Control Interface for a High Velocity Teleoperated Robot

by

John D. Thele

Bachelor of Science in Electrical Engineering and Computer Science, Massachusetts Institute of Technology, 1998

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 1999

© John D. Thele, MCMXCIX. All rights reserved.

Author: ............................................................

Department of Electrical Engineering and Computer Science

February 3, 1999

Certified by: ............................................................

David S. Kang

Program Development Manager, Charles Stark Draper Laboratory
Thesis Supervisor

Certified by: ............................................................

John J. Leonard

Assistant Professor of Ocean Engineering, Massachusetts Institute of Technology
Thesis Supervisor

Accepted by: ............................................................

Arthur C. Smith

Chairman, Departmental Committee on Graduate Theses
Department of Electrical Engineering and Computer Science
Control Interface for a High Velocity Teleoperated Robot
by
John D. Thele

Submitted to the Department of Electrical Engineering and Computer Science
on February 3, 1999, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science

Abstract

This thesis describes the design and implementation of a control interface suitable for controlling a microrover at high speed. For the purposes of this thesis, a control interface describes the collection of software and input devices used by an operator to interact with a microrover that is not within sight. A microrover is a small, lightweight, robotic vehicle, often less than a meter in length, used to provide a mobility platform for mounting a variety of sensors. Traditional remotely operated microrover systems (where the operator cannot see the microrover) are unable to operate at speeds greater than 0.5 m/s. The microrover used to prepare this thesis operates at a maximum speed of 1.5 m/s, but the control interface described herein should be applicable at even greater speeds, perhaps as fast as 5.0 m/s. Preliminary results show that this control interface performs adequately in comparison to a more traditional interface that uses only live video from the robot.

Thesis Supervisor: David S. Kang
Title: Program Development Manager, Charles Stark Draper Laboratory

Thesis Supervisor: John J. Leonard
Title: Assistant Professor of Ocean Engineering, Massachusetts Institute of Technology
Acknowledgments

First of all, I would like to thank my Draper supervisor, Dr. David Kang, and my MIT advisor, Professor John Leonard, for giving me the opportunity to develop the HVTR system. Without their assistance, this thesis would not have been possible. I must also thank my partner on the HVTR project, Atif Chaudhry, for his help in putting the whole system together. I wish him luck on his upcoming thesis. I would also like to recognize all of my coworkers in the Unmanned Vehicle Lab for their enthusiasm and assistance. The lab has been a fun and challenging place to work for the last two and a half years. Finally, many thanks to all my friends who have made MIT such an enjoyable experience, and to my parents who have encouraged me in all of my endeavors.
Assignment

Draper Laboratory Report Number T-1334

In consideration for the research opportunity and permission to prepare my thesis by and at The Charles Stark Draper Laboratory, Inc., I hereby assign my copyright of the thesis to The Charles Stark Draper Laboratory, Inc., Cambridge, Massachusetts.

Copyright assigned by: .................................................................

John D. Thele

This thesis was prepared at The Charles Stark Draper Laboratory, Inc. under Contract 18621. Publication of this thesis does not constitute approval by The Charles Stark Draper Laboratory, Inc. or the sponsoring agency of the findings or conclusions contained herein. It is published for the exchange and stimulation of ideas. Permission is hereby granted by the author to the Massachusetts Institute of Technology to reproduce any or all of this thesis.

Permission granted by: .................................................................

John D. Thele
Contents

1 Introduction ............................................ 13
   1.1 Background ......................................... 13
   1.2 HVTR Goal ........................................... 14
   1.3 Thesis Outline ...................................... 14

2 Developing the HVTR Interface ......................... 17
   2.1 Microrover Control Methods ....................... 17
      2.1.1 Autonomy .................................... 17
      2.1.2 Teleoperation ................................ 20
   2.2 Difficulties Caused by High Velocity .............. 21
      2.2.1 Sensor Shortfalls ............................ 21
      2.2.2 Processing Shortfalls ....................... 22
   2.3 HVTR Approach .................................... 22
   2.4 Supervisory Control Model ....................... 23
   2.5 Development of the Human Computer Interface .... 27

3 HVTR Implementation .................................. 29
   3.1 Microrover Hardware ............................... 29
      3.1.1 Platform ..................................... 29
      3.1.2 Battery ...................................... 31
      3.1.3 Laptop ...................................... 32
      3.1.4 Pioneer AT Microcontroller .................. 33
      3.1.5 Modem ....................................... 33
      3.1.6 Motors and Encoders .......................... 33
      3.1.7 Bumpers ..................................... 34
      3.1.8 Sonars ...................................... 34
      3.1.9 Video Camera and Transmitter ............... 35
      3.1.10 Gyro ....................................... 36
3.1.11 Future Positioning Sensors ........................................ 36
3.2 Microrover Software .................................................. 37
  3.2.1 Gyro Microcontroller ........................................... 37
  3.2.2 Pioneer AT Microcontroller ................................... 39
  3.2.3 Laptop ................................................................... 44
3.3 Control Station Software ............................................... 45
  3.3.1 Graphical User Interface ......................................... 46
  3.3.2 Kalman Filter ....................................................... 54
  3.3.3 Position Recalibration ............................................ 55
  3.3.4 Evolution of the Computer Interface .......................... 55

4 HVTR Testing ................................................................. 59
  4.1 Testing Environment .................................................. 59
  4.2 Testing Methodology .................................................. 60
  4.3 Testing Results ........................................................ 63
  4.4 Testing Analysis ........................................................ 75
    4.4.1 Time .................................................................. 75
    4.4.2 Driving Technique ................................................ 75
    4.4.3 Summary ............................................................ 76

5 Conclusion ................................................................. 79
  5.1 Thesis Summary ........................................................ 79
  5.2 Future Possibilities .................................................... 80
  5.3 Conclusion ............................................................... 80

A Unmanned Vehicle Laboratory ....................................... 85
  A.1 The MITy series ....................................................... 85
    A.1.1 MITy-1 ............................................................... 85
    A.1.2 MITy-2 ............................................................... 85
    A.1.3 MITy-3 ............................................................... 85
  A.2 The EOD series ......................................................... 86
  A.3 Companion ............................................................... 86
  A.4 DSAAV .................................................................. 86
  A.5 WASP .................................................................... 86
  A.6 VCUUV ................................................................ 87
B EOD-2

B.1 EOD-2 Robot ................................................. 89
B.2 EOD-2 Control Station ..................................... 89
  B.2.1 Interface .................................................. 89
  B.2.2 Path Planning ............................................ 90
  B.2.3 Navigation ................................................ 90
  B.2.4 Map Building ............................................. 91
  B.2.5 Multiple Robot Simulation ........................... 91
## List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-1</td>
<td>Genghis Behavior Architecture</td>
<td>19</td>
</tr>
<tr>
<td>2-2</td>
<td>Typical User-Centered Design Cycle</td>
<td>28</td>
</tr>
<tr>
<td>3-1</td>
<td>HVTR Microrover</td>
<td>30</td>
</tr>
<tr>
<td>3-2</td>
<td>Pioneer AT Components</td>
<td>30</td>
</tr>
<tr>
<td>3-3</td>
<td>Pioneer AT Dimensions</td>
<td>31</td>
</tr>
<tr>
<td>3-4</td>
<td>Sonar Footprint</td>
<td>35</td>
</tr>
<tr>
<td>3-5</td>
<td>Gyro Microcontroller I/O</td>
<td>38</td>
</tr>
<tr>
<td>3-6</td>
<td>Trapezoidal Velocity Profile</td>
<td>42</td>
</tr>
<tr>
<td>3-7</td>
<td>HVTR Control Station</td>
<td>45</td>
</tr>
<tr>
<td>3-8</td>
<td>Window Layout of HVTR Control Station</td>
<td>46</td>
</tr>
<tr>
<td>3-9</td>
<td>Navigation Window</td>
<td>47</td>
</tr>
<tr>
<td>3-10</td>
<td>Zoom Window</td>
<td>49</td>
</tr>
<tr>
<td>3-11</td>
<td>Video Window</td>
<td>50</td>
</tr>
<tr>
<td>3-12</td>
<td>Sensor Window</td>
<td>50</td>
</tr>
<tr>
<td>3-13</td>
<td>Battery Voltage Window</td>
<td>52</td>
</tr>
<tr>
<td>3-14</td>
<td>Velocity Window</td>
<td>53</td>
</tr>
<tr>
<td>3-15</td>
<td>Button Window</td>
<td>54</td>
</tr>
<tr>
<td>4-1</td>
<td>Floorplan of Test Course</td>
<td>61</td>
</tr>
<tr>
<td>4-2</td>
<td>Picture of Test Course</td>
<td>62</td>
</tr>
<tr>
<td>4-3</td>
<td>Velocity Inputs of Subject 1</td>
<td>76</td>
</tr>
<tr>
<td>4-4</td>
<td>Velocity Inputs of Subject 3</td>
<td>77</td>
</tr>
</tbody>
</table>
List of Tables

3.1 Pioneer AT Mobility Specifications .................................. 32
3.2 Hitachi Traveler Laptop Computer Specifications ............. 32
3.3 Gyro Data Bit Selection .................................................. 39
3.4 Commands Used by HVTR .............................................. 40
3.5 Command Packet Structure ............................................. 40
3.6 Pioneer AT Return Packet Structure .............................. 41

4.1 Subject 1 Test Results .................................................. 65
4.2 Subject 2 Test Results .................................................. 66
4.3 Subject 3 Test Results .................................................. 67
4.4 Subject 4 Test Results .................................................. 68
4.5 Subject 5 Test Results .................................................. 69
4.6 Subject 6 Test Results .................................................. 70
4.7 Test Results for All Subjects Grouped by Subject, Part A .... 71
4.8 Test Results for All Subjects Grouped by Subject, Part B .... 72
4.9 Test Results for All Subjects Grouped by Test, Part A ........ 73
4.10 Test Results for All Subjects Grouped by Test, Part B ....... 74
Chapter 1

Introduction

This thesis is a case study in the development of a new control interface for mobile robots. Unlike many conventional interfaces which operate a robot at low speed, this interface is designed to control the robot at high speed. The contents of this thesis explore the design, implementation, and testing of this new interface.

1.1 Background

Microrovers are small, lightweight, robotic vehicles, often less than a meter in length, that are useful mobility platforms for mounting a variety of sensors. The Unmanned Vehicle Laboratory (UVL) at The Charles Stark Draper Laboratory has had many years of experience in developing microrovers [5][6][8][14]. See Appendix A for more information about the many robotic platforms developed by the UVL.

Recently, the UVL built a pair of microrovers tailored for an explosive ordinance disposal (EOD) mission named EOD-1 and EOD-2. Their mission was to aid in the location and disposal of small unexploded surface ordinance scattered by cluster bombs [14]. A brief description of the EOD microrovers is located in Appendix A.2.

EOD-1 was built to test the custom six-wheeled flexible frame mobility platform to be used for the EOD microrovers. The EOD-1 microrover had no control station, but instead used a joystick connected to an interface that allowed it to communicate velocity and steering inputs to the robot. It demonstrated the speed and off-road capabilities of the EOD microrovers.

