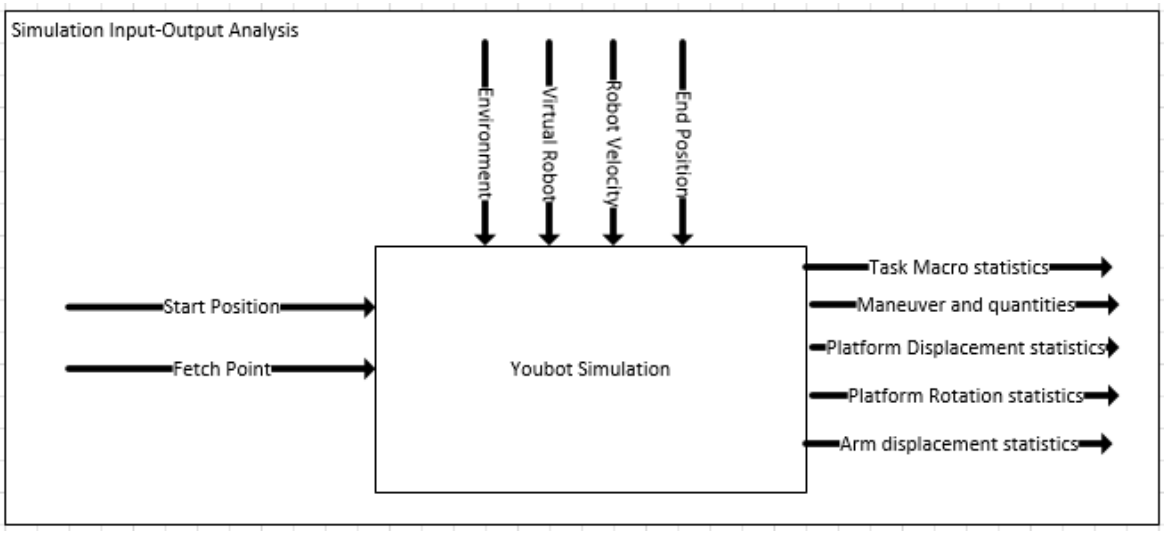


Figure 8 Input-Output Diagram

Figure 8 shows the system's inputs, controls and output variables. The start and fetch point variables will be randomly selected.

6.2.2 2D Simulation Design

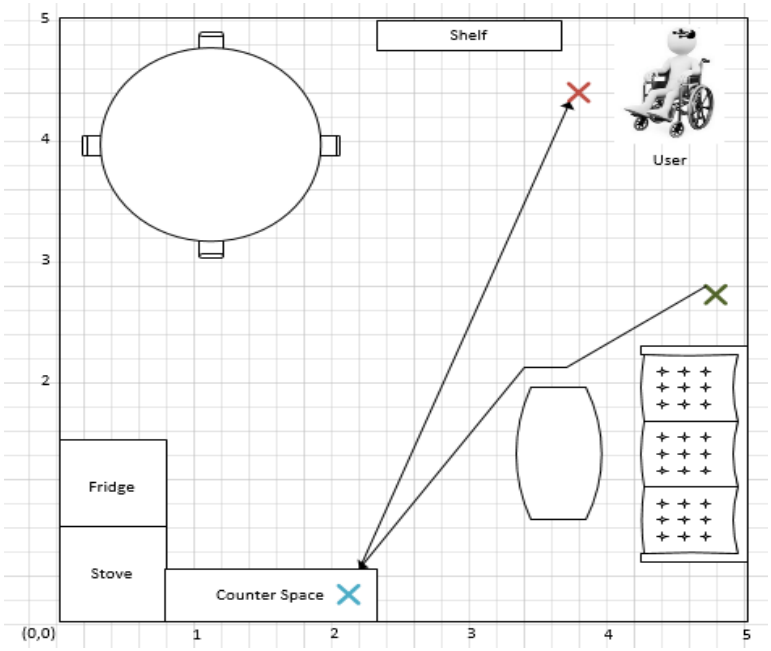


Figure 9 Sample Path

6.3 Simulation in 2 Dimensions

The two dimensional simulation was completed in three modules, as described by Table X. Source code for all modules of the simulation can be found in Appendix C.

Table 3 Simulation Modules

Name	**Description**	**Output**
GeneratePath.m	Select start and fetch coordinates then uses D* navigation algorithm from Robotics Toolbox	Array of coordinates stepping through shortest path to fetch location.
StatisticsScript.m	Locates turns along shortest path. Calculates angle of turn and distance traveled on straight-aways	Matrix Column 1,2 - (x,y) Column 3 - angle of turn @ pivot point Column 4 - distance traveled in previous straight away
GraphicalData	Displays frequency of maneuvers	Bar graphs for frequency of travel distance and angle
CallAll	Calls the 3 previous scripts, allows you to enter the number of trials	Outputs of the three scripts above.

6.3.1 GeneratePath

The simulation begins by randomly selecting one start point, fetch point and end point from the array using:

```
%Randomly select a start location
s = round (rand(1) * 10);
```

The simulation displays these values, then initiates the D* Navigation Function from the TRS toolkit, which uses the occupancy grid, start point and fetch point as inputs. A sample occupancy grid can be seen in Figure 10. The red objects in the occupancy grid mark obstacles in the room

which the robot must avoid. The green line is a display of the shortest path as calculated by the D* algorithm. The robot cannot pass through or collide with marked obstacles.

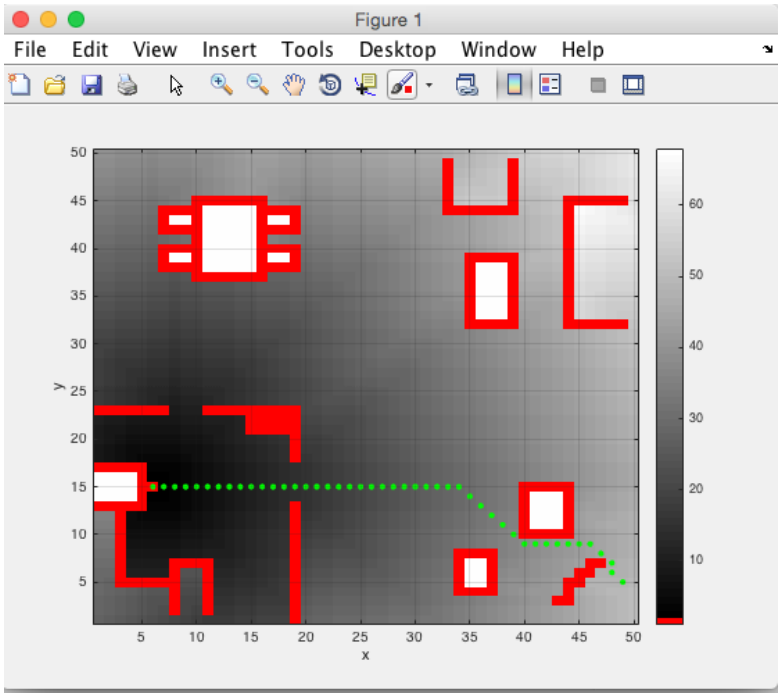


Figure 10 Shortest Path Generation

The module provides a mapped course, as seen in Figure 15, but in addition provides an array of [x,y] coordinates for each grid space which the robot passes through along its route. These grid points allowed for the collection of data.

6.3.2 Statistics Script

The function of the Statistics Script is to determine at what points in the array of points provided from the GeneratePath algorithm a turn occur. Once the turns are identified, the length of the straight away runs can be calculated and the angle of the turn can be measured.

```
for row = 1: rows-2

    % calculate slope from first point to second point
    p1 = [path1(i), path1(i,2)];
    p2 = [path1(i+1), path1(i+1,2)];
    slope1 = (p2(2)-p1(2)) / (p2(1)-p1(1));

    % calculate slope from second point to third point
    p3 = [path1(i+2), path1(i+2,2)];
    slope2 = (p3(2)-p2(2)) / (p3(1)-p2(1));

    % if slope is not equal, there is a corner
    if (slope1 ~= slope2)
        Y = ['There is a turn at ', num2str(p2)];
        disp(Y)
        %measure angle of corner
        angle =  atan2(abs(det([p3-p1;p2-p1])),dot(p3-p1,p2-p1));
        angle = angle *(180/pi);
        %record angle
        pathstats(i+1,3) = angle;
        YY = ['The angle is ', num2str(angle), ' degrees.'];
        disp(YY)
    %if slopes are equal, the points are on a straight line
    else
        X = ['This is a straight line from (', num2str(p1), ') to (', num2
        disp(X)
    end
    i = i+1;
end
```

Figure 11 Code for Angle Occurrence and Measurement

A turn was found by calculating the slope between Points A and B and points B and C. If the slopes were equal, there was not a turn at this time step. If the slopes were not equal, a turn occurred. If a turn occurred the angle of the turn was determined using the four-quadrant inverse tangent function, *atan2*. Figure 11 shows the angle finding and measurement loop.

At the each turn, the distance of the previous straight away was calculated. All calculations were stored in a matrix, displayed later in Section 6.5.

6.3.3 Graphical Data

The graphical data script was used simply to display the calculations from the Statistical Script. This section generated two bar graphs - one for the frequency and degree of turns, and one for frequency and distance traveled for straight aways.

```
%Bar Graph for Distance traveled
% Y Frequency Distance occurs
% X Distance traveled before turn
x = (pathstats(:,4));
[a,b]=hist(x,unique(x));
subplot(2,1,1)

bar(b,a)
title ('Frequency of Distance Traveled Before Turn')
xlabel('Distance Traveled') % x-axis label
ylabel('Frequency') % y-axis label
```

Figure 12 Graphical Data Code

Figure 12 shows the creation of a graph for the Frequency of Distance Traveled between Turns.

6.3.4 CallAll

The CallAll script allows the user to call the three previous scripts and enter the number of replications desired. The script compiles and saves the output provided from the other scripts.

6.4 Three Dimensional Simulation

The 3-dimensional simulation was planned to run in a similar manner to the 2D one - the simulation would begin with the robot arm within reach of the fetch item, the arm and gripper would be configured so that the object was grasped and the simulation would capture the final configuration of the arm and gripper. Several repetitions of the simulation were manually performed using the arrow keys to control the robot's arm and gripper movement.

The keyboard controls for the YouBot simulation in Vrep are:

Arrow keys for platform movement

QE/WS/AD keys for X/Y/Z directional arm movement

Space to toggle the gripper

The project team found that even a very slight change in the location of the fetch object required major configuration changes in each joint of the robot arm. Figure 13 and 14 show the change in arm configuration after moving the object duration after lifting the object several inches.

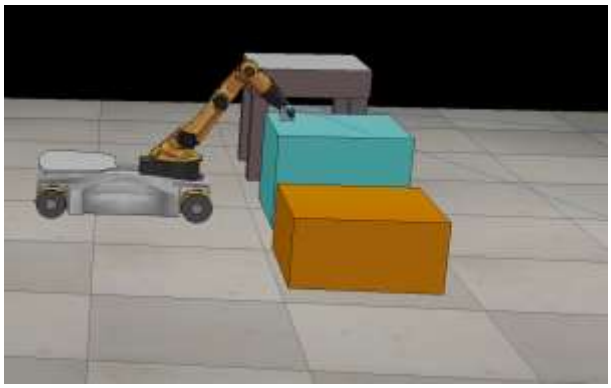


Figure 13 Arm Configuration

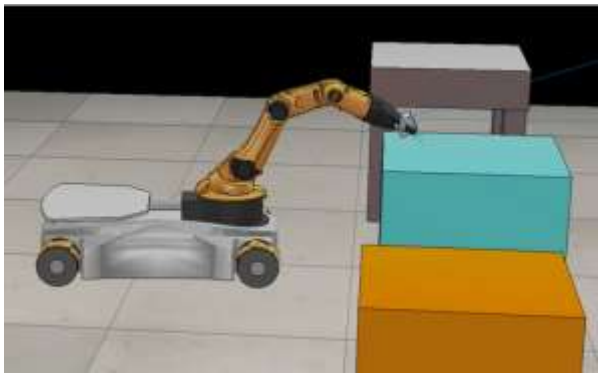


Figure 14 Arm Configuration Change

The project team found that precise control of the arm and gripped required a huge amount of patience and skill, even when using a physical control interface. Controlling the arm and gripper with the precision needed to grasp an object in three dimensions was ruled out as a feasible solution based on this experience.

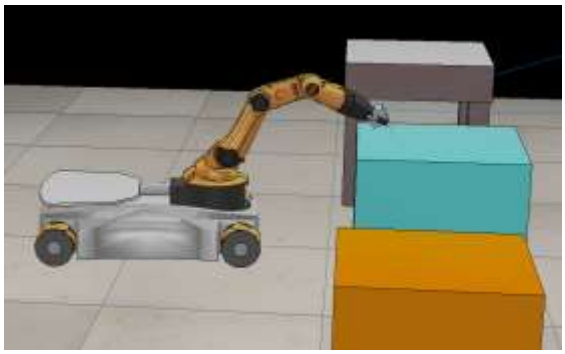


Figure 15 Object Grasping

6.5 2- Dimensional Simulation Results - Single Run

The 2D simulation was run to determine the level of control and number of commands necessary for control over the robot in order to perform domestic fetch and return tasks. Results will be shown to break down a single run of the simulation and to display a roll up of simulation runs.

In this single run, the start point selected was [49, 5] and the fetch point selected was [6, 15]. The occupancy map and D* shortest route (output of the GeneratePath module) from this path are shown in Figure 16.

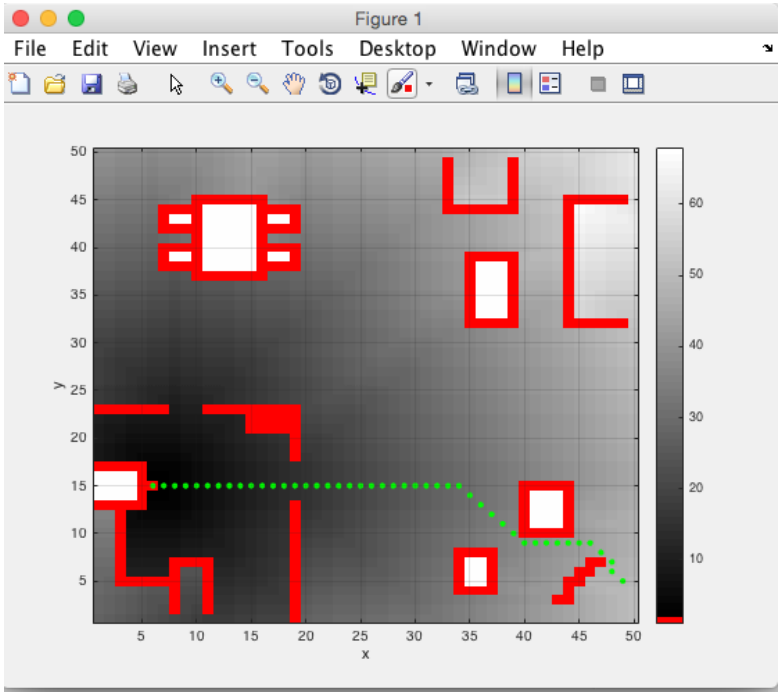


Figure 16 Path Generation

The array of grid points was then fed into the Statistical Analysis Module, where the points were analyzed to determine distances that the robot was moving forward, the number of turns which occurred and the degree of each turn. This data was stored in a matrix, as seen in Figure 17, in which Columns 1 and 3 are the robot location. Column 3 is the degree of a turn which occurred and Column 4 is the distance traveled for each straight away.

Editor - StatisticsScript.m

pathstats

21x4 double

	1	2	3	4	5
1	19	34	0	0	
2	19	33	0	0	
3	19	32	0	0	
4	19	31	0	0	
5	19	30	26.5651	4	
6	20	29	18.4349	0	
7	20	28	0	0	
8	20	27	0	0	
9	20	26	0	0	
10	20	25	0	0	
11	20	24	0	0	
12	20	23	0	0	
13	20	22	0	0	
14	20	21	0	0	
15	20	20	0	0	
16	20	19	0	0	
17	20	18	26.5651	10	
18	19	17	45	0	
19	18	18	18.4349	0	

Figure 17 Storage Matrix

This matrix was then stored and opened in the Graphical Data module where the number and distance of the straight aways and degree and quantity of the turns was calculated and displayed as a set of bar graphs. These graphs are displayed as Figure 18.

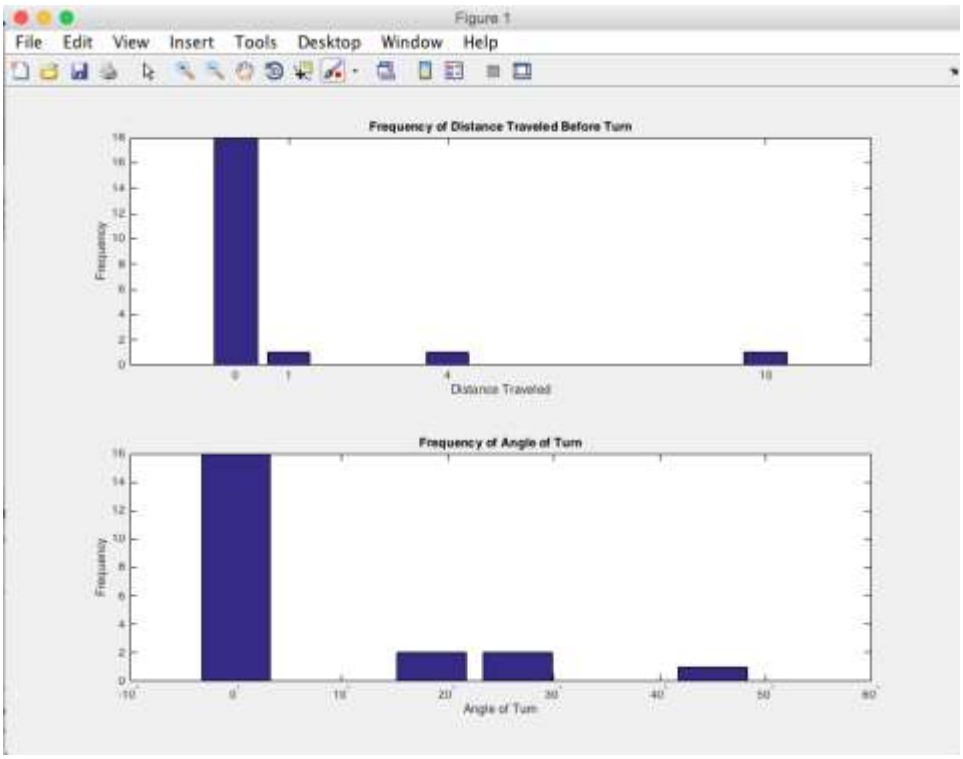


Figure 18 Graphical Output for One Run

A zero in the distance traveled graph indicated that the robot was still continuing along a straight path, as the distance was only considered a total at the end of each straight away. In the top Distance Traveled graph, we see that the majority of the distances which occurred were in either a 1 unit movement, 4 unit movement or 10 unit movement. The second graph, Frequency of Angle of Turn, shows each turns' degree. A zero in this graph means that a turn did not occur during that time step- ie the robot continued straight. We see that turns occurred in angles of approximately 18 degrees, 26 degrees and occasionally 45 degrees.

6.6 2-Dimensional Simulation Results - Roll Up

The GraphicalData Module allowed for multiple runs to be completed and for the data to be stored without having to stop the simulation between trials. The same process as that described in Section 6.5 occurred for each iteration in the Roll-Up of results. The below results reflect 50 runs of the simulation.