EOD-2 was an autonomous microrover, built to perform such tasks as searching for unexploded ordinance within an obstacle-ridden field. Its control station allowed an operator both to command the robot to perform tasks, as well as display a two-dimensional model of the robot’s environment. The operator could then modify the
environment model with a priori knowledge of areas that were free of unexploded ordinance. It was intended that EOD-2 would maneuver at high speeds in areas that were safe, and traverse unexplored territory at low speed while searching for unexploded ordinance. A detailed description of the EOD-2 microrover and its control station can be found in Appendix B.

Though EOD-1 proved that the platform was capable of being controlled at high speed, the software, sensors, and computers within the EOD-2 system were unable to control the microrover at speeds greater than 0.5 m/s.

1.2 HVTR Goal

The failure of EOD-2 to perform at high speeds prompted the UVL to study the design of a microrover that could be controlled by an operator at high speeds, even when the operator does not have direct visual contact with the robot.

This new project is named HVTR, an acronym for High Velocity Teleoperated Rover. The long-term goal of this project is to control a microrover at speeds as high as 5.0 m/s. The interim goal is to control the Pioneer AT robotic platform, which the HVTR is currently using as its mobility platform, at its top speed of 1.5 m/s.

High speed is important because there are many potential uses for a microrover where it is necessary to traverse long, relatively obstacle-free distances. Navigating in an urban environment is an excellent example of where a high speed microrover would be useful for moving from block to block in a timely manner. The total duration of the robot’s mission is largely dependent on how much time the robot spends traveling through these relatively unobstructed areas.

1.3 Thesis Outline

The remaining chapters of this thesis will explore the development of the HVTR system, provide a detailed description of its implementation, analyze its performance, and provide a look at future research that might build on the lessons learned from the HVTR project. A summary, chapter by chapter, follows.

Chapter 2 - Developing the HVTR Interface

This chapter begins with an analysis of several control architectures that are in use today. It then explains why controlling a microrover at high speed is challenging for some of these architectures, and impossible for others. Next, it describes the approach that the HVTR project is using to overcome the difficulties imposed by
high speed. Finally, the influence of the theory of supervisory control and human computer interface development paradigms are presented.

Chapter 3 - HVTR Implementation

All of the implementation details of the HVTR system are described in this chapter. The hardware and software of both the microrover and its control station are explained. Additionally, a design history of the graphical user interface is given.

Chapter 4 - HVTR Testing

This chapter describes the results of the tests conducted using the current version of the HVTR system.

Chapter 5 - Conclusion

An analysis of what can be learned from the HVTR project as well as some future research opportunities are described in this chapter.
Chapter 2

Developing the HVTR Interface

This chapter explores how the HVTR interface was developed through five major sections:

- a review of several control architectures that are in use today,
- an analysis of several problems specific to high speed microrovers,
- a description of the initial insight into the HVTR interface,
- the influence of the theory of supervisory control on the HVTR interface, and
- an overview of a user-centered design technique for human computer interface development.

2.1 Microrover Control Methods

To better understand the problem that the HVTR project is trying to solve, it is necessary to analyze the control techniques used by other mobile robotic platforms. Different robotic systems can be compared by examining the method by which the vehicle’s speed and heading are controlled. Some robots use intelligent software to control these low-level functions. These autonomous robots only require the operator to provide abstract tasks for the robot to perform. Alternatively, some systems require the operator to directly control the robot’s speed and heading. These robots are said to be teleoperated.

2.1.1 Autonomy

Autonomous robots tend to fall into two categories, those that determine their next action based on a symbolic model of their environment, and those that base
their next action solely on the input they are receiving from their sensors at that moment in time.

Symbolic Architecture

A symbolic architecture is the most traditional approach for developing an autonomous robot. It involves building a model of the environment surrounding the robot. This model can include prior knowledge about the environment that the robot will be operating in as well as information gathered by the robot’s sensors once it has begun its mission. To move, the robot continuously determines a speed and heading that will allow it to maneuver without collisions based on its internal model of its surroundings.

This method has been shown to work in very simple environments. For example, the Stanford Research Institute developed a mobile robot named Shakey in the late 1960’s that navigated within a set of specially prepared rooms. Its task was to push large colored blocks and wedges between rooms at the direction of an operator who input commands via a teletype. Except for the blocks and wedges, the rooms were completely bare. The walls were of a uniform color and well lit. The floor was of a lighter color than the walls, and there were dark baseboards to provide an easily identifiable boundary between the floor and the walls. Shakey’s primary sensor was a black and white television camera, which it used to identify walls and objects when creating its model of the world [2].

The Companion robot, described in Appendix A.3, is an example of a more modern robot based on a symbolic architecture that is capable of navigating hallways and rooms without the special preparation that Shakey required. To produce its symbolic map, Companion relies solely on data gathered from its twenty-four Polaroid ultrasonic transducers and its laser range finder. The laser range finder is mounted in a movable housing that provides full lateral coverage about the robot at pitch angles ranging from straight down to beyond horizontal. For navigation, it uses a gyroscope and two wheel encoders to determine where it has been and is accurate to within 1.2% of path length [3].

Behavior-Based Architecture

Unlike symbolic architectures, behavior-based architectures require very little, if any, stored state about the robot’s environment. In this architecture, robots are programmed with many predefined behaviors that are activated or deactivated based on stimuli from the robot’s sensors. Multiple behaviors can be active at one time and
can interact with each other to produce a more complex resultant behavior.

An interesting behavior-based architecture popularized by Rodney Brooks in the mid-1980’s is the subsumption architecture. The idea behind a subsumption architecture is to layer multiple behaviors in such a way that behaviors at a higher level influence more basic behaviors at a lower level.

![Diagram of Genghis Behavior Architecture](image)

Figure 2-1: Genghis Behavior Architecture

The best way to explain Brooks’ subsumption architecture is to explain the Genghis robot built in the MIT Artificial Intelligence Laboratory in 1988. Genghis, an early example of subsumption architecture, is a six legged robot built with eight behavior layers\(^1\) that can walk over obstacles while steering toward a moving object [2]. All components of Genghis’ software are illustrated in Figure 2-1. The eight behaviors that define Genghis are:

1. **Stand** - The lowest-level behavior controlling Genghis simply commands the legs to hold the robot in a standing position.

2. **Simple Walk** - This behavior signals a simple pattern of leg movements that causes Genghis to walk. The robot is insensitive to its terrain, so it performs poorly when it encounters obstacles.

3. **Force Balancing** - This behavior begins to compensate for rough terrain by reducing how far the leg moves downward when it is positioned on top of an obstacle. This lessens the rolling and pitching of Genghis.

---

\(^1\) Brooks actually describes Genghis as having twelve layers, with five different behaviors implementing the mechanics of the simple walk behavior that is outlined as one behavior layer in this list.
4. **Leg Lifting** - When Genghis bumps into an obstacle with its legs, this behavior causes the legs to be raised higher to allow the robot to more easily travel over obstacles.

5. **Whiskers** - Genghis' whiskers are used to better anticipate obstacles with this behavior. When a whisker is bumped, Genghis raises its front legs higher to overcome whatever may be in its path.

6. **Pitch Stabilization** - This behavior decreases the pitching of the robot as it maneuvers over obstacles to further improve Genghis' walking ability.

7. **Prowling** - Unless Genghis' forward-looking IR sensors sense movement nearby, the prowling behavior inhibits the simple walk behavior.

8. **Steered Prowling** - This behavior steers the robot in the direction of the motion that it is seeing with its IR sensors.

In general, behavior-based robots tend to work well in real-world environments. Unfortunately, designing behaviors more complex than the simple prowling done by Genghis becomes a very complex problem. More complex behaviors will require a large number of additional layers. Each of these layers must influence one or more of its subordinate layers. Very shortly, the layer design which appeared very simple in Genghis' case becomes dauntingly complex as all of the layers and their interactions compete to control the robot's behavior.

### 2.1.2 Teleoperation

The simplest teleoperated robots use no sensors at all. An operator stays within close visual proximity of the robot and uses a control device of some form to direct the robots movements. Other teleoperated robots rely on sensors to provide information to the operator about the environment that the robot is being operated in. This allows the operator to relax the constraint of keeping close visual proximity to the robot at all times. Most teleoperated microrover systems only use sensors onboard the robot, though some use additional sensors external to the robot.

**Onboard Sensors**

The most common and simplest teleoperated robot is one which has a simple video camera mounted on the robot. An operator watches the live video on a television, and controls the robot via an input device such as a joystick. The primary difficulty in using such a system is the lack of peripheral vision. It is difficult for the operator
to visualize the complete environment that the robot is located in using only the ground-level view from the camera.

Therefore, external sensors can be used alone or in combination to provide the operator with more information about the environment than is gathered by a video camera alone.

**External Sensors**

External sensors complement the data being gathered by a robot’s onboard sensors, thereby providing a more accurate depiction of its surrounding environment. An obvious problem with this solution is that the robot can not travel out of range of the external sensors. If it does, the benefit of the external sensors will be lost.

A prime example of a teleoperated robot that used external sensors as well as onboard sensors is the NASA/JPL Sojourner mission to Mars. The Sojourner was programmed to have several autonomous capabilities, but these were not used during the early part of the mission. Instead, all movements were carefully choreographed by operators on Earth.

A five foot mast holding a stereo camera system was mounted on top of the Pathfinder. The cameras would photograph the Sojourner and its surrounding terrain and send these images back to Earth. Each night\(^2\), the stereo images would be processed into a three-dimensional map of the terrain surrounding Sojourner. This map enabled the operators to plan a sequence of movements that would safely navigate Sojourner around nearby objects. In the morning, these commands would be broadcast to the robot to be executed [11].

### 2.2 Difficulties Caused by High Velocity

A microrover traveling at high speed introduces several problems which make it difficult to control. These problems can be inherent in the sensors or in the processing ability of the robot’s computers.

#### 2.2.1 Sensor Shortfalls

All sensors have an inherent sampling rate that cannot be exceeded. For example, sonars are limited by the speed of sound, and by the limitations of their own sounding mechanism. The sonar controlling circuitry must wait until the sonar transducer has

\(^2\)For the Sojourner microrover, night and day refer to the Martian day and night. Sojourner moved only during the day when it had solar power to prevent unnecessary drain on its non-rechargeable batteries.
stopped vibrating before it can begin to identify a return signal. Laser range finders can be fast because the laser beam travels at the speed of light, but they only focus on one point in space at a time. This means that many samples need to be taken to determine if there are obstacles near the microrover. The collection of the multiple samples is time consuming. Video cameras can sample very quickly, but robots that process video for obstacle detection subsample the video signal at a much slower rate because they are unable to process the large quantity of data in real time.

Recall that microrovers are small vehicles, normally less than a meter in length. The requirement of high speed capability necessitates a drive system using either wheels or tracks because it would be amazingly complex to engineer a stable legged robot of small scale that could move quickly. Because of the robot's small size, the inclination of the short wheelbase platform changes rapidly as it bounces over any irregularities in the terrain. This hinders the effectiveness of most sensors, especially the forward looking ones. The sensors spend a considerable amount of time focused on the ground or focused at the sky where they collect spurious and misleading data [11].