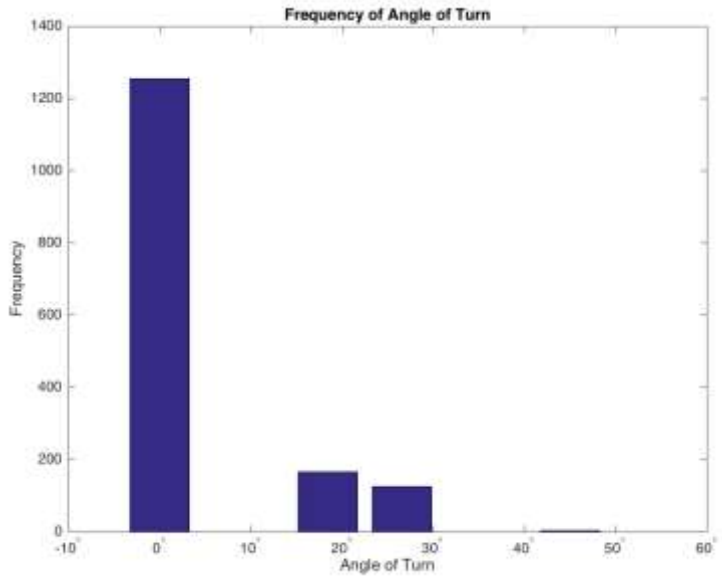


Figure 19 50 Run Graphical Angle Frequency

Figure 19 shows the angle of each turn completed over 50 runs of the simulation. Again, a zero in this graph reflects that no turn occurred at the given time step. We see that all of the turns completed fit into one of the three categories identified in the single run, Section 6.5. Almost 200 18 degree turns occurred, 156 26 degree turns and 21 45 degree turns.

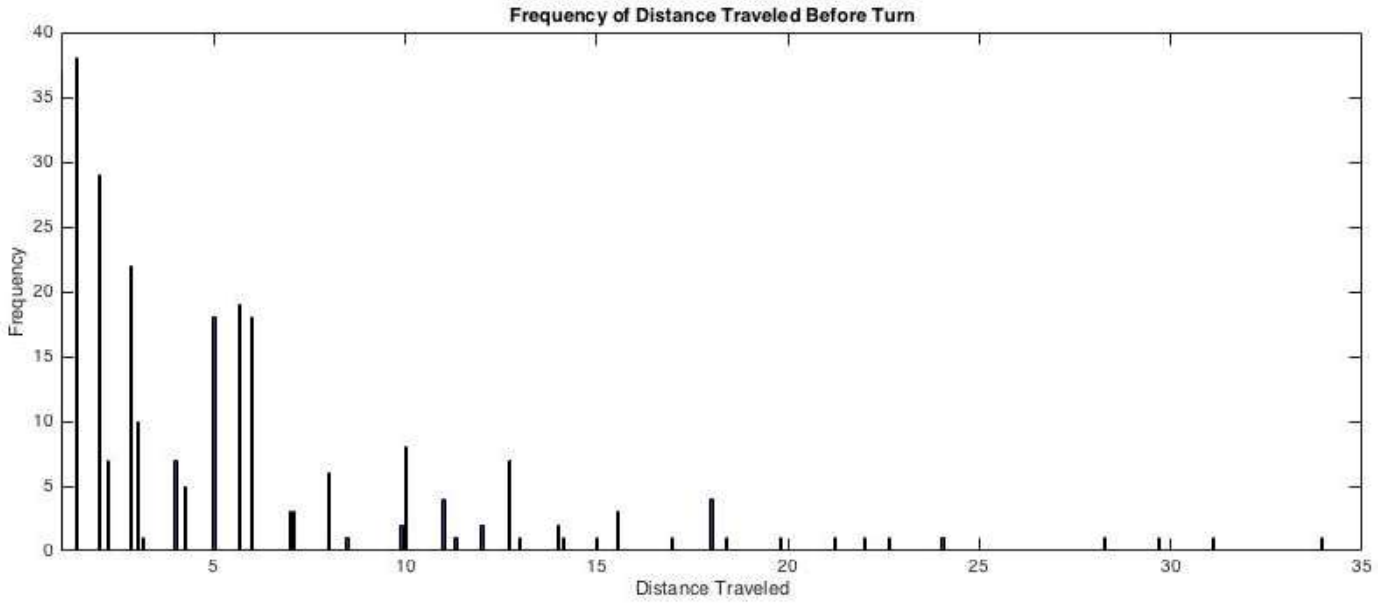


Figure 20 Distance Frequency for 50 Runs

Figure 20 shows the distance traveled per straight away. We see that among the straight aways, there is a slant towards short distances but no real trend in the distance the robot will travel before a turn must occur.

6.7 Simulation Conclusion

From these results, we can answer the question posed in Section 6.3. For navigation, 6 platform commands are needed. Turn left 18 degrees, turn left 2 degrees, turn right 18 degrees, turn right 2 degrees, start and stop (forward motion). From these commands, we believe the robot can be moved close enough to each feasible fetch object for the task to be completed.

In addition to navigational commands, it is recommended that a "Return to user" function is included for safety and ease of use. This would require 1 additional command.

From the 3-dimensional simulation, we determined that an alternative to manual control should be used to instruct the robot to grasp the object. One such alternative is the use of a vision sensor; this will be explored in detail in Section 6.8. This implementation would require two additional commands - "select" and "grasp."

6.8 Vision Sensor and Active Select

In the VREP simulation, the tip of the arm, where the grabber is, has a vision sensor and a ray type proximity sensor. The vision sensor is a very simple sensor that filters objects based on colors and patterns and returns the known regions with a variable that contains basic information such as the relative direction from the vision sensor as a region in a 2D plane orthogonal to the vision sensor and degrees of colors detected. The ray type proximity sensor is aligned with the vision sensor and returns the distance to the nearest renderable object in its direction. With these two sensors, the relative positions of the objects from the YouBot can be calculated using polar coordinates since the direction is given by the vision sensor and the distance away is given by the proximity sensor. Polar coordinates are best used because the movement range of the arm is in a rough hemisphere. Since only the regions of the objects is detected by the vision sensor, it is

difficult to determine the exact center and orientation of the detected objects and the current simulation cannot automatically pick up the detected objects. The simulation currently only can detect very simple colored shapes such as a small cube using simplistic color differentiation. Better object recognition methods exist, but due to limited robotics programming expertise, we have been unable to replicate them in this simulation.

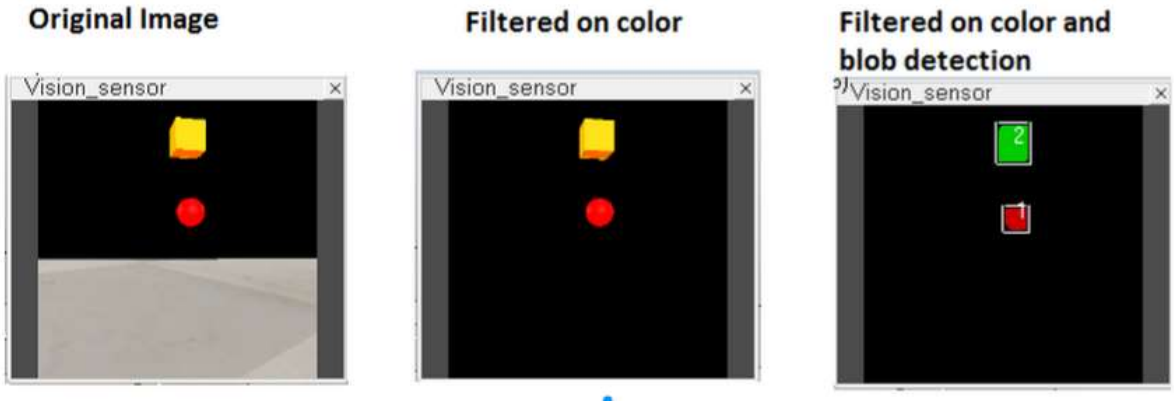


Figure 21 Vision Sensor Filtration

The intention is that each object can be detected and labeled with a number. Then the user can select a number to indicate which object they wish to pick up. If the distance indicated by the proximity sensor exceeds the arm range of the YouBot, the YouBot will have to be maneuvered closer.

6.9 Full Design Requirements Set

The requirements set detailed in Section 6.2 was revised after completing the simulation. A complete requirement set can be found below; requirements which were modified or appended as a result of the simulation are shown in *italics*.

R.0 The HFCS shall be operable without physical user input.

R.1 The HFCS shall direct the motion of a robot in 2 dimensions.

R.1.1 The HFCS shall direct the robot to start movement forward.

R.1.2 The HFCS shall direct the robot to stop motion forward.

R.1.3 The HFCS shall direct the robot to pivot left 18 degrees.

R.1.4 The HFCS shall direct the robot to pivot left 2 degrees.

R.1.5 The HFCS shall direct the robot to pivot right 18 degrees.

R.1.6 The HFCS shall direct the robot to pivot right 2 degrees.

R.2 The robot shall perform motion directed by HFCS.

R.2.1 The robot shall move forward upon command.

R.2.2 The robot shall stop forward movement upon command.

R.2.3 The robot shall pivot left upon command.

R.2.4 The robot shall pivot right upon command.

R.3 The HFCS shall allow the user to select an object in view of the camera.

R.4 The HFCS shall allow the user to command the system to grasp the object.

R.5 The HFCS shall direct the movement of the robot arm, via automated function.

R.5.1 The HFCS shall direct the arm to raise.

R.5.2 The HFCS shall direct the arm to lower.

R.5.3 The HFCS shall direct the hand to open.

R.5.4 The HFCS shall direct the hand to close.

R.6 The robot arm shall perform movement directed by HFCS.

R.6.1 The robot arm shall raise upon command

R.6.2 The robot arm shall lower upon command.

R.6.3 The robot arm shall open upon command.

R.6.4 The robot arm shall close upon command.

R.7 The HFCS shall not harm the user in any way.

R.7.1 The HFCS shall employ only non-invasive technology.

R.7.2 The HFCS shall weigh less than 25 lbs.

R.7.3 The HFCS shall be worn by an operator with head size min = 19 inches, max = 25 inches.

R.7.4 The HFCS shall not impair the operator's ability to move their limbs or neck.

R.7.5 The HFCS shall not impair the operator's ability move from one location to the next.

R.7.6 The robot shall employ automatic collision avoidance.

R.7.7 The robot shall employ an automatic stop function if signal is lost.

R.8 The HFCS shall provide flexibility for use for a variety of functions.

R.8.1 The robot shall have a total radius of 8 inches.

R.8.2 The robot shall be able to grasp an object up to 4 feet off the ground.

R.8.3 The robot shall be able to grasp an object up to 3 inches away from the edge of a table.

R.8.4 The robot shall be able to lift an object up to diameter 5 inches.

R.8.5 The robot shall be able to lift an object up to 5 lbs.

R.8.7 The robot shall be able to transport up to 25 lbs.

R.8.8 The HFCS shall operate for a minimum of 3 hours before recharging is required.

R.8.9 The robot shall operate for a minimum of 3 hours before recharging is required.

R.8.10 The HFCS shall broadcast signal to 20 ft radius of system.

R.9 The HFCS shall operate in real-time.

R.9.1 The HFCS shall direct the robot within 2 seconds of user command.

R.9.2 The robot shall perform motion within 1 second of HFCS direction.

R.9.3 The HFCS shall correctly identify 75% of user commands.

R.10 The HFCS shall execute a "Return to user" function.

7.0 Design of System

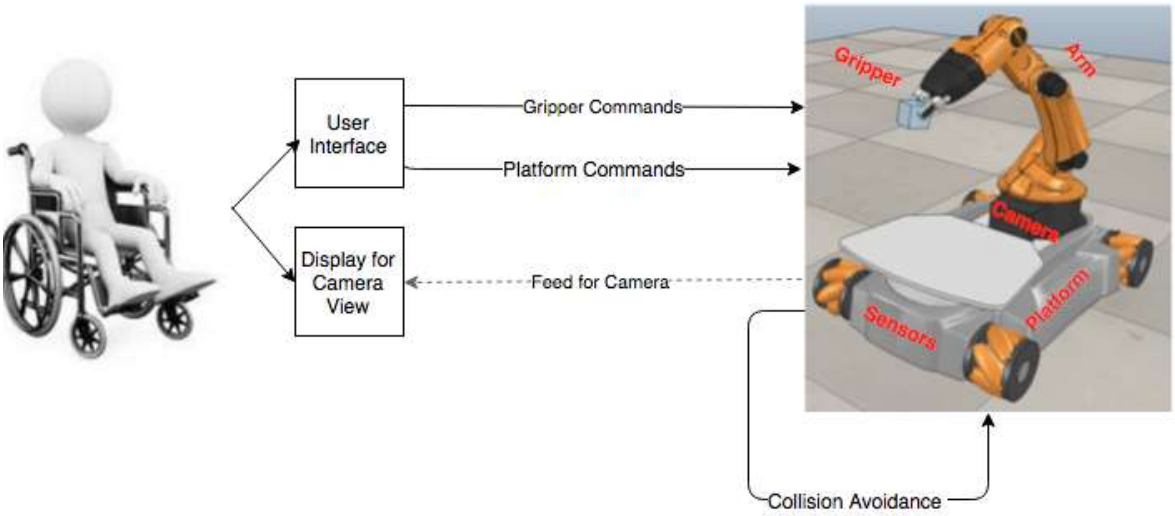


Figure 22 Design Independent Diagram

Figure 22 provides a design independent view of the system. The system has 4 major components, as defined below:

- Collision Avoidance Feedback Loop - utilize sensors mounted on robot to avoid collisions with stationary objects.
- User Interface - comprised of interface method to allow the use to command the robot
- Display for Camera View - Projection of the view of a camera mounted on the robot to allow for active select as described in
- Command Interface - translates the user command into command for robot.

The following Section 8 will analyze three hands-free interface methods to determine the optimal method for the system. Section 7.1 will introduce each of the potential interface methods.

7.1 Interface Alternatives

Several interface alternatives exist which fit all requirements for the system. These alternatives are brain-computer interface (BCI), eye movement tracking, and voice command. Similar alternatives, such as muscle contraction detection, were not selected because these interfaces require physical movement which the targeted user group cannot provide.

7.1.1 Brain-Computer Interface (BCI)

A brain-computer interface is a system that connects the brain with an external machine that interprets the thoughts of the brain as commands. In a noninvasive brain-computer interface, an electrode is placed on the head of the user [10]. These electrodes detect the electrical activity of the brain, and an external machine uses these as commands. In an invasive brain-computer interface, the electrodes are placed in the grey matter of the brain. In order for the external machine to accurately interpret the electrical signals as input, the user has to calibrate it by repetitively thinking of the same thoughts.

The largest advantage to BCI technology is seen as its universal application- regardless of physical functioning, users will be able to interact with a BCI. The shortcoming of BCI is seen as a function of its technical readiness level (TRL). The low TRL contributes to a lack of usability.

Appendix A summarizes a large amount of information related to the mechanics and feasibility of the use of BCI interfaces, specifically EEG as a robotics control system. This research was performed by the project team throughout the course of the project to gain a deeper understanding of the potential of EEG technology.

7.1.2 Eye Tracking

In an eye-movement tracking system, a machine is attached on the user's head. This machine uses infrared to scan the pupil. The machine tracks the reflection of the eye as input. This machine requires calibration, but is more stable than brain-computer interface.

The advantage of the eye-tracking software is seen in its time-sensitivity and it's TRL. This means the alternative could be immediately integrated into the system. Eye-tracking software does pose a risk of false-positives, as a user's fixation with an image may be interpreted as selection.

7.1.3 Voice Commands

Voice commands use a software to interpret a user's verbal words into commands. When a user speaks, he/she emits vibrations in the form of waves and the software converts these waves to executable commands by using a built-in library of commands. Calibration is usually optional and is used for the system to be more accurate.

The advantage to voice command software is seen in the user's familiarity with the interface format. Voice command software does, however, struggle in environments with large amounts of background noise and requires some calibration time for maximum accuracy.

8.0 Method of Analysis

8.1 Utility Function

Analysis of alternative utility was driven by four weighted categories. These categories were decomposed into weighted measures as depicted in Figure X.

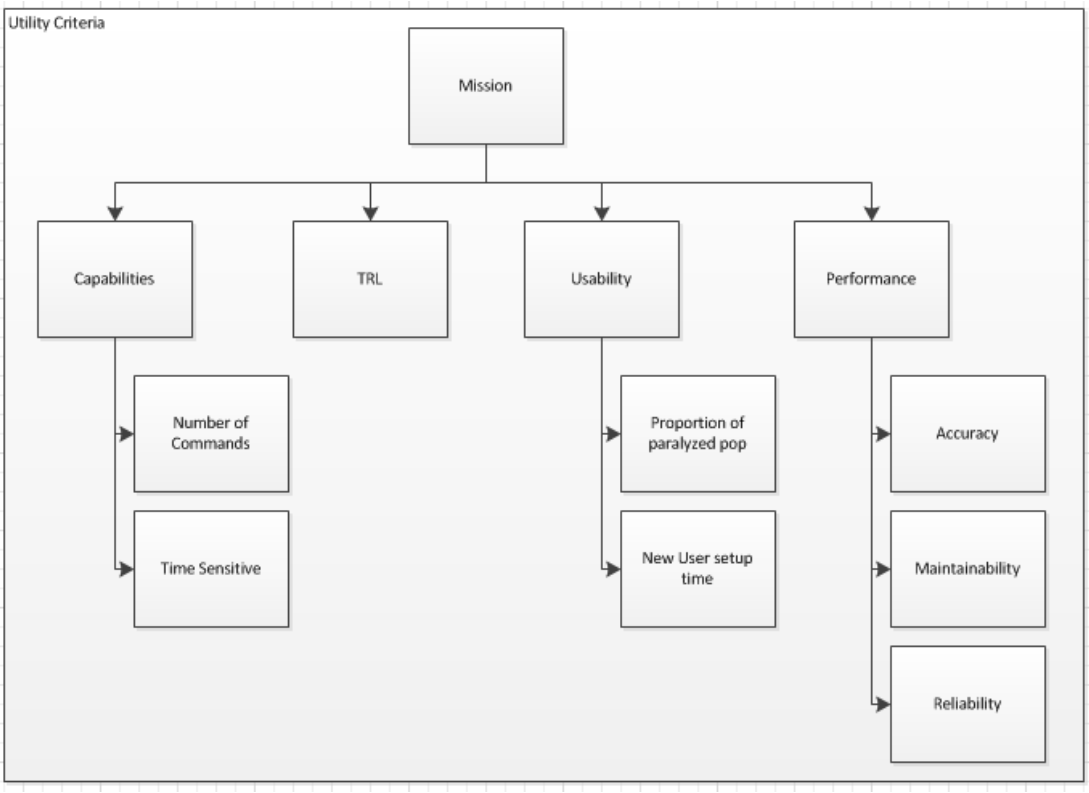


Figure 23 Category Breakdown

8.1.1 Category and Measure Definitions

Each category and its measures are defined below.

Capabilities - ability of the interface to meet critical mission requirements

Number of Commands- ability of the interface to differentiate between and execute 9 commands, where 9 is derived from the simulation as described in Section 6.0

Time Sensitivity - ability of the interface to perform in a time sensitive environment. Derived from researched lag times.

TRL- Technological Readiness Level; ability of interface to be utilized without substantial design/ testing time and costs.

Usability - Ease of use for user

User Population- Number of users capable of meeting performance requirements of the interface.

Set-Up time- Amount of time and cost of setting up/ calibrating the system for a new user.

Performance - Measure of interface accuracy, reliability and up-time

Accuracy - Ability of the interface to identify and execute the intended command

Reliability- Measure of interface durability

Maintainability - Derived from maintenance time required and calibration time.