2.2.2 Processing Shortfalls

As speed increases, more demands are placed on the microrover's processors. If they are to be used to detect obstacles, they must first filter out the useless sensor data gathered by misdirected sensors before they can begin to assess their environment. Invariably, the microrover gets too close to an obstacle before it can sense the obstacle's location, determine a course of action (or have an operator determine a course of action), and then signal the drive system what should be done to avoid the obstacle [11].

Greater care must be taken in designing a drive system that is to be used at high speeds and on rough terrain. It must be robust enough to not become unstable under these conditions. For example, the EOD-2 drive system was well-suited for very precise feedback and control of the microrover at low speeds, but it would become so unstable at high speeds that the large fluctuations in current demanded by the motors would reboot the onboard processor.

2.3 HVTR Approach

Because of the sensor and processor shortfalls common to high-speed travel, the use of autonomy in the HVTR project was deemed impractical. Further, high speed travel cannot be teleoperated reliably using only onboard sensors because the minimal and
noisy data presented to the operator does not provide an adequate understanding of the microrover's environment. Therefore, a teleoperated system making use of external data was deemed necessary.

The insight for the HVTR approach came from popular video games which provide a downward facing omniscient view of a simulated terrain. To mimic this, a reconnaissance photograph taken by an airborne vehicle or satellite is used as the primary window in the control station. Using a joystick, the operator "drives" a graphic representation of the microrover around on this terrain image. These control inputs are also sent to the microrover. If the on-screen robot icon is properly scaled and positioned on the photo, then the icon's course over the picture of the terrain should echo the microrover's track over the true terrain.

This approach relies on the assumption that most objects in the reconnaissance image will remain static after the photograph is taken. This will be true for buildings, roads, trees, and other immovable objects, but cars, people, and similar mobile objects will change position. For this reason, a forward-facing video camera is mounted on the microrover to allow the operator to identify gross differences between the terrain photograph and the current world. Though the video may not be stable, identifying large dissimilarities is an easy task for the robot operator.

The operator's knowledge of world phenomenon becomes very useful when interpreting the terrain photograph. For example, if an operator sees a road in the reconnaissance photo, it is reasonable to assume that cars might be traveling on that road that are not shown in the terrain image. Similarly, sidewalks might contain people. If the robot is to be used in a populated urban environment, it might be best to take the reconnaissance photo during the day for best lighting, but perform the actual mission at night when the streets and sidewalks are relatively free of obstructions.

2.4 Supervisory Control Model

The theory of supervisory control as presented by Thomas Sheridan in *Telerobotics, Automation, and Human Supervisory Control* outlines five generic supervisory functions for remotely controlling a robot.

Planning

This is the hardest function to model. Formally it means

1. gaining experience and understanding of the physical process to be controlled, including the constraints set by nature and circumstances
surrounding the job,

2. setting goals that are attainable, or objectives along with tradeoffs, that the computer can “understand” sufficiently well to give proper advice or make control decisions, and

3. formulating a strategy for going from the initial state to the goal state.

Teaching the computer

The supervisor must translate goals and strategy into detailed instructions to the computer such that it can perform at least some part of the task automatically, at least until the instructions are updated or changed or the human takes over by manual control. This includes knowing the requisite command language sufficiently well that goals and instructions can be communicated to the computer in correct and timely fashion.

Monitoring automatic control

Once the goals and instructions are properly communicated to the computer for automatic execution of that part of the task, the supervisor must observe this performance to ensure that it is done properly, using direct viewing or whatever remote sensing instruments are available. The prompt detection of the presence and location of failures, or of conflicts between actions and goals, and the anticipation that either of these is about to occur, are essential parts of the supervisor’s job.

Intervening to update instructions or assume direct control

If the computer signals that it has accomplished its assigned part-task, or if it has apparently run into trouble along the way, the human supervisor must step in to update instructions to the computer or to take over control in direct manual fashion, or some combination of the two. Since the controlled process is an ongoing dynamic system, not a machine that can be arbitrarily stopped and started again like a computer, the takeover itself must be smooth so as not to cause instability. Similarly, reverting to the automation must be smooth.
Learning from experience

The supervisor must ensure that appropriate data are recorded and computer-based models are updated so as to characterize current conditions with the most accurate information. Historical data must continuously be analyzed for trends or contingencies leading to abnormalities. All such information must be in a form usable in the future in the four preceding steps [10].

Applying the supervisory control model to the HVTR project introduces three more components needed in the control interface. The first is an easily activated routine which halts the movement of the microrover. The second is a method of recorrelating the heading and position of the robot icon with that of the microrover. The third is a method for logging traversed path data both on screen and on disk for later analysis.

Here is a step-by-step description of the HVTR control model within the framework of Sheridan’s supervisory control model:

Planning

1. The supervisor begins by analyzing the reconnaissance image within the control station for regions that the microrover can be safely maneuvered in. The supervisor’s prior knowledge about terrain features (pavement, grass, dirt, etc.) and obstacles (buildings, curbs, cars, etc.) contained within the image, or in the case of cars, implied to be within the imaged region, is used to determine the safe regions. The supervisor’s understanding of the physical constraints and maneuverability of the microrover also influence the decision of what regions the microrover can be maneuvered in.

2. Next, the supervisor chooses a goal location or set of locations that the microrover is to be maneuvered to. These objectives should be within the regions that the supervisor has already determined to be safe for the microrover to maneuver in.

3. Finally, the supervisor formulates a strategy for directing the microrover from its beginning location to each of its goal locations that involves traversing only regions that the supervisor has determined to be safe for the microrover to travel in.
Teaching

By navigating the on screen robot icon through the safe regions to each of the goal locations with joystick inputs, the supervisor is instructing the computer how it should maneuver the microrover in its true environment.

Monitoring

While directing the on screen robot icon, the supervisor must monitor the live video from the microrover’s camera for gross dissimilarities between what the supervisor thinks the microrover should be seeing based on the terrain image, and what the microrover video is actually showing.

Intervening

If there is a discrepancy between what the supervisor thinks the microrover video should be showing and what it actually shows, the supervisor will press and hold any of the buttons on the joystick. This signals the microrover to deaccelerate quickly to a stop. One of three options can then be taken depending on the nature of what the supervisor sees in the video image.

1. If there is an immobile object in front of the microrover that is not present on the reconnaissance image, the supervisor must return to the planning stage and formulate a new strategy for maneuvering the microrover to its goal locations.

2. If there is a mobile object, such as a person, in front of the microrover, the supervisor can choose to either wait for the mobile object to move out of the way, or return to the planning stage and reformulate a strategy for reaching the goal locations as done when encountering immobile objects.

3. If the supervisor determines that the microrover is not in the position represented by the robot icon on the terrain imagery, than the supervisor must perform an operation that recorrelates the robot icon with the true location of the microrover. A further explanation of how this is done is described in Section 3.3.3.

Learning

To aid the supervisor in reviewing what instructions were given to the computer, an onscreen track of the robot icon’s location is presented. This data is also logged to disk for future analysis.
2.5 Development of the Human Computer Interface

With both the general approach and supervisory model outlined for the control interface, the next reasonable step is to begin the code development for the HVTR operator station. Developing a piece of software as visually complex as the operator station requires multiple revisions to slowly build the software features into a cohesive user interface. Jenny Preece, in *A Guide to Usability: Human Factors in Computing*, outlines some key ideas to keep in mind when designing for the user:

- Focus on the users and the users’ needs, and in so doing make user issues rather than technical considerations central in the design process.

- Carry out a task analysis in which details of the users’ tasks and information about the task environment are collected, so that users’ needs are well understood. Task analysis needs to be done in addition to a general requirements analysis, which tends to focus on what functionality is required and not on how to provide that functionality.

- Carry out early testing and evaluation with users to ensure that the system is designed to meet their needs.

- Design iteratively with many cycles of ‘design – test with users – re-design’. Do not expect to produce one ‘right’ solution which is not changed, but instead aim to design an evolving system which is tailored to users’ needs more with each iteration [7].

A flowchart illustrating these design philosophies is shown in Figure 2-2. The current implementation of the graphical user interface for the operator station is described in Section 3.3.1, but of course this current implementation is the result of numerous revisions. An analysis of the major revisions and the reasons they were made can be found in Section 3.3.4.
Figure 2-2: Typical User-Centered Design Cycle
Chapter 3

HVTR Implementation

This chapter describes in detail all components of the HVTR implementation. Major sections include

- a description of the microrover hardware,
- an explanation of the function of the microrover software,
- a look at the features and functionality of the control station, and
- an analysis of how the computer interface has evolved.

3.1 Microrover Hardware

The HVTR Microrover is a fusion of primarily off-the-shelf commercial products in order to speed up the assembly of the robot as well as to ensure that any damaged component could be easily replaced. Only the circuitry that processes the gyro outputs is custom-made. Figure 3-1 is a picture of the HVTR Microrover in its current configuration.

3.1.1 Platform

The preliminary implementation of the HVTR system was done using the EOD-1 robot (Appendix A.2), but its repeated mechanical and electrical failures prompted the decision to use the well-tested and readily available Pioneer AT.

The Pioneer AT is a four-wheeled robotic platform manufactured by Real World Interface, Inc. (RWI) and distributed by ActivMedia Inc. Figure 3-2 illustrates the components of the Pioneer AT. Figure 3-3 shows its dimensions. The Pioneer Operation Manual describes the Pioneer family of robots in the following way.
Figure 3-1: HVTR Microrover

Figure 3-2: Pioneer AT Components
They are truly off-the-shelf, *plug-and-play*, intelligent mobile robots, containing all of the basic components for sensing and navigation in a real-world environment, including battery power, drive motors and wheels, position/speed encoders, and ultrasonic range finders—all managed via an onboard MC68HC11-based microcontroller [1].

![Image of Pioneer AT Dimensions]

**Figure 3-3: Pioneer AT Dimensions**

As the Pioneer Operation Manual states, the Pioneer AT is capable of intelligent "sensing and navigation in a real-world environment", but like many other intelligent microrovers, the autonomous capabilities are only effective at low speed (see Section 2.2: Difficulties Caused by High Velocity). More important to the HVTR system is the mechanical capabilities of the Pioneer AT. At a rated top speed of 1.5 m/s, the Pioneer AT is the fastest off-the-shelf microrover available for purchase. An additional advantage is the wealth of usable space both inside the robot and on the payload deck for all of the additional components necessary for the HVTR system. Table 3.1 summarizes the mobility specifications of the Pioneer AT.

### 3.1.2 Battery

Inside the Pioneer AT is a single 12 VDC, 7 A/hr sealed lead-acid battery which provides power for everything within the Pioneer AT except the external laptop. The power for the modem, the video camera, and the video transmitter is converted to 9 VDC by a linear regulator. The Pioneer AT microcontroller as well as the gyro
<table>
<thead>
<tr>
<th>Maximum Translational Speed</th>
<th>1.5 m/s</th>
</tr>
</thead>
<tbody>
<tr>
<td>Traversable Slope</td>
<td>80% grade</td>
</tr>
<tr>
<td>Steering</td>
<td>Differential</td>
</tr>
<tr>
<td>Swing Radius (see Figure 3-3)</td>
<td>350 mm</td>
</tr>
<tr>
<td>Turn Radius</td>
<td>0 mm</td>
</tr>
<tr>
<td>Maximum Rotational Speed</td>
<td>360 °/s</td>
</tr>
<tr>
<td>Pushing Force</td>
<td>8 kg</td>
</tr>
</tbody>
</table>

Table 3.1: Pioneer AT Mobility Specifications

use the 12 VDC output from the battery. During intensive use of the robot at high speed, the battery has enough power for about one hour of operation.