8.1.2 Swing Weights

The swing weights method was used to determine an overall weight for each measure. Weights are shown in Table I.

Table 4 Weight Breakdown

Goal	Category Weight	Measures	Measure Weight	Tot Weight
Capabilities	0.5	# of Commands	0.7	0.35
		Time Sensitivity	0.3	0.15
TRL	0.1	TRL	1	0.14
Usability	0.2	User Population	0.75	0.15
		New User Time	0.25	0.05
Performance	0.2	Accuracy	0.4	0.08
		Maintainability	0.2	0.04
		Reliability	0.4	0.08

Each alternative was scored based on reviewed journal articles and other available data [11] [12] [13]. The utility score was calculated via the equation $\Sigma w_i * s_i$, where w_i is the measure weight

and s_i is the alternative's score for the measure. Figure <> shows the scores received for each alternative. The Logical Decision software [] was used to perform this analysis.

8.2 Utility Analysis Results

Voice command (0.812) ranks the highest, followed by eye-tracking (0.804), then BCI (0.536). Utility is shown in Figure 24.

Figure 24 Utility Calculation

Figure 24 also shows the breakdown of Utility by category. We see that the high utility score of the Voice Command option is primarily created by its capabilities score. We see that the BCI alternative received a comparable score in the Usability category but falls behind in the Capabilities and Performance categories.

8.3 Cost vs Utility Analysis

An estimation of the cost of each alternative was generated using COTS software and hardware. The estimated cost of each alternative can be seen in in Table 5.

Table 5 Cost of Alternatives

	Alternative		
	BCI	Eye Tracking	Voice Command
Display	150	300	150
Equipment	600	150	200
Software, if not included with equip	0	0	100
Set-Up Total	750	450	450
Training Hrs	20	5	0.5
Training Labor	5	0	0
Personal Time Cost	20	20	20
Labor Cost	50	0	0
Learning Cost	650	100	10
Yearly Calibration T	91.25	8.67	0
Yearly Calibrartion S	1825	173.33	0

These costs were placed into categories Set, Learning and Calibration. The totals were graphed by alternative for one year to determine where major costs are incurred. This can be seen in Figure 25.

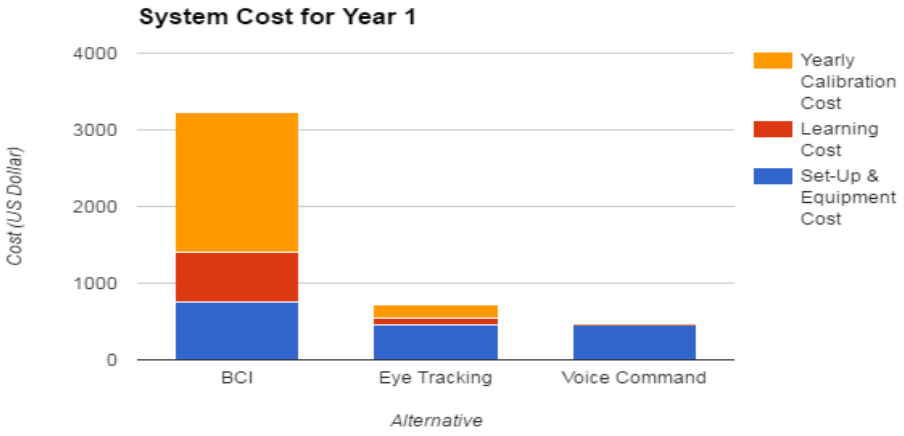


Figure 25 Cost For Year 1

We notice that the BCI alternative incurs a very high cost because of the yearly calibration cost, even for just the first year. The Cost vs Utility for each alternative is shown in Figure 26. The higher utility does not correlate with a higher price for the alternative.

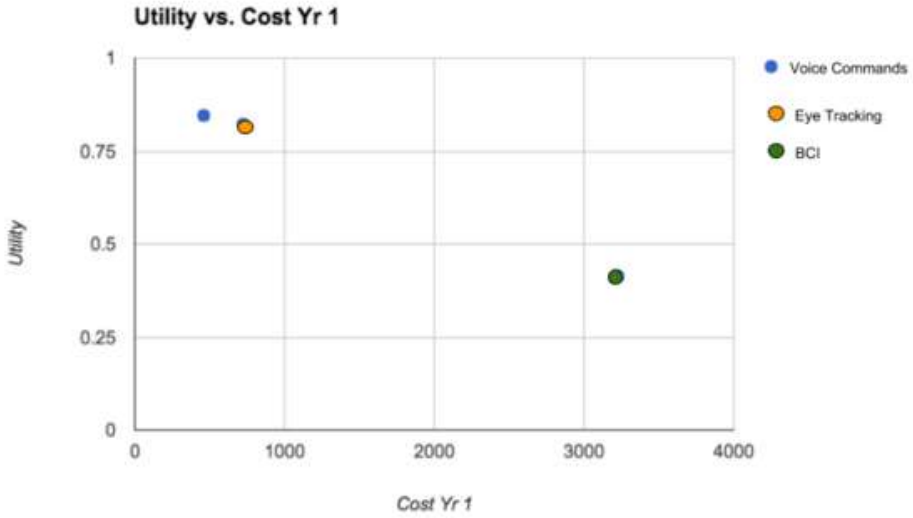


Figure 26 Utility vs Cost Yr 1

Figure 26 shows that the Voice Command option offers both the highest utility and the lowest cost of the three alternatives analyzed.

9.0 Recommendations

From the results of the simulation and the trade-off analysis, the project team recommend that a voice command based interface is used to control the motion of a robot to perform fetch and return tasks. The voice command method exhibited the highest utility of the methods analyzed and is available at the lowest cost.

In addition, the project team recommends that several automated features are included in the design.

1) Automated collision avoidance
2) Return to user feature
3) Automated grasping

10.0 Business Plan

According to a 2014 New York Times article, 1.3 million new caregivers will be needed to fill growing demand in the United States within the next decade. The national average wage for caregivers is about $20 per hour and currently 75% of American caregivers are paid by either Medicare or Medicaid. The low wages and grueling work required of caregivers makes it unlikely that this gap will be filled. In addition, the fact that government subsidized programs fund the majority of these positions makes it unlikely that the hourly wage of caregivers will increase to meet growing demand - it is more likely that middle class families will simply do without the additional care.

The robotic aid for object relocation as described in this report is not intended to replace the role of a full- time caregiver in the life of a severely paralyzed person, but would rather serve as as supplement, especially in low activity hours. The robotic aid would allow a paralyzed person to regain a degree of independence and increase their quality of life, while reducing the number of hours which a caregiver must be present.

Our design would be primarily marketed to in-home care agencies, who could determine the best ratio of caregiver interaction to robotic aid for a specific patient. This would ensure that the person still receives the attention and care that they need, while providing cost savings.

A unit for purchase is defined as a robotic aid and interface, sold together, ready to use. The price per unit would be $4,500.

Table 6 shows the anticipated expenses for the business.

Table 6 Business Plan Cost Breakdown

Non Reoccuring			**Reoccuring**	
Consulting	180000		Software	10000
Branding and Website	3000		Rent and Utilities Monthly	4000
Business Entitity and Pe	2000		Labor Management	140000
Office Equipment	10000		Labor Employee	360000
TOTAL	195000		YEARLY	504000

Our business would break even after the sale of 300 units. An expected sales estimate would be to sell 100 units per year. At that rate, our business would break even in approximately 3.5 years. An optimistic sales rate would anticipate that 200 units would be sold for the first two years before plateauing to 100 units per year. At the optimistic rate, the business would break even after a year and a half.

11.0 Acknowledgements

The project team would like to thank our sponsors at General Dynamics Mission Systems – Mike DiMichele and Nathan Cushing. In addition, we would like to thank Professor Sherry and Mr. Bahram Youseffi.

Appendix A References

[1] "Human Factors and Medical Devices." U.S. Food and Drug Administration. U.S. Food and Drug Administration, n.d. Web. 19 Nov. 2015.

[2] B. Shneiderman, 'The limits of speech recognition', *Communications of the ACM*, vol. 43, no. 9, pp. 63-65, 2000.

[3] Eyetracking.com, 'Powerful eye tracking software developed for researchers', 2015. [Online]. Available: http://www.eyetracking.com/Software. [Accessed: 04- Nov- 2015].

[4] Y. Hotta and K. Ito, 'EMG-based detection of muscle fatigue during low-level isometric contraction: Effects of electrode configuration and blood flow restriction', *2011 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, 2011.

[5] Google Self-Driving Car Project, 'Google Self-Driving Car Project', 2015. [Online]. Available: https://www.google.com/selfdrivingcar/how/. [Accessed: 09- Dec- 2015].B. Smith, "An approach to graphs of linear forms (Unpublished work style)," unpublished.

[6] J. Wolpaw and D. McFarland, 'Control of a two-dimensional movement signal by a noninvasive brain-computer interface in humans', *Proceedings of the National Academy of* Sciences, vol. 101, no. 51, pp. 17849-17854, 2004.

[7] "Prevalence of Paralysis." Christopher & Dana Reeve Foundation. The Reeve Foundation Paralysis Resource Center, n.d. Web. 19 Nov. 2015.

[8] Coppeliarobotics.com, 'V-REP User Manual', 2015. [Online]. Available: http://www.coppeliarobotics.com/helpFiles/. [Accessed: 09- Dec- 2015].

[9] "YouBot." YouBot Store. Kuka, n.d. Web. 19 Nov. 2015.

[10] R. Detry, 'TRS: An Open-source Recipe for Teaching (and Learning) Robotics with a Simulator', Ulgrobotics.github.io, 2015. [Online]. Available: http://ulgrobotics.github.io/trs/project.html. [Accessed: 09- Dec- 2015]. G. R. Faulhaber, "Design of service systems with priority reservation," in Conf. Rec. 1995 IEEE Int. Conf. Communications, pp. 3–8.