3.1.3 Laptop

Attached to the payload deck of the Pioneer AT by vibration reducing mounts is a platform that holds a Hitachi Traveler laptop computer. The laptop has a Pentium 133 processor with 40 MB of RAM. More specifications are given in Table 3.2. This computer was chosen because of its small size and its 3 PCMCIA slots. Using multiple RS232 serial port PCMCIA cards, it is possible for this laptop to communicate with a large number of sensors using the popular RS232 serial interface standard.

<table>
<thead>
<tr>
<th>Processor</th>
<th>Pentium 133 MMX</th>
</tr>
</thead>
<tbody>
<tr>
<td>RAM</td>
<td>40 MB</td>
</tr>
<tr>
<td>Hard Drive</td>
<td>1083 MB</td>
</tr>
<tr>
<td>Length</td>
<td>173 mm</td>
</tr>
<tr>
<td>Width</td>
<td>234 mm</td>
</tr>
<tr>
<td>Height</td>
<td>33 mm</td>
</tr>
<tr>
<td>Weight with Batteries</td>
<td>1.25 kg</td>
</tr>
</tbody>
</table>

Table 3.2: Hitachi Traveler Laptop Computer Specifications

Using a laptop as the primary onboard computer is not the standard approach used by most of the robots developed in the UVL. Obviously, elements like the display and keyboard serve no purpose when the microrover is in motion, but the extra space they occupy and their extra weight were a reasonable tradeoff when compared to the time it would have taken to construct and configure a more traditional PC104 computer.
system. Additionally, the display and keyboard are beneficial when one wants to make a small change to the onboard code. With the laptop mounted in the highly accessible position on top of the robot, it is possible to modify code in the field without having to connect the robot to any external devices.

### 3.1.4 Pioneer AT Microcontroller

The electronic system of the Pioneer AT, excluding the sensors, the battery, and the laptop, are all contained on two circuit boards. The heart of this electronics is a 16MHz Motorola MC68HC11F1 running on a 4MHz bus. This microcontroller controls the following inputs and outputs (I/O) [1].

- two high-power, reversible motor drivers
- two position-encoder inputs
- eight multiplexed sonar outputs
- eight digital input ports
- eight digital output ports
- one 8-bit analog-to-digital input port (0-5 VDC at 10 Hz)
- one digital timer output (1 µs resolution)
- one digital timer input (1 µs resolution; 65 µs duration)
- one RS232 serial port

### 3.1.5 Modem

Communication between the microrover and the control station is done using a pair of Proxim 900 MHz, 9600 baud radio modems. The microrover’s modem is mounted inside the top console of the Pioneer AT, and it connects to one of the laptop’s serial ports.

### 3.1.6 Motors and Encoders

Each of the four wheels of the Pioneer AT are driven by a reversible-DC motor. The two motors on each side of the microrover are connected by a toothed belt. This forces the two wheels on each side of the robot to always spin at the same velocity,

---

1The PC104 standard describes a class of stackable computer modules that share a small footprint and a common bus. Popular modules for robotic systems include Pentium processors and digital signal processing (DSP) boards.
thereby minimizing slippage. The front motors each include a high-resolution (100
ticks per revolution) optical shaft encoder to provide accurate data on the position of
the wheels. An optical shaft encoder is an optical plate with many transparent lines
inscribed on it that is attached to the wheel axle.

As the plate spins, it interrupts an optical beam. A pulse counter counts how
many inscribed lines have passed the beam [2]. These encoder counts are processed
by the Pioneer AT’s MC68HC11 microcontroller.

3.1.7 Bumpers

Because the HVTR system often commands the Pioneer AT at high speed, it
was correctly reasoned that the microrover would experience many collisions during
testing. To protect the body of the robot as well as its wheels, the Pioneer AT for
the HVTR project was purchased with the optional bumpers. The bumpers extend
the full width of the vehicle, including the wheels, on both the front and back of
the robot. They consist of aluminum framing holding a 3 inch hollow rubber insert.
Inside the hollow rubber insert is a sensor that signals the MC68HC11 microcontroller
whenever the rubber insert is compressed.

3.1.8 Sonars

Built into the console on top of the Pioneer AT are seven ultrasonic sonar trans-
ducers which provide range information to nearby objects. Five of the sonars create
a forward facing arc, each separated by 15 degrees from the next. The other two
sonars are side-facing, one on the left side of the console, and the other on the right.
The sonars fire sequentially at a rate of 25 Hz. Their sensitivity ranges from about
10 cm to nearly 3.6 m. The ranging information is processed by the MC68HC11
microcontroller.

It is a widely-held misconception that sonars return accurate depth measurements
under most conditions. They are actually quite prone to error. Brooks outlines several
problems [2].

1. The speed of sound varies with air temperature. It is not uncommon
for a sonar to change its distance measurement by 10 cm in an area
near a large window where sunlight has significantly raised the air tem-
perature.

2. Most commercial sonars use the same diaphragm to generate the sound
and to detect its return. There must be a null time in which the gen-
erating vibrations attenuate before looking at the diaphragm to detect
the return. This usually means that the sonar is blind out to about 25 cm.

3. Many objects absorb sonar and so appear invisible.

4. At a corner there is quite a large shift in the sonar measurements.

5. A first approximation to the range of sensitivity of a sonar is a cone shape out in front of the sonar. This is illustrated in [Figure 3-4 part a]. In fact the area is much more complex, including some small side lobes, as illustrated in [Figure 3-4 part b].

6. Many objects act as mirrors to sonar. As in a house of mirrors it is therefore very difficult to determine just where the sensed object is. Additionally the mirror surfaces themselves are invisible—and they are what the artificial creature will soon collide with.

![Figure 3-4: Sonar Footprint](image)

3.1.9 Video Camera and Transmitter

Mounted on top of the Pioneer AT's console is the microrover's color video camera. The camera has 380 lines of resolution, and has a C/CS style lens mount. The HVTR project is using a Computar 1:1.3 lens. This type of lens was chosen in favor of a wide-angle lens because wide-angle lenses tend to have too much distortion. This distortion would cause incorrect behavior when using the video for position recalibration (see Section 3.3.3). Mounted alongside the camera is a small microphone.

The audio and video signals are wired to a transmitter mounted inside the robot’s console. This transmitter broadcasts the video and audio signals at 2.4 Ghz via an antenna mounted on top of the console.
The transmitter broadcasts a poor quality video signal. Though the specifications say the range is 100 meters, in practice the range extends less than 10 meters when the microrover is in motion. Part of this is due to the fact that the 2.4 Ghz signal is very directional, but the problem primarily stems from the fact that the transmitter as a whole is of poor quality. The video transmitter used in the HVTR project was chosen for its low cost. It will soon be replaced because its video reception is unsatisfactory.

3.1.10 Gyro

Installed in the nose of the Pioneer AT is a Systron-Donner angular rate gyro positioned to detect changes in heading. The output of the rate gyro is an analog voltage proportional to the rate of rotation. Electronics next to the gyro in the nose of the robot low-pass filter the signal to remove aliasing, and then digitize the signal into a 12-bit digital signal. This signal is then integrated by a PIC16C84 microcontroller [15]. The PIC16C84 outputs the relative heading of the vehicle to the Pioneer AT's MC68HC11 microcontroller, but it is the laptop that actually interprets the signal from the PIC16C84. The mechanics of this process and why it is necessary are explained in the Gyro Microcontroller portion of Section 3.2.

The circuitry which processes the analog voltage output of the gyro is the only custom-made component of the HVTR Microrover. Alternate gyro designs come packaged with their own circuitry which provide a simple RS232 serial interface to the gyro. This would have worked well for the HVTR microrover and would have been consistent with the design philosophy of using only off-the-shelf components. The Systron-Donner gyro and its supporting electronics was used because it had previously been used on the EOD-2 robot. The gyro was no longer being used, so it was less expensive to use the existing gyro than to purchase a new one.

The advantages of using a gyro instead of a magnetic compass are the gyro's higher bandwidth and its insensitivity to abnormalities in the Earth's magnetic fields caused by proximity to large metal objects. The disadvantage is that the gyro has a continual and varying drift factor which produces error in the heading determination.

3.1.11 Future Positioning Sensors

The HVTR project is still under development, hence there will likely be additional sensors beyond the wheel encoders and gyro installed to assist in position determination. The most likely new sensor is a magnetic compass. As explained in Section 3.1.10, a compass has low bandwidth and is easily affected by large metal objects, but it lacks the drift that plague all gyros. If the compass is used in conjunc-
tion with the gyro and a carefully crafted Kalman filter (see Section 3.3.2), it should greatly improve the heading determination of the robot.

One final positioning system of interest is a Global Positioning Satellite (GPS) receiver. A dynamic GPS system like the one used by the DSAAV (Appendix A.4) can provide an absolute global position to within 10 cm of accuracy. The main reason it has not yet been considered for the HVTR project is its high cost.

### 3.2 Microrover Software

Code exists in three locations on the HVTR microrover: the gyro’s microcontroller, the Pioneer AT’s microcontroller, and the laptop. Because the robot possesses no autonomy, there is very little processing done onboard the microrover.

#### 3.2.1 Gyro Microcontroller

The actions taken by the PIC16C84 microcontroller in the gyro circuitry are controlled by four digital input lines connected to four of the Pioneer AT microcontroller’s digital output lines. The state of the Pioneer AT microcontroller’s output lines is controlled by the laptop via a RS232 serial connection between them.

Why is the gyro controlled in this way? The gyro microcontroller uses only digital I/O lines, and the Pioneer AT’s microcontroller has several digital I/O lines that are not being used by any devices. Connecting the two microcontrollers via their digital I/O lines was easier than engineering an interface between the laptop and the gyro’s microcontroller. Unfortunately, the code loaded on the Pioneer AT’s microcontroller is proprietary and controlled by RWI. Therefore, it is impossible to program the Pioneer AT’s microcontroller to process the data that it is receiving on its digital input lines. Via the RS232 serial link, the laptop can look at the digital input lines and tell the Pioneer AT microcontroller to change the state of its digital output lines. In short, this process is a software hack to avoid having to greatly modify the gyro circuitry.

Figure 3-5 shows a schematic of the digital I/O handled by the gyro microcontroller [13]. The four digital input lines of the gyro microcontroller controlled by the Pioneer AT are labeled Calibrate, Request, Selector1, Selector2. The five digital output lines of the gyro microcontroller that connect to the digital inputs of the Pioneer AT are labeled Ready, Data1, Data2, Data3, and Data4. The gyro controller has one digital output line connected to the gyro labeled Zero, and there are also twelve digital input lines through which it receives the digital representation of the angular
rotation rate from the analog-to-digital converter.