[11] Coppeliarobotics.com, 'Child scripts', 2015. [Online]. Available: http://www.coppeliarobotics.com/helpFiles/en/childScripts.htm. [Accessed: 09- Dec- 2015].

[12] Coppeliarobotics.com, 'Path planning', 2015. [Online]. Available: http://www.coppeliarobotics.com/helpFiles/en/pathPlanningModule.htm. [Accessed: 09- Dec- 2015].

[13] Coppeliarobotics.com, 'Vision sensors', 2015. [Online]. Available: http://www.coppeliarobotics.com/helpFiles/en/visionSensors.htm. [Accessed: 09- Dec- 2015].

[14] "Prevalence of Paralysis." Christopher & Dana Reeve Foundation. The Reeve Foundation Paralysis Resource Center, n.d. Web. 19 Nov. 2015.

[15] "Human Factors and Medical Devices." U.S. Food and Drug Administration. U.S. Food and Drug Administration, n.d. Web. 19 Nov. 2015.

[16] "YouBot." YouBot Store. Kuka, n.d. Web. 19 Nov. 2015.

[17] Graham, Judith. "A Shortage of Caregivers." New York Times [New York] 26 Feb. 2014. New York Times. 26 Feb. 2014. Web. 2 Mar. 2016.

[18] "Care-O-bot 3." Fraunhofer Institute for Manufacturing Engineering and Automation. Fraunhofer Institute, n.d. Web. 19 Nov. 2015.

[19] "Home Use Devices Initiative." Medical Devices. U.S. Food and Drug Administration, 17 Apr. 2015. Web. 2 Mar. 2016.

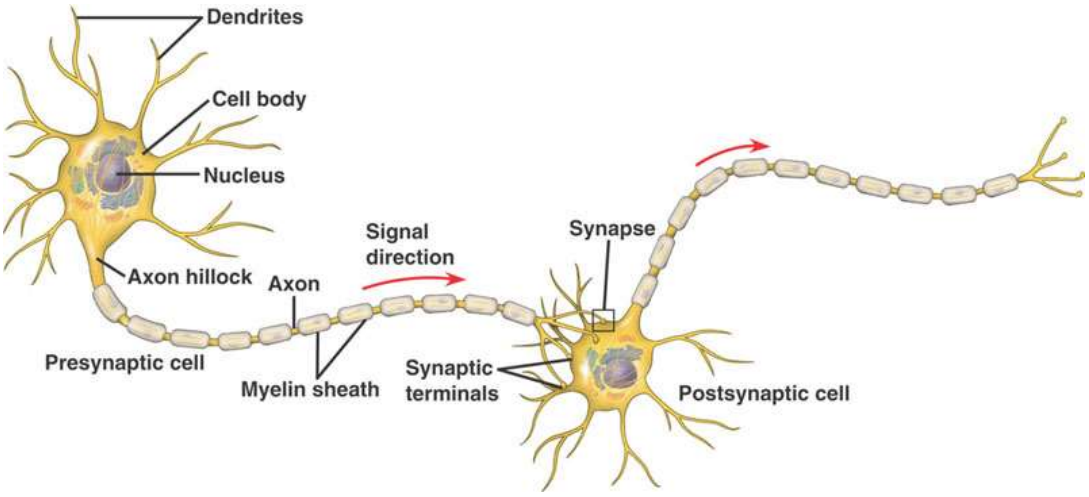
[20] United States. U.S. Census Bureau. Population Estimates and Projections. An Aging Nation: The Older Population in the United States. By Jennifer Ortman and Victoria Velkoff. May 2014. Web. 2 Feb. 2016.

Appendix B Design of Hands-Free Controller with BCI Interface

1.1 brain machine interfaces

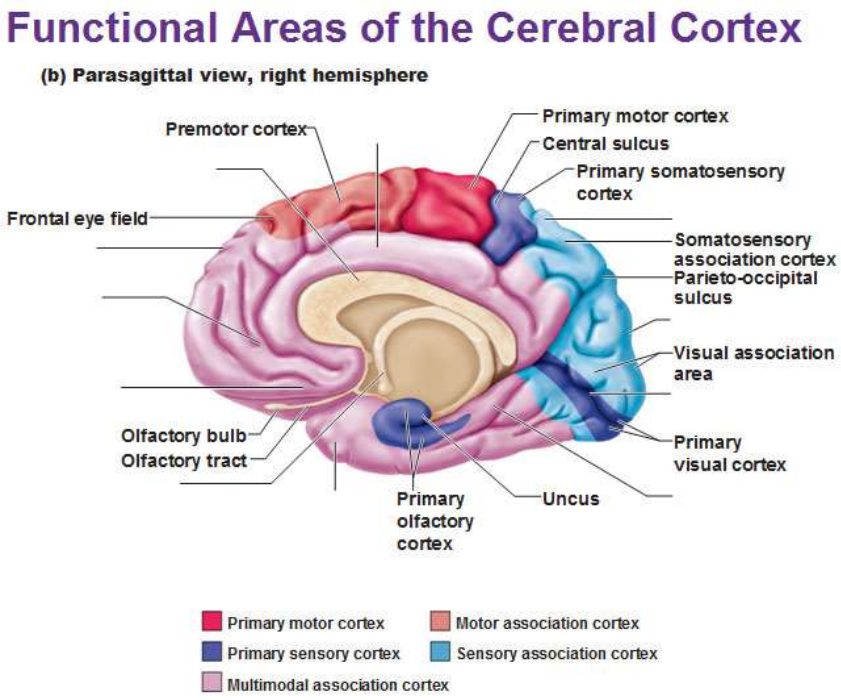
Brain-machine interfaces are very much still a developing technology. The ability to begin research into brain-machine interfacing has been made possible by advances in machine learning, computing power and our understanding of the human brain, however, this technology is still at a technological readiness level of 3.

The human brain is made up of hundreds of millions of neurons. Neurons are nerve cells which can process and transfer information, when "excited". Neurons enter an excited state when they are connected to other neurons at specific transfer points referred to as synapses. These chains of excited neurons, referred to as neural networks, produce electrical signals and heat when transferring information. Information is transferred between the neurons on chains called axoms. Figure 4 shows an axon connecting two neurons at synapses, and an electrical signal being transferred along the axon.



The electrical signal produced by the transfer of information can be sensed on the outermost layer of the brain, referred to as the cortex. Different segments of the brain map to different functions. Figure 5 maps the segments of the brain to their function, for example, the top left

segment, shaded red and labeled the primary motor cortex is responsible for the command messages controlling the physical motion of the body.



This modern understanding of the brain as an electrical control system has provided a crucial component to the development of brain-computer interfaces. Because the electrical signals commanding the physical control of the human body are measurable from the outer cortex of the brain, it is reasonable that these signals can be used to control a mechanical device.

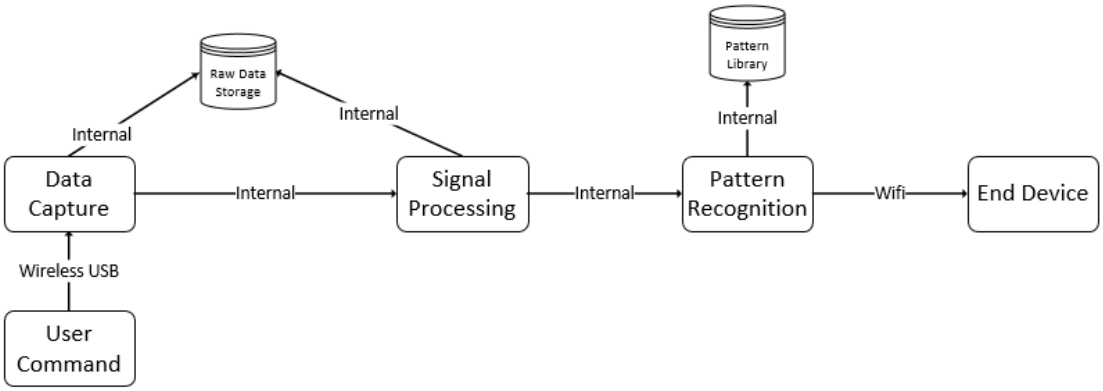
1.2 Electroencephalography

A normally functioning brain communicates by transmitting a series of electrical signals, referred to as impulses. This electrical activity can be measured and recorded through a process called electroencephalogy (EEG).

The waveforms shown in Figure 6 allow scientists and doctors to categorize the signals as a step towards identifying the thought that mapped to the waveform. Brain waves, as shown in the EEG output, usually fall into one of 4 major categories- alpha, beta, theta and delta waves. Each major category is distinguished by the frequency at which the brain waves are transmitted and have

been mapped to specific emotions or behaviors. For example, beta waves are commonly associated with anxiety. While the basic waveforms seen in an EEG frequently allow doctors to diagnose neurological disorders, the EEG also records bursts of energy, responses to stimuli and unique waveform.

Advancements in data science and machine learning have made huge advancements in our ability to map specific motor controls to their respective brain signal output. The ability to identify the wave form that maps to a specific control, “look down” for example, gives us the ability to create a control system based on conscious thought.



1.3 Technology

Until very recently, brain activity could only be measured through invasive procedures. Two procedures were available to retrieve this data. In the most invasive procedure, a chip in the grey matter of the brain; in the partially invasive procedure, a chip was placed in the skull but on the outside of the brain. While these procedures yielded very accurate data, the invasive and potentially dangerous nature of the procedure prevented the advancement of EEG research.

However, recently a number of noninvasive methods to measure brain signals have been developed. The most common of these methods is seen in the medical profession, where a series of individual sensors are applied to the outside of the skull, as shown in Figure 7.

This method, while invasive, is still labor intensive and requires a large amount of preparation and calibration time. An alternative to the sensor method has been developed in the form of the EEG headset. This headset is comprised of an adjustable plastic frame supporting a series of sensors. While the headset does not match the level of data accuracy seen in invasive procedures, the data is frequently sufficient for non-medical purposes. As this technology continues to expand, it is expected that the accuracy of non-invasive headsets will improve.