When a communication link is established between the laptop and the control station, the laptop assumes that the robot is stationary. It then commands the Pioneer AT microcontroller to trigger\(^2\) the \textit{Calibrate} line. On sensing this, the gyro controller triggers the \textit{Zero} line, informing the gyro that it is stationary and therefore should recalibrate itself to output zero angular rotation. The gyro controller also reinitializes its internal heading state to zero.

The gyro microcontroller samples the 12-bit digital signal output by the analog-to-digital converter at a fixed rate. It then integrates this signal to determine the change in heading since the last sample, and updates its internal state regarding the microrover’s heading.

When the laptop needs a new heading value, it commands the Pioneer AT microcontroller to trigger the \textit{Request} line. On sensing this, the gyro controller first sets the \textit{Ready} line low, and then stores its most recent calculation of the heading into a temporary variable. The temporary variable allows the gyro microcontroller to

\(^2\)For the purposes of this thesis, to trigger a digital line means to toggle the signal on the line from low to high, wait some time duration longer than the clock rate of the microprocessor, and then toggle the line back to low again.
continue to update its estimation of heading while the heading is being read by the Pioneer AT microcontroller. Once the heading is ready to be read, the gyro controller sets the Ready line back to high.

The heading is a 16-bit number, but not all 16 bits of the value can be read by the Pioneer AT microcontroller at one time. There are only four data lines connecting the gyro controller and the Pioneer AT microcontroller. Which four bits of the heading are being output is determined by the selector lines. Table 3.3 shows which bits are carried on which data lines for each combination of selector inputs. The laptop, through the Pioneer AT microcontroller, reads four bits, changes the selector lines, reads another four bits, and repeats until it has received all 16 bits of the heading. Using this process, the laptop receives a complete heading value at a rate of about 1 Hz.

<table>
<thead>
<tr>
<th>Selector1</th>
<th>Selector2</th>
<th>Data1</th>
<th>Data2</th>
<th>Data3</th>
<th>Data4</th>
</tr>
</thead>
<tbody>
<tr>
<td>low</td>
<td>low</td>
<td>13th bit</td>
<td>9th bit</td>
<td>5th bit</td>
<td>1st bit</td>
</tr>
<tr>
<td>low</td>
<td>high</td>
<td>14th bit</td>
<td>10th bit</td>
<td>6th bit</td>
<td>2nd bit</td>
</tr>
<tr>
<td>high</td>
<td>low</td>
<td>15th bit</td>
<td>11th bit</td>
<td>7th bit</td>
<td>3rd bit</td>
</tr>
<tr>
<td>high</td>
<td>high</td>
<td>16th bit</td>
<td>12th bit</td>
<td>8th bit</td>
<td>4th bit</td>
</tr>
</tbody>
</table>

Table 3.3: Gyro Data Bit Selection

3.2.2 Pioneer AT Microcontroller

The Pioneer AT Microcontroller is in charge of the low level interaction with the motors, encoders, sonars, and bumpers. It receives directives passed to it by the laptop, and continuously sends information about the state of the devices it controls back to the laptop. The code that performs these functions was written by RWI and cannot be modified by the end-user.

Communication Protocol

Communicating with the Pioneer AT microcontroller is done via RWI defined data packets over a RS232 serial connection. There are 24 different commands that can be sent to the Pioneer AT, of which the HVTR project uses only 9. About every 100 ms, the microcontroller sends a packet that contains all of the information that the microcontroller has gathered from its sensors. Table 3.4 lists the different commands that the HVTR project uses. Table 3.5 details the communication packet used for
sending commands to the Pioneer AT, and Table 3.6 shows the contents of the packet sent by the Pioneer AT [1].

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYNC0</td>
<td>Communication synchronization packet 0</td>
</tr>
<tr>
<td>SYNC1</td>
<td>Communication synchronization packet 1</td>
</tr>
<tr>
<td>SYNC2</td>
<td>Communication synchronization packet 2</td>
</tr>
<tr>
<td>PULSE</td>
<td>Communication pulse to verify connection, sent every 1.2 s</td>
</tr>
<tr>
<td>CLOSE</td>
<td>End communication</td>
</tr>
<tr>
<td>SET0</td>
<td>Zero the encoder counters</td>
</tr>
<tr>
<td>VEL</td>
<td>Set translational velocity</td>
</tr>
<tr>
<td>RVEL</td>
<td>Set rotational velocity</td>
</tr>
<tr>
<td>DIGOUT</td>
<td>Set digital output lines</td>
</tr>
</tbody>
</table>

Table 3.4: Commands Used by HVTR

<table>
<thead>
<tr>
<th>Component</th>
<th>Bytes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Header</td>
<td>2</td>
<td>Packet header</td>
</tr>
<tr>
<td>Byte Count</td>
<td>1</td>
<td>Number of command bytes plus checksum</td>
</tr>
<tr>
<td>Command Number</td>
<td>1</td>
<td>Client command number</td>
</tr>
<tr>
<td>Argument Type</td>
<td>1</td>
<td>Command argument type (integer or string)</td>
</tr>
<tr>
<td>Argument</td>
<td>n</td>
<td>Command argument</td>
</tr>
<tr>
<td>Checksum</td>
<td>2</td>
<td>Packet integrity checksum</td>
</tr>
</tbody>
</table>

Table 3.5: Command Packet Structure

**Velocity and Rotation Processing**

To command the microrover to move, the HVTR project makes use of the VEL and RVEL commands to command translational velocity and rotational velocity. The Pioneer AT microcontroller must convert these values into rotational velocities for the left and right wheels. Because the microrover has differential steering, there are some easy formulas for describing the forward velocity and rotational velocity of the robot. First, some variable definitions.
<table>
<thead>
<tr>
<th>Component</th>
<th>Bytes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Header</td>
<td>2</td>
<td>Packet header</td>
</tr>
<tr>
<td>Byte Count</td>
<td>1</td>
<td>Number of command bytes plus checksum</td>
</tr>
<tr>
<td>Status</td>
<td>1</td>
<td>Motor status (off, stopped, or moving)</td>
</tr>
<tr>
<td>X Position</td>
<td>2</td>
<td>X position of robot</td>
</tr>
<tr>
<td>Y Position</td>
<td>2</td>
<td>Y position of robot</td>
</tr>
<tr>
<td>Heading</td>
<td>2</td>
<td>Heading of robot</td>
</tr>
<tr>
<td>Left Velocity</td>
<td>2</td>
<td>Velocity of left wheels</td>
</tr>
<tr>
<td>Right Velocity</td>
<td>2</td>
<td>Velocity of right wheels</td>
</tr>
<tr>
<td>Battery</td>
<td>1</td>
<td>Battery voltage</td>
</tr>
<tr>
<td>Unused Data</td>
<td>7</td>
<td>State of unused Pioneer AT functionality</td>
</tr>
<tr>
<td>New Sonar Readings</td>
<td>1</td>
<td>Number of new sonar readings</td>
</tr>
<tr>
<td>Sonar Number</td>
<td>1</td>
<td>Sonar identification number</td>
</tr>
<tr>
<td>Sonar Range</td>
<td>2</td>
<td>Distance to ranged object</td>
</tr>
<tr>
<td>More Sonars</td>
<td>3n</td>
<td>Two sonar fields repeated as necessary</td>
</tr>
<tr>
<td>Unused Data</td>
<td>3</td>
<td>State of unused Pioneer AT functionality</td>
</tr>
<tr>
<td>Digital Inputs</td>
<td>1</td>
<td>Status of digital input lines</td>
</tr>
<tr>
<td>Digital Outputs</td>
<td>1</td>
<td>Status of digital output lines</td>
</tr>
<tr>
<td>Checksum</td>
<td>2</td>
<td>Packet integrity checksum</td>
</tr>
</tbody>
</table>

Table 3.6: Pioneer AT Return Packet Structure
$v_c$ = Commanded translational velocity

$\omega_c$ = Commanded rotational velocity

$\omega_r$ = Right wheel rotational velocity

$\omega_l$ = Left wheel rotational velocity

$r$ = Radius of the wheels

$d$ = Distance between the center lines of the wheels

The instantaneous forward velocity of the microrover is [2]

$$v = \frac{r(\omega_r + \omega_l)}{2}$$

(3.1)

The instantaneous rotational velocity of the microrover is

$$\omega = \frac{r(\omega_r - \omega_l)}{d}$$

(3.2)

Combining Equation 3.1 and Equation 3.2 makes it possible to solve for $\omega_r$ and $\omega_l$.

$$\omega_r = \frac{2v_c + d\omega_c}{2r}$$

(3.3)

$$\omega_l = \frac{2v_c - d\omega_c}{2r}$$

(3.4)

Once the Pioneer AT microcontroller has calculated the necessary rotational velocities for the wheels, it does not immediately try to command the motors to spin at that speed. Instead, it accelerates and deaccelerates between the current velocity and the target velocity by means of a trapezoidal velocity profile as shown in Figure 3-6 [1]. Because the HVTR project needs responsiveness as opposed to smooth velocity transitions, the acceleration rate is set as high as possible without causing extreme jerkiness.

![Figure 3-6: Trapezoidal Velocity Profile](image-url)
To command the wheels to achieve the correct velocity, a PID feedback loop is used. PID stands for Proportional Integral Differential, which is a class of feedback loops that can be very useful in velocity control situations. The proportional part of the feedback system controls the responsiveness of the drive system. The differential component dampens oscillation and overshoot. The integral portion adjusts for residual errors.

**Encoder Processing**

The Pioneer AT microcontroller keeps an internal representation of its position and heading based on analysis of the wheel encoders. This set of computations is often called dead reckoning. To do this, it begins with the knowledge of how many encoder counts occur per wheel revolution, \( E_{\text{rev}} \). For the Pioneer AT, \( E_{\text{rev}} \) is 100. The microcontroller can then compute the actual rotational velocity of each wheel based on \( t \), the time elapsed since the last update of position and heading, \( \Delta E_r \), the change in encoder counts of the right wheels since the last update, and \( \Delta E_l \), the change in encoder counts for the left wheels.

\[
\omega_r = \frac{2\pi \Delta E_r}{t E_{\text{rev}}} \tag{3.5}
\]

\[
\omega_l = \frac{2\pi \Delta E_l}{t E_{\text{rev}}} \tag{3.6}
\]

Before these rotational velocities can be used any further, an equation describing \( R \), the radius of the circle that the center of the vehicle follows, is necessary [2].

\[
R = \frac{d(\omega_r + \omega_l)}{2(\omega_r - \omega_l)} \tag{3.7}
\]

Next, the instantaneous rotational velocity of the robot found by Equation 3.2 is used to compute \( \Delta \theta \), the change in heading.

\[
\Delta \theta = \omega t \tag{3.8}
\]

The change in heading can be used to calculate the robot’s displacement along a circle. Be warned, these values do not yet represent the change in position in global coordinates. [3]

\[
\Delta x = R \sin \Delta \theta \tag{3.9}
\]
$$\Delta y = R(1 - \cos \Delta \theta)$$ \hfill (3.10)

Finally, the new \( \theta_i, x_i, \) and \( y_i \) values can be calculated using the \( \theta_{i-1}, x_{i-1}, \) and \( y_{i-1} \) from the last sample. \[3\]

\[\theta_i = \theta_{i-1} + \Delta \theta\] \hfill (3.11)

\[x_i = x_{i-1} + \Delta x \cos \theta_{i-1} - \Delta y \sin \theta_{i-1}\] \hfill (3.12)

\[y_i = y_{i-1} + \Delta x \sin \theta_{i-1} + \Delta y \cos \theta_{i-1}\] \hfill (3.13)

**Sonar Processing**

The Pioneer AT contains circuitry to provide the microcontroller with ranging information for each of the sonars. No extra processing is done to this data. New sonar ranges are simply packaged in the communication packet that is broadcast to the control station.

### 3.2.3 Laptop

The Laptop is using Red Hat Linux 5.1 for its operating system. A program written in C and running continuously handles the gyro and communication processing tasks.

**Gyro Processing**

As explained in Section 3.2.1: Gyro Microcontroller, the laptop reads the Pioneer AT microcontroller's digital inputs while instructing the Pioneer AT microcontroller to change its digital outputs in the correct order to retrieve what the gyro microcontroller has determined the heading to be. Once the laptop has converted the 16 bits back into a number, it packages the gyro heading into the communication protocol so the control station can receive the data.

**Communication Processing**

The laptop is connected via one serial port to the robot's modem, which communicates with the control station. Another serial port connects to the Pioneer AT's microcontroller. All communications from the control station are sent to the Pioneer AT microcontroller by copying the data from one serial port to the other. All communications from the Pioneer AT microcontroller are sent to the control station, but
not before the laptop places gyro heading information into the status packet in two of the bytes labeled as **Unused Data** in Table 3.6. After updating the packet, the checksum is recomputed to reflect the changes to the packet.

When the laptop needs to send a packet to update the digital outputs of the Pioneer AT microcontroller, the packet must be carefully inserted between two other packets bound for the microcontroller.

### 3.3 Control Station Software

The HVTR control station is written in C, and uses the Xt, Xlib, and Motif graphics libraries for the graphical interface. It runs on a Pentium II 233 desktop PC using Red Hat Linux 5.1 for its operating system. For I/O, the control station uses a mouse, a joystick, a Hauppauge TV tuner connected to a 2.4 Ghz video receiver, and a Proxim radio modem like the one used on the robot (see Section 3.1.5: Modem). This platform is less expensive than more specialized systems, but provides ample computing power for control station applications. Figure 3-7 illustrates the control station in action.

![Figure 3-7: HVTR Control Station](image)
3.3.1 Graphical User Interface

The control station graphical user interface (GUI) uses 16 bit video depth (up to 65536 simultaneous colors) and occupies a full-screen window set at a resolution of 1152x864 pixels. Except for the video window, the GUI updates at a rate of about 8.4 Hz. The video window updates at 60 Hz because it is controlled by the TV tuner's special video hardware which writes directly to video memory.

The GUI is divided into seven fixed windows, each responsible for displaying specific data or processing inputs from the user. Figure 3-8 shows the relative locations and sizes of the seven fixed windows.

![Diagram of window layout](image)

Figure 3-8: Window Layout of HVTR Control Station

Navigation Window

The navigation window is the largest and most important window in the GUI. It occupies all of the upper left corner of the GUI. The navigation window displays a picture or drawing of the terrain that the microrover is being driven on. Currently, the control station only supports the display of a single 800x600 pixel image, but a future modification should allow it to extract the relevant portion of a larger image or mosaic multiple smaller images, and then re-center the robot and display the relevant
portion of the terrain whenever the microrover nears the boundaries of the currently displayed image.

![Figure 3-9: Navigation Window](image)

There are many graphical elements that are overlaid on top of the navigation window. Each of them can be seen in Figure 3-9. The first and most necessary is the robot icon. This icon is scaled in size\(^3\) and rotated so that it accurately illustrates the robot's footprint and heading on the terrain image. To allow easy identification of the front of the robot icon, there is a colored rectangle on the icon that represents the elevated console at the front of the robot. Before a communication link is established between the control station and the microrover, this rectangle is colored grey. Once communication is established, this rectangle is colored red. The final elements of the robot icon shown on the navigation window are the bumpers. In their default state, they are colored black. When a bumper is compressed, the icon's bumper is then colored red and an audio warning sounds.

To provide the user with a reference of where the robot has already traveled, a yellow line connecting the last 1000 screen positions of the robot icon is drawn. The starting location of the robot is represented by a small red triangle, with the narrowest

\(^3\)The scale of the terrain image is stored in a file that is read when the control station is started.
point of the triangle pointing to the initial heading of the microrover.

Surrounding the robot icon is a red oval. This oval represents the system's uncertainty of its actual position. This signifies to the operator that the robot could actually be located anywhere within the red oval. To lessen the probability of a collision, the operator should avoid allowing any portion of the red oval from intersecting with an obstacle. The size and growth rate of the oval is determined by the Kalman filter (see Section 3.3.2). The red oval cannot be seen in Figure 3-9 because the robot has not traveled far enough to accumulate appreciable errors. The oval can be shrunk by performing a position recalibration (see Section 3.3.3).

The final graphical elements that can be displayed within the navigation window are the colored circles used for correlation of visual references when repositioning the robot via the live video (see Section 3.3.3). When in the position calibration mode, clicking a mouse button in the navigation window places a circle identifying a point to be used for the position recalculation. The first circle is colored red, the second is yellow, and the third circle is green.

Clicking a mouse button in the navigation window when not in position calibration mode performs a rough positioning that is useful at the beginning of a mission. If the operator knows exactly where the robot is going to begin its mission, and which direction it will be facing, the operator can click and hold a mouse button on the location where the robot icon is to be placed. While the mouse button is held down, rotating the mouse cursor around the newly placed robot icon will rotate the icon to face the cursor. This allows an initial heading to be set. This positioning scheme is not to be confused with the position calibration mode detailed in Section 3.3.3.

**Zoom Window**

Because some objects in the navigation window might be tough to discern due to their small scale, a 320x320 pixel zoom window is provided to the right of the navigation window. The view in this window is always centered, and the scale is always one pixel to every inch.

As can be seen in Figure 3-10, all graphical elements that can be found in the navigation window are drawn in the zoom window at the one pixel to every inch scale. Clicking a mouse button in the zoom window has the same functionality as if the button was clicked in the navigation window.

**Video Window**

The video window is a 320x240 pixel window on the right side of the GUI displaying
video from the robot’s onboard camera (see Section 3.1.9) at a 60 Hz refresh rate. Only one set of graphical elements ever overlay the video window, and they do so only when the robot is being repositioned using the visual feature correlation mode (see Section 3.3.3: Position Recalibration). Like in the navigation window, a colored circle is drawn each time a mouse button is clicked within the video window. The first circle is red, the second is yellow, and the third circle is green. Figure 3-11 shows a frame of video taken from the microrover’s perspective during a position recalibration. The red circle is obscured on the left side of the planter, but the yellow circle on the near corner of the planter and the green circle on the corner of the bench are visible.

**Sensor Window**

The sensor window is a 320x245 pixel window in the lower right corner of the screen that presents information from many of the microrover’s sensors. Figure 3-12 is a screenshot of this window.

The sensor window is dominated by a graphical representation of the sensing zones of the seven sonars located on the microrover. These sensing cones are completely grey when a sonar is receiving no echos. If there is an echo, the cone is filled with yellow to a distance proportional to the range of the sonar echo. If the sonar echo is within 30 cm of the sonar, the background of the cone turns to red, and an audio warning sounds. The combination of these two behaviors should alert the operator to a nearby obstacle. The seven cones rotate as the microrover rotates, so the main
Figure 3-11: Video Window

Figure 3-12: Sensor Window
robot icon in the navigation window, the zoomed icon in the zoom window, and the
sonar cones in the sensor window all rotate in a consistent manner.

If the front bumper on the micro rover is compressed, a half circle behind the sonar
cones illuminates in red to provide a very visual alert to the condition. This is in
addition to the coloring of the bumpers red in the navigation and zoom windows,
and the audio warning. If the back bumper is compressed, a half circle illuminates
opposite of the sonar cones.

In the corners of the sensor window are text outputs of several state variables.
These are useful for debugging purposes, but are not meant for general use. The
variables shown are the Pioneer AT microcontroller’s internal x and y coordinates,
the robot icon’s x and y screen coordinates, the robot’s internal heading, and the
gyro heading.

Battery Voltage Window

The battery voltage window is a very simple window in the lower center of the
screen, occupying 132x215 pixels. It graphically illustrates the voltage being received
by the robot’s MC68HC11 microcontroller with a vertical bar of size proportional to
the voltage. This window can be seen in Figure 3-13. When the voltage is greater
than or equal to 12 V, the vertical bar is drawn in green, indicating that the electrical
system has sufficient voltage for all devices to work properly. When the voltage drops
below 12 V, the vertical bar turns red, indicating that the voltage is at or close to the
level where some of the electrical devices on the robot either degrade in performance,
or shut down altogether. The voltage is also displayed as a decimal number in yellow
text at the top of the bar with 0.1 V precision.

Why does the battery voltage receive such a prominent location in the control
station? During experimentation with the HVTR system, it became apparent how
useful the voltage level at the processor is in diagnosing certain problems. When an
electrical device on the robot, such as a motor, draws a lot of current, the output
voltage of the battery drops. This drop in voltage is significant enough to be seen on
the control station’s battery voltage indicator.

An example of a situation that illustrates the effectiveness of the voltage indicator
occurs when the robot is driven into an obstacle in such a way that the robot can
no longer move in the direction that the operator is commanding it to move. In
most cases, the bumper would be compressed by the obstacle, and the control station
would activate the visual and audio warnings that indicate a bumper is in contact
with something, thereby alerting the user to the robot’s situation. Occasionally,
the obstacle catches the robot in such a way that the bumper is not compressed. Without the bumper’s input, the operator might not know that the robot is stuck on an obstacle, and wonder why the control inputs are not being executed by the robot. Fortunately, the changing of the battery voltage indicator from green to red because of the extra load on the battery is easily observed, even when the operator is focusing on other parts of the screen.

**Velocity Window**

The velocity window is a 516x215 window that contains three colored horizontal bars which illustrate the microrover’s speed as shown in Figure 3-14. The horizontal bars represent the velocity of the left wheels, the translational velocity of the whole vehicle, and the velocity of the right wheels, in that order from top to bottom. Forward velocities are drawn in green, and backward velocities are drawn in red. Next to each bar, a textual readout of the velocity is also given. Around the bar, tick marks provide another reference for determining speed.

Some simple heuristics for analyzing these three bars can be constructed. If the top bar is approximately the same size as the lowest bar, then the robot is traveling roughly straight ahead. If they are different, then the robot is turning. The greater the difference in their length, the lesser the radius of the robot’s turn (meaning it is turning more sharply). If the top bar is green and the bottom bar is red, or vice
versa, then the robot is using its differential steering ability to turn within the radius of the vehicle. If the center bar reads zero during the turn, then the robot is turning in place.

This window is positioned on the bottom left-center of the screen, and occupies what seems to be a large space for the simple set of values it displays. This was done because the HVTR project, as its name describes, is concerned with maximizing speed. Since speed is such an important issue in the design, its indicator should receive a sizeable portion of the screen.