1.4 Current Uses

Research into EEG began during World War II in an effort to evaluate fighter pilots for pre-existing neurological diseases. Today, the majority of the applications of EEG technology and leading EEG research remain in medical fields. EEG tests are regularly prescribed in an effort to evaluate a patient for risk of seizure disorders such as epilepsy. EEG tests can also reveal neurological disease creating sleep or stress disorders. A current study at Massachusetts General Hospital is being performed to evaluate the use of EEG to monitor the brain function of patients under anesthesia or in a coma.

The increased availability of EEG headsets has lead to research into uses for EEG data in commercial fields. The study of neuromarketing focuses on the consumer's conscious and subconscious reaction to a product. The application of EEG to this field has allowed researchers to assign quantitative values to a user's subconscious reactions. A hallmark study of this field compared a consumer's reaction to Coke versus Pepsi. In a blind taste test, reactions and EEG data yielded very little difference between the two products. In a second test, the users were able to see the packaging of the beverage and results showed a strong preference towards Coke products. This study showed that marketing and a consumer's perception of a product can affect the user's subconscious reaction to the product.

In military organizations, the application of EEG technology has the potential to reduce the user workload in command and control systems. A current project undertaken at University of California, Irvine is investigating the potential to revolutionize the way soldiers communicate through "Imagined Speech." Because soldiers are trained to communicate in very concise, specific terms, researchers believe these terms may create specific patterns in EEG output. The

ability to recognize short key terms for communication is the same capability which will allow for EEG data to be used in a control system.

1.5 Emotiv

Emotiv is a highly accredited bioinformatics company based in southern California. Their main goal is to promote the research of the human brain and stimulate public interest in bioinformatics. In addition to performing research, they offer many varieties of EEG headset products. Particularly, the standard EPOC headset comes equipped with fourteen passive EEG channels, wireless capabilities, and a software development kit created by the company. The EPOC+ variation includes all of these components, among bluetooth, and nine inertial sensors. For the purpose of this project, we purchased the standard EPOC headset with the raw data package. The raw data option will provide further insight to the finite signals that are emitted from the brain. This will enable a more accurate diagnosis of the flaws involved with utilizing EEG as a control system.

Appendix C Matlab Simulation Code

GeneratePath.m

```
%uncomment line2 for first run after restarting matlab
%run('~/desktop/trs/matlab/startup_robot.m')
load ('room2.mat') % open room file.
%Generate feasible start locations
start1 = [2, 45];
start2 = [25, 45];
start3 = [48, 48];
start4 = [15, 5];
start5 = [49, 5];
start6 = [25,32];
start7 = [32, 16];
start8 = [18, 35];
start9 = [30, 28];
start10 = [7, 20];

%Randomly select a start location
s = round (rand(1) * 10);
%initialize variable
start = [25, 25];
%Set start to selected location
switch s
   case 1
      start = start1;
   case 2
      start = start2;
   case 3
      start = start3;
   case 4
      start = start4;
   case 5
      start = start5;
   case 6
      start = start6;
   case 7
      start = start7;
   case 8
      start = start8;
   case 9
      start = start9;
```

```
    case 10
        start = start10;
end
disp ('start location is:')
disp (start)
```

StatisticsScript.m

```
%initialize variable
goal = [25, 25];
% Generate fetch locations
fetch1 = [6, 15];
fetch2 = [5, 6];
fetch3 = [18, 20];
fetch4 = [12, 37];
fetch5 = [36, 43];
fetch6 = [40, 35];
fetch7 = [45, 13];
fetch8 = [36, 3];
fetch9 = [39, 13];
fetch10 = [12, 46];

%Randomly select a start location
f = round (rand(1) * 10);

%Set start to selected location
switch f
    case 1
        goal = fetch1;
    case 2
        goal = fetch2;
    case 3
        goal = fetch3;
    case 4
        goal = fetch4;
    case 5
        goal = fetch5;
    case 6
        goal = fetch6;
    case 7
        goal = fetch7;
    case 8
        goal = fetch8;
```

```
    case 9
        goal = fetch9;
    case 10
        goal = fetch10;
end
disp ('fetch location is')
disp (goal)

ds = Dstar(world)
ds.plan(goal)
path1 = ds.path(start);
ds.path(start)

[rows, columns] = size(path1);
 i = 1;
 %create new variable to record statistics
 pathstats = path1;

for row = 1: rows-2

    % calculate slope from first point to second point
    p1 = [path1(i), path1(i,2)];
    p2 = [path1(i+1), path1(i+1,2)];
    slope1 = (p2(2)-p1(2)) / (p2(1)-p1(1));

    % calculate slope from second point to third point
    p3 = [path1(i+2), path1(i+2,2)];
    slope2 = (p3(2)-p2(2)) / (p3(1)-p2(1));

    % if slope is not equal, there is a corner
    if (slope1 ~= slope2)
        Y = ['There is a turn at ', num2str(p2)];
        disp(Y)
        %measure angle of corner
        angle =  atan2(abs(det([p3-p1;p2-p1])),dot(p3-p1,p2-p1));
        angle = angle *(180/pi);
        %record angle
        pathstats(i+1,3) = angle;
        YY = ['The angle is ', num2str(angle), ' degrees.'];
        disp(YY)
    %if slopes are equal, the points are on a straight line
    else
        X = ['This is a straight line from (', num2str(p1), ') to (', num2str(p3), ')'];
```

```
        disp(X)
    end
    i = i+1;
end

beginline = [pathstats(1), pathstats(1,2)];
endline = [0,0];
for row = 1:rows-2
    if (pathstats(row, 3) ~= 0)
        endline = [pathstats(row), pathstats(row,2)];
        d = norm(endline - beginline);
        XX = ['The line is ', num2str(d), ' units long'];
        disp (XX);
        pathstats(row, 4) = d;
        beginline = [pathstats(row+1), pathstats(row+1,2)];
    end
end
endline = [pathstats(size(pathstats,1)), pathstats(size(pathstats,1),2)];
lastline = norm(endline- beginline);
pathstats(size(pathstats,1), 4) = lastline;
```

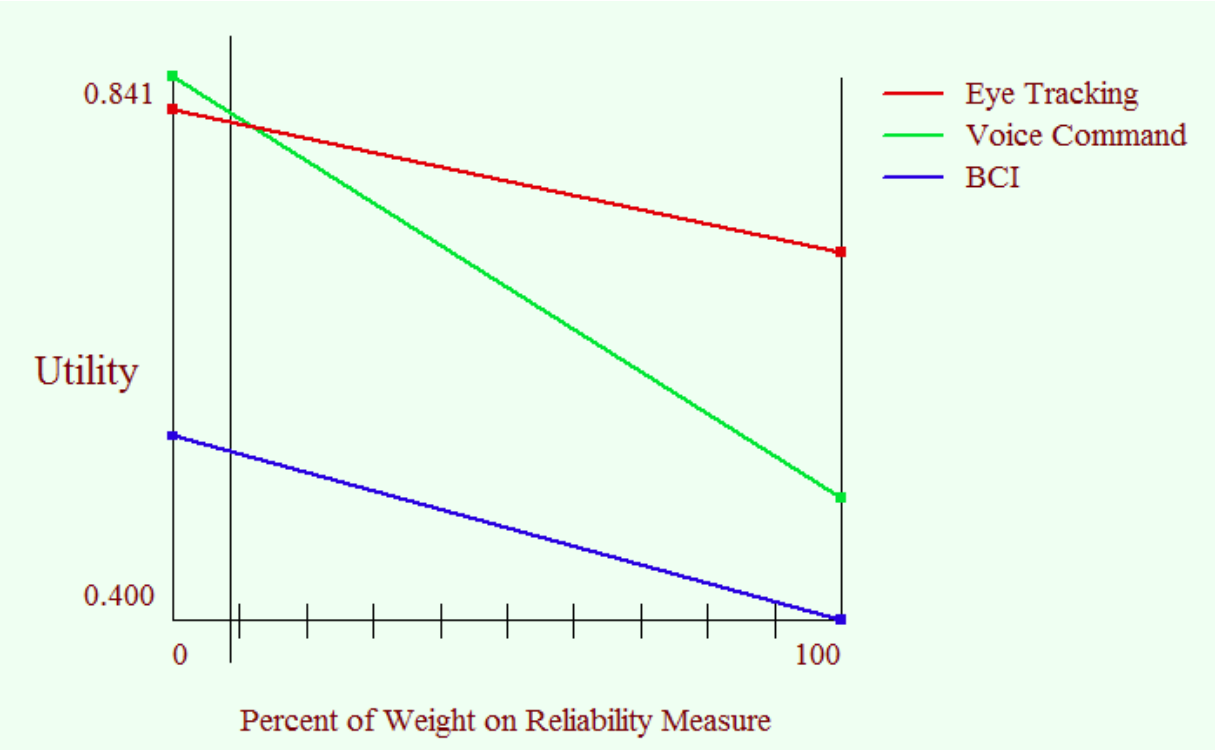
GraphicalData.m

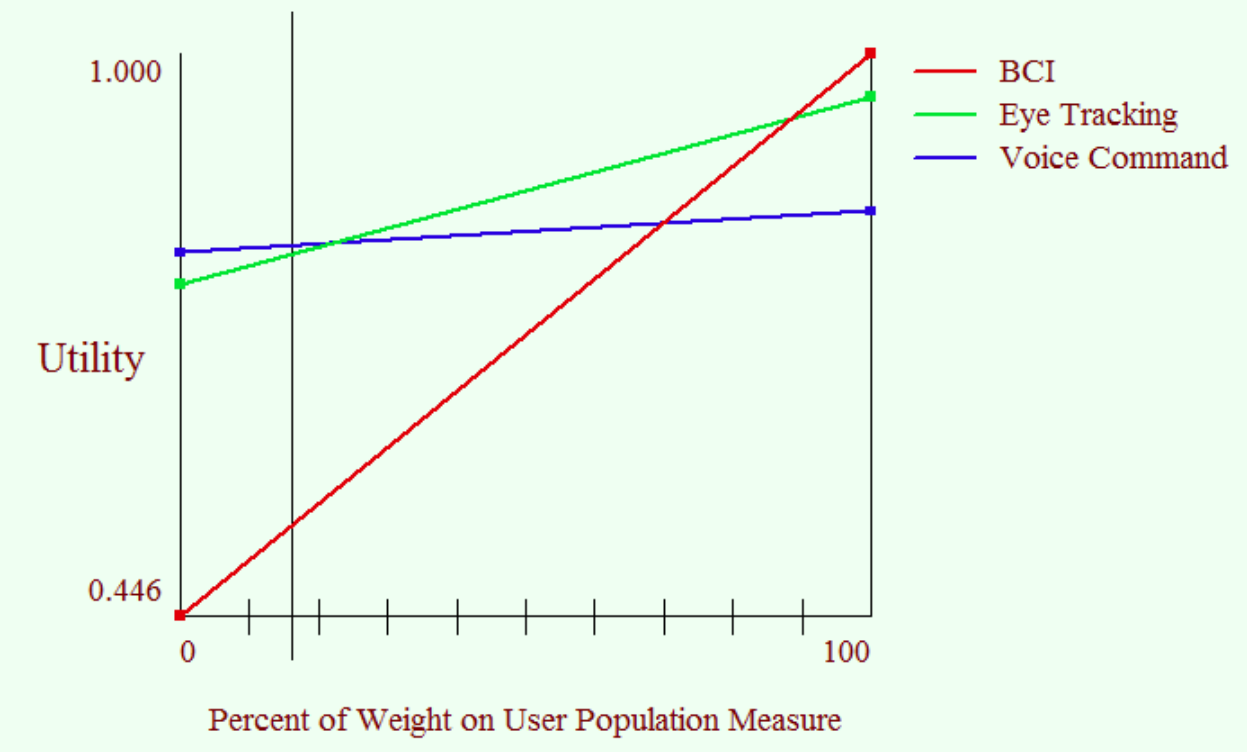
```
%figure(2)
%Plot Path
%subplot(2,2,1)
%ds.path(start)

%Bar Graph for Distance traveled
% Y Frequency Distance occurs
% X Distance traveled before turn
x = (pathstats(:,4));
[a,b]=hist(x,unique(x));
subplot(2,1,1)

bar(b,a)
title ('Frequency of Distance Traveled Before Turn')
xlabel('Distance Traveled') % x-axis label
ylabel('Frequency') % y-axis label


% Bar Graph for Degree of Turn
x = (pathstats(:,3));
[a,b]=hist(x,unique(x));
subplot(2,1,2)
figure(1)
bar(b,a)
title ('Frequency of Angle of Turn')
xlabel('Angle of Turn') % x-axis label
ylabel('Frequency') % y-axis label
ax = gca;
ax.XAxis.TickLabelFormat = '%g^\\circ';
```

CallAll.m

```
run ('~/desktop/GeneratePath.m')
run ('~/desktop/StatisticsScript.m')
run ('~/desktop/GraphicalData.m')
```

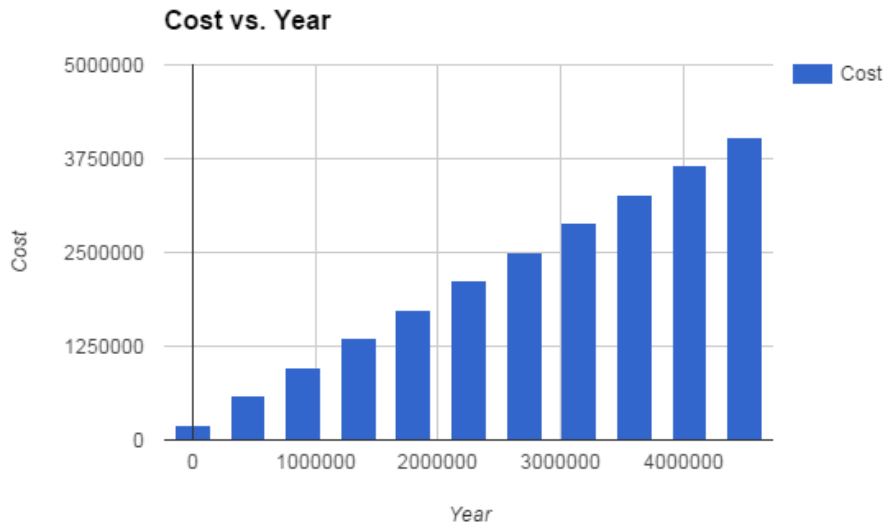
Appendix D Sensitivity Analysis

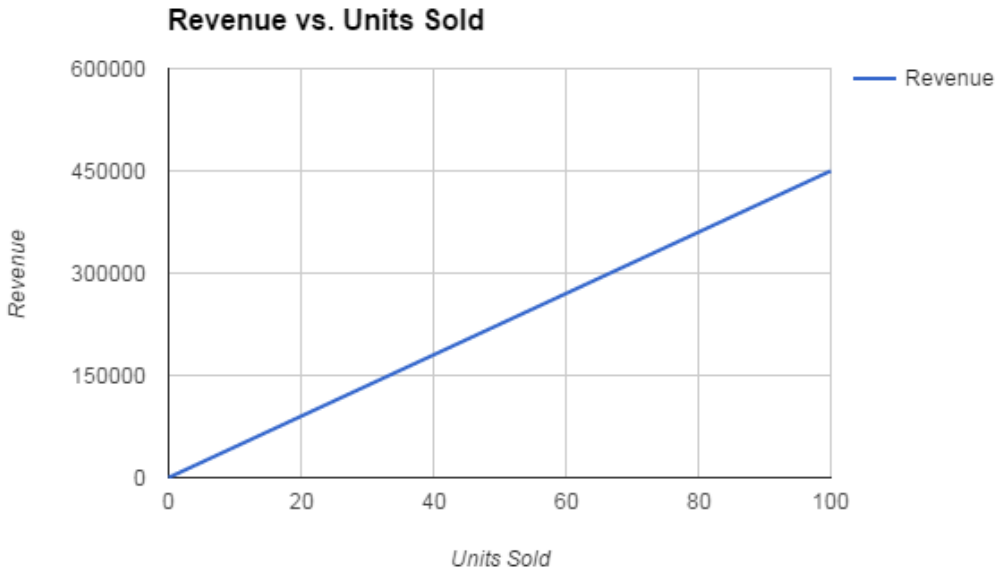



1.000
Utility
0.446
0
100
Percent of Weight on User Population Measure
BCI
Eye Tracking
Voice Command

Appendix E Business Case Cost Analysis

Cost Break Down			
Non Reoccuring		**Reoccuring**	
Consulting	180000	Software	10000
Branding and Website	3000	Rent and Utilities Monthly	4000
Business Entitity and Permits	2000	Labor Management	140000
Office Equipment	10000	Labor Employee	240000
Down Payment on lease	10000		
TOTAL	205000	YEARLY	384000




Revenue vs. Units Sold
600000
450000
300000
150000
0
Revenue
0
20
40
60
80
100
Units Sold
Revenue

Appendix F Trade-Off Analysis Data

Goal	Category Weight	Measures	Measure Weight	Total Weight
Capabilities	0.5			
		Number of Commands	0.7	0.35
		Time Sensitivity	0.3	0.15
TRL	0.1			
		TRL	1	0.1
Usability	0.2			
		User Population	0.75	0.15
		New User Time	0.25	0.05
Performance	0.2			
		Accuracy	0.4	0.08
		Maintainability	0.2	0.04
		Reliability	0.4	0.08

		ALTERNATIVES					
		BCI		Eye Tracking		Voice Commands	
Attribute	Weight	Score	Weighted Score	Score	Weighted Score	Score	Weighted Score
Number of Commands	0.35	3	1.05	8	2.8	10	3.5
Time Sensitivity	0.15	3	0.45	9	1.35	7	1.05
TRL	0.1	2	0.2	9	0.9	9	0.9
User Population	0.15	10	1.5	9.2	1.38	8.1	1.215
New User Time	0.05	2	0.1	7	0.35	7	0.35
Accuracy	0.08	5	0.4	7	0.56	8	0.64
Maintainability	0.04	3	0.12	8	0.32	10	0.4
Reliability	0.08	4	0.32	7	0.56	5	0.4
Utility	1		0.414		0.822		0.8455

	ALTERNATIVES		
	BCI	**Eye Tracking**	**Voice Commands**
Number of Commands	Documented success with 4 commands Adding more than 4 reduces accuracy	Would require screen showing map of room, user look to desired location or to drive robot, fixation to select. Midas touch phenomena leads to a lot of false selections	Unlimited
Time Sensitivity	4 s delay	40ms slower than mouse	2.2 s
TRL	2	9	9
User Population	100%	92.22%	81%
New User Time	Significant	Calibration is a key factor defining the accuracy of any eye tracker.	Set up profile, 20 min
Accuracy	%70 accuracy to select a block They were able to classify EEG signals into right and left hand movements using a neural network classifier with an accuracy of 80% A single trial right/left hand movement classification is reported in [18]. The authors analyzed both executed and imagined hand movement EEG signals and created a feature vector consisting of the ERD/ERS patterns of the mu and beta rhythms and the coefficients of the autoregressive model.	-eye tracking is also subject to false positives that can interfere with everyday use. -fixation on an object (without intent to manipulate it) inadvertently triggers its selection and manipulation -Tobii Pro Accuracy and Precision report	The results demonstrated all of them can controlthe wheelchair by speech efficiently with an average success rate and response time are 92% and 2.2s respectively.

	Artificial Neural Networks (ANNs) is applied to two kinds of testing datasets and an average recognition rate of 93% is achieved 79.48% and 85.00% for 4 signal wheelchair manipulation		
Maintainability	Minor re-calibration, re-training	No re-training, occasional calibration required	No calibration
Reliability	%15 no signal aquired; 67% success rate	- Glassses and light eye color may lead to detection problems	Few documented changes.