**Button Window**

The button window occupies 132x215 pixels in the lower left hand corner of the GUI. The control station is a very straightforward program, so it does not need a system of menus to be user friendly. Instead, the button window provides three buttons, the only buttons necessary in the whole program. Figure 3-15 shows a picture of this simple window.

The first button toggles the state of the data logger. When the logger is off, the button is raised, and when the logger is on, the button is depressed. Hitting the 'I' button also toggles the state of this button. When logging, the control station records the following seven variables at approximately 8.4 Hz.

- Timestamp
- X Position
- Y Position
- Heading
• Left Side Wheel Velocity

• Right Side Wheel Velocity

• Joystick X Position (correlates to commanded rotational velocity)

• Joystick Y Position (correlates to commanded velocity)

The second button toggles the position recalibration mode. The keyboard shortcut for this command is ‘r’. More detail into the functional qualities of the position recalibration mode can be found in Section 3.3.3.

The final button is a quit button. It closes the data logging file if one is open, tells the robot that it is closing down communications, and then exits the program. Its keyboard shortcut is ‘q’.

3.3.2 Kalman Filter

A Kalman filter is a computational algorithm that uses multiple inputs to deduce an optimal estimate of a system’s behavior. In the HVTR project, the data from all of the positioning sensors is input into a Kalman filter that returns an estimate of the microrover’s actual position, as well as a quantitative measure of uncertainty in the position calculation. This uncertainty measure dictates the size of the uncertainty
oval that surrounds the robot icon in the navigation window of the control station GUI (see Section 3.3.1).

3.3.3 Position Recalibration

Occasionally, it is necessary to recalibrate where the robot icon is located on the terrain image with where the microrover is truly located. To place the control station in recalibration mode, the operator clicks on the position recalibration mode button or types 'r', the keyboard shortcut.

Upon entering the position recalibration mode, if the microrover was not already stopped, it is commanded to stop moving as quickly as possible. While in this mode, all joystick commands are ignored. Once the robot has stopped moving, the video is frozen into a static image. This is done to simplify the process of drawing graphical elements on top of the video image.

Now the operator must look at the terrain image and the frozen video image and try to identify three common features found in both images. For example, static features of the environment such as a corner of a curb or sidewalk, a pole, or a tree trunk are useful. The operator must then click on the three common features in either the navigation window (equivalent functionality in the zoom window) or the video window. Each mouse click draws a colored circle. Then, in the window that was not used for the first three circles, the same three objects need to be selected in the same order. By measuring the differences in angles between the objects, it is possible to compute the exact position of the microrover based on what is displayed in the video window. Once the new position is calculated, the robot icon is placed there, and the position recalibration mode is exited, erasing the circles from the screen.

The position recalibration operation can be aborted by clicking on the position recalibration button, or by typing the 'r' key. This will erase the circles from the screen, restore joystick control, and allow robot movement.

3.3.4 Evolution of the Computer Interface

The GUI (Section 3.3.1) as described in the implementation has gone through a number of revisions. Some revisions simply added new features, while others addressed mistakes in older versions. A history of these revisions and the reasons behind the changes is insightful in analyzing the human computer interface.

1. Initial version. The first version was simply a terrain image and an icon. Control inputs were made using the joystick.
2. **Added zoom window.** It became apparent that at some scale factors, the robot icon was too small to maneuver easily. Therefore, the zoom window with its fixed scale was added. At this time in the development, the zoom window was a completely separate window that could be moved independently of the navigation window.

3. **Added sensor window and audio alerts.** This revision began as an effort to retrieve the state of the bumpers. Since the control station was receiving information about both the bumpers and sonars, it seemed reasonable to add sonar functionality at the same time as the bumper addition. A separate new window was created to graphically display the sonar and bumper data.

4. **Added trail of past screen positions.** Situations often occur where it is useful to know where the robot has been. The trail was originally added for its help in assessing and improving the dead reckoning ability of the microrover. The robot would be driven through several turns and straightaways before returning it to its starting point. If the yellow trail did not lead back to where the robot started, some insight can be gathered into the magnitude of the dead reckoning error by how far the starting and ending locations differed.

5. **Error circle added.** Early experimentation showed that there is often some error between the position of the robot icon onscreen and its actual position within its environment. An error circle illustrating the increasing errors over time seemed a very useful tool in determining how close the robot can safely approach obstacles.

6. **Video window added.** The need for an accurate method of recalibrating on-screen position prompted the feature correlation technique. This necessitated the addition of a separate video window. To get the video window to work properly, the color depth had to be increased to 16 bits, necessitating a decrease in screen resolution to 1152x864.

7. **Refined windows into a consistent interface.** With the zoom, sensor, and video interfaces as stand-alone windows, the interface lacked cohesion. The current single window, multiple subwindow interface was designed. The extra space at the bottom was turned into a velocity window and a voltage window.

8. **Made zoom window consistent with the navigation window.** Up until this point, the navigation window had all of the graphics and mouse capabilities.
To provide consistency among components, all graphics overlays and mouse inputs were programmed so they would function in both the navigation window and the zoom window.

9. **Added a data logger.** The logger spools data to disk in a format that can be read easily by any spreadsheet program.

10. **Shrunk velocity window to make room for button window.** The interface had no buttons or menus before now. The operator had to know the key shortcuts to recalibrate position, log data, or quit the control station. Now a more novice operator can use these features.
Chapter 4

HVTR Testing

This chapter explains the testing recently performed on the HVTR system. Contents include

- an overview of how the test environment was chosen,
- a description of the testing methodology,
- detailed results gathered from the testing, and
- an analysis of the results.

4.1 Testing Environment

Creating a series of tests for this thesis was an exercise in compromise. The goal of the HVTR project is to control the microrover at high speed. This goal is best suited for situations involving large, obstacle-sparse environments. The HVTR microrover, at its current stage of development, has two shortcomings that in combination make testing in a large, obstacle-sparse environment impossible.

First, the only sensors that the robot is currently using to determine its position are its encoders. The gyro is installed and being tested in combination with the Kalman filter, but not enough calibration of the Kalman filter has been performed to allow its output to be used as a gyro-corrected dead reckoning estimate of the microrover’s position. This in itself is not crippling to tests over large areas, but position recalibration would have to occur often.

Second, the range of the video transmitter is less than ten meters. This problem alone could be partially overcome if the position system was accurate, but its current degraded performance makes determining the position of the robot at great distances impossible. Since a high speed microrover’s mission would often involve traversing
large distances, a higher quality video transmitter would be necessary for the system to be usable at greater distances.

It was therefore necessary to conduct tests in a small, obstacle-rich environment so that the video transmitter would always be in range. The use of a small, obstacle-sparse environment was avoided because it would be too easy for the operator to memorize the control inputs needed to control the robot from its starting location to its goal location. For example, imagine navigating the small, obstacle-sparse environment involved the following actions:

1. Full speed forward for two seconds.
2. Gentle right turn for three seconds, continuing at maximum velocity.
3. End turn, but continue at full speed for a second.
4. Harder left turn for a little over a second.
5. Stop.

After several attempts, an operator would be able to complete the course on memory alone. The test would no longer be fairly analyzing the operator's capabilities as a function of the system's control technique, but instead as a function of the operator's ability to repeat a simple series of hand motions.

To maintain the best dead reckoning abilities of the robot, the tests were performed on a flat carpeted floor. This allowed the positioning system to be as accurate as possible, causing position recalibration to become unnecessary in the tests. This was advantageous because even when the video transmitter was within range of the video receiver, the reception was often poor, which would have made position recalibration very difficult and error-prone.

Figure 4-1 gives a floorplan of the room that was used to conduct the HVTR testing, and Figure 4-2 shows a picture of the obstacles in the room, taken from near the starting location. Notice that the obstacles not only block the path of the microrover, but they also obscure parts of the course from being seen by an operator who is controlling by sight alone. The arrows assist users during the tests that use the microrover's video camera.

4.2 Testing Methodology

Six test subjects were asked to maneuver the microrover from its starting location to the goal location. They did so using three different control techniques:
Figure 4-1: Floorplan of Test Course
1. **Sight:** The operator stands in a predefined location in the room and maneuvers the robot using nothing more than their visual faculties. In the tables of test data, these tests are referred to as S1, S2, and S3.

2. **Video:** The operator sits in front of a television and navigates the microrover using only the video from the robot mounted camera. The test results label these tests as V1, V2, and V3.

3. **Computer:** The operator sits in front of the computer and traverses the course using only the tools provided by the control station. C1, C2, and C3 mark these tests in the tables of results.

Before conducting these tests, each operator was given about 20 minutes to familiarize themselves with the three control techniques being tested. Once they were ready, they performed each test a minimum of five times. The tests with the fastest three times for each of the control techniques were used in this thesis for analysis, and the remaining test data was discarded.

The first test subject performed the sight tests first, then the video tests and finally the computer tests. The next subject did the sight tests followed by the computer tests
and lastly the video tests. For each new subject, the order of the tests was changed so that each of the six test subjects performed one of the six possible orderings of the three tests. This prevents the increased familiarity with the robot in the latter test cases from skewing the data for any one class of tests.

4.3 Testing Results

Each test subject was asked three questions to discover if they might have previous experience or attributes that could help them in the experiment. Their height plays a role in the sight tests. Their experiences both with video games and remote control vehicles might make them more comfortable with a joystick, or with controlling vehicles of the micro Rover’s scale.

- **Test Subject 1** is a male of height 5'8” who occasionally plays video games and has had no experience with remote control vehicles. Table 4.1 shows his results.

- **Test Subject 2** is a male of height 5'10” who occasionally plays video games and has had experience with remote control airplanes. Table 4.2 shows his results.

- **Test Subject 3** is a female of height 5'6” who has very little exposure to video games and has had no experience with remote control vehicles. Table 4.3 shows her results.

- **Test Subject 4** is a male of height 6'3” who occasionally plays video games and has had experience with remote control cars. Table 4.4 shows his results.

- **Test Subject 5** is a male of height 5'11” who occasionally plays video games and has had very little experience with remote control cars. Table 4.5 shows his results.

- **Test Subject 6** is a male of height 5'11” who has not recently played video games and has had experience with remote control helicopters. Table 4.6 shows his results.

To simplify comparisons between subjects, each subject’s results for each type of test were averaged and placed in a group of tables. Table 4.7 and Table 4.8 show this data grouped by test subjects. Table 4.9 and Table 4.10 show the same data arranged by type of test as opposed to by test subject.

The tables of results are given in terms of nine metrics.

- **Time:** This is the time in seconds needed to travel from the starting location to the goal location.
• **Collisions:** This is the number of times the microrover collided with an obstacle or wall.

• **Distance From Goal:** This is the distance, in meters, from the center of the robot to the center of the goal destination.

• **Total Distance:** This is the total distance, in meters, that the microrover traveled. A total distance of 19 m or less suggests that the operator navigated the course well. Very low distances indicate that the operator navigated very close to the obstacles at all times with little room for error. The main cause of increased total distance was the misjudging of the width of the robot. The operator would run the corner of the robot into an obstacle, be forced to reverse the microrover (adding unnecessary distance), and then attempting the passage again.

• **Total Rotation:** This is a measure of how much turning was necessary to navigate the course, given in degrees. If the course is navigated with sharp 90 degree turns, the total rotation will be 540°. Lesser values suggest the operator cut closer to the obstacles, and higher values suggest the operator had trouble choosing the heading they wanted to travel at.

• **Average Velocity:** This is the average velocity in m/s at which the microrover navigated the course.

• **Maximum Velocity:** This is the maximum velocity attained during the test, given in m/s.

• **Average VEL:** VEL is the command for setting the velocity of the robot (see Table 3.4). Of the velocities the robot was commanded to attain, this is the average, given as a percent of the maximum velocity. Because the robot does not necessarily obtain the velocity commanded by the operator, this measure is less interesting as a velocity value and more interesting as an indicator of the operator’s control style because the VEL values are directly linked to the operator’s joystick position.

• **Average RVEL:** RVEL is the command for setting the rotational velocity of the robot (see Table 3.4). Like the average VEL, this value is given as a percent of the maximum rotational velocity, and is most interesting as an indicator of the operator’s control style because it too is linked to the operator’s joystick position.
<table>
<thead>
<tr>
<th>Test</th>
<th>Time (s)</th>
<th>Collisions</th>
<th>Distance from Goal (m)</th>
<th>Total Distance (m)</th>
<th>Total Rotation (°)</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>20.64</td>
<td>2</td>
<td>0.48</td>
<td>17.90</td>
<td>681.0</td>
</tr>
<tr>
<td>S2</td>
<td>19.56</td>
<td>1</td>
<td>0.48</td>
<td>17.22</td>
<td>683.1</td>
</tr>
<tr>
<td>S3</td>
<td>19.56</td>
<td>2</td>
<td>0.15</td>
<td>18.47</td>
<td>605.4</td>
</tr>
<tr>
<td>V1</td>
<td>54.18</td>
<td>1</td>
<td>0.64</td>
<td>19.38</td>
<td>1151.4</td>
</tr>
<tr>
<td>V2</td>
<td>49.46</td>
<td>1</td>
<td>0.28</td>
<td>19.06</td>
<td>1015.0</td>
</tr>
<tr>
<td>V3</td>
<td>59.56</td>
<td>4</td>
<td>0.38</td>
<td>23.37</td>
<td>1074.0</td>
</tr>
<tr>
<td>C1</td>
<td>39.70</td>
<td>1</td>
<td>0.48</td>
<td>20.33</td>
<td>728.4</td>
</tr>
<tr>
<td>C2</td>
<td>34.59</td>
<td>1</td>
<td>0.23</td>
<td>18.51</td>
<td>667.3</td>
</tr>
<tr>
<td>C3</td>
<td>38.25</td>
<td>0</td>
<td>0.28</td>
<td>18.93</td>
<td>764.7</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Test</th>
<th>Average Velocity (m/s)</th>
<th>Maximum Velocity (m/s)</th>
<th>Average VEL (%)</th>
<th>Average RVEL (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>0.815</td>
<td>1.379</td>
<td>86.9</td>
<td>84.6</td>
</tr>
<tr>
<td>S2</td>
<td>0.860</td>
<td>1.383</td>
<td>86.1</td>
<td>80.6</td>
</tr>
<tr>
<td>S3</td>
<td>0.855</td>
<td>1.376</td>
<td>91.8</td>
<td>81.0</td>
</tr>
<tr>
<td>V1</td>
<td>0.332</td>
<td>1.438</td>
<td>58.6</td>
<td>77.7</td>
</tr>
<tr>
<td>V2</td>
<td>0.355</td>
<td>1.424</td>
<td>66.9</td>
<td>79.6</td>
</tr>
<tr>
<td>V3</td>
<td>0.281</td>
<td>1.411</td>
<td>67.3</td>
<td>80.9</td>
</tr>
<tr>
<td>C1</td>
<td>0.501</td>
<td>1.432</td>
<td>68.9</td>
<td>78.6</td>
</tr>
<tr>
<td>C2</td>
<td>0.513</td>
<td>1.418</td>
<td>76.1</td>
<td>75.1</td>
</tr>
<tr>
<td>C3</td>
<td>0.466</td>
<td>1.411</td>
<td>74.7</td>
<td>76.5</td>
</tr>
</tbody>
</table>

Table 4.1: Subject 1 Test Results
<table>
<thead>
<tr>
<th>Test</th>
<th>Time (s)</th>
<th>Collisions</th>
<th>Distance from Goal (m)</th>
<th>Total Distance (m)</th>
<th>Total Rotation (°)</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>26.40</td>
<td>1</td>
<td>0.15</td>
<td>20.48</td>
<td>718.2</td>
</tr>
<tr>
<td>S2</td>
<td>20.16</td>
<td>2</td>
<td>0.23</td>
<td>18.22</td>
<td>627.9</td>
</tr>
<tr>
<td>S3</td>
<td>16.92</td>
<td>1</td>
<td>0.20</td>
<td>16.87</td>
<td>575.2</td>
</tr>
<tr>
<td>V1</td>
<td>36.27</td>
<td>0</td>
<td>0.43</td>
<td>18.98</td>
<td>756.6</td>
</tr>
<tr>
<td>V2</td>
<td>33.89</td>
<td>4</td>
<td>0.36</td>
<td>25.24</td>
<td>631.4</td>
</tr>
<tr>
<td>V3</td>
<td>25.81</td>
<td>0</td>
<td>0.23</td>
<td>17.43</td>
<td>593.1</td>
</tr>
<tr>
<td>C1</td>
<td>44.22</td>
<td>2</td>
<td>0.30</td>
<td>20.11</td>
<td>737.6</td>
</tr>
<tr>
<td>C2</td>
<td>49.28</td>
<td>1</td>
<td>0.20</td>
<td>19.72</td>
<td>707.7</td>
</tr>
<tr>
<td>C3</td>
<td>37.94</td>
<td>0</td>
<td>0.13</td>
<td>21.05</td>
<td>871.9</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Test</th>
<th>Average Velocity (m/s)</th>
<th>Maximum Velocity (m/s)</th>
<th>Average VEL (%)</th>
<th>Average RVEL (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>0.681</td>
<td>1.420</td>
<td>89.3</td>
<td>72.4</td>
</tr>
<tr>
<td>S2</td>
<td>0.872</td>
<td>1.432</td>
<td>80.1</td>
<td>70.7</td>
</tr>
<tr>
<td>S3</td>
<td>0.977</td>
<td>1.377</td>
<td>90.6</td>
<td>69.5</td>
</tr>
<tr>
<td>V1</td>
<td>0.506</td>
<td>1.385</td>
<td>47.8</td>
<td>53.3</td>
</tr>
<tr>
<td>V2</td>
<td>0.578</td>
<td>1.353</td>
<td>67.6</td>
<td>49.6</td>
</tr>
<tr>
<td>V3</td>
<td>0.652</td>
<td>1.358</td>
<td>58.2</td>
<td>53.9</td>
</tr>
<tr>
<td>C1</td>
<td>0.442</td>
<td>1.386</td>
<td>39.7</td>
<td>45.7</td>
</tr>
<tr>
<td>C2</td>
<td>0.354</td>
<td>1.018</td>
<td>35.8</td>
<td>36.0</td>
</tr>
<tr>
<td>C3</td>
<td>0.533</td>
<td>1.349</td>
<td>53.9</td>
<td>51.9</td>
</tr>
</tbody>
</table>

Table 4.2: Subject 2 Test Results
<table>
<thead>
<tr>
<th>Test</th>
<th>Time (s)</th>
<th>Collisions</th>
<th>Distance from Goal (m)</th>
<th>Total Distance (m)</th>
<th>Total Rotation (°)</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>52.66</td>
<td>1</td>
<td>0.46</td>
<td>17.85</td>
<td>517.5</td>
</tr>
<tr>
<td>S2</td>
<td>50.62</td>
<td>1</td>
<td>0.33</td>
<td>19.54</td>
<td>1172.1</td>
</tr>
<tr>
<td>S3</td>
<td>36.33</td>
<td>1</td>
<td>0.43</td>
<td>17.30</td>
<td>966.8</td>
</tr>
<tr>
<td>V1</td>
<td>116.66</td>
<td>3</td>
<td>0.10</td>
<td>23.73</td>
<td>1337.3</td>
</tr>
<tr>
<td>V2</td>
<td>85.63</td>
<td>1</td>
<td>0.10</td>
<td>21.98</td>
<td>707.3</td>
</tr>
<tr>
<td>V3</td>
<td>102.40</td>
<td>4</td>
<td>0.05</td>
<td>26.05</td>
<td>930.2</td>
</tr>
<tr>
<td>C1</td>
<td>100.21</td>
<td>0</td>
<td>0.38</td>
<td>17.80</td>
<td>615.2</td>
</tr>
<tr>
<td>C2</td>
<td>106.36</td>
<td>1</td>
<td>0.36</td>
<td>18.33</td>
<td>796.6</td>
</tr>
<tr>
<td>C3</td>
<td>118.26</td>
<td>0</td>
<td>0.13</td>
<td>18.38</td>
<td>850.1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Test</th>
<th>Average Velocity (m/s)</th>
<th>Maximum Velocity (m/s)</th>
<th>Average VEL (%)</th>
<th>Average RVEL (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>0.331</td>
<td>1.013</td>
<td>30.8</td>
<td>38.7</td>
</tr>
<tr>
<td>S2</td>
<td>0.366</td>
<td>1.178</td>
<td>36.5</td>
<td>46.9</td>
</tr>
<tr>
<td>S3</td>
<td>0.460</td>
<td>1.101</td>
<td>41.1</td>
<td>34.2</td>
</tr>
<tr>
<td>V1</td>
<td>0.144</td>
<td>0.948</td>
<td>25.0</td>
<td>40.1</td>
</tr>
<tr>
<td>V2</td>
<td>0.238</td>
<td>0.797</td>
<td>26.8</td>
<td>24.4</td>
</tr>
<tr>
<td>V3</td>
<td>0.193</td>
<td>1.078</td>
<td>26.8</td>
<td>28.2</td>
</tr>
<tr>
<td>C1</td>
<td>0.172</td>
<td>-0.487</td>
<td>17.0</td>
<td>21.9</td>
</tr>
<tr>
<td>C2</td>
<td>0.160</td>
<td>0.544</td>
<td>16.0</td>
<td>24.0</td>
</tr>
<tr>
<td>C3</td>
<td>0.149</td>
<td>0.558</td>
<td>14.6</td>
<td>21.2</td>
</tr>
</tbody>
</table>

Table 4.3: Subject 3 Test Results
<table>
<thead>
<tr>
<th>Test</th>
<th>Time (s)</th>
<th>Collisions</th>
<th>Distance from Goal (m)</th>
<th>Total Distance (m)</th>
<th>Total Rotation (°)</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>21.38</td>
<td>1</td>
<td>0.30</td>
<td>18.17</td>
<td>603.6</td>
</tr>
<tr>
<td>S2</td>
<td>30.16</td>
<td>1</td>
<td>0.25</td>
<td>18.56</td>
<td>724.6</td>
</tr>
<tr>
<td>S3</td>
<td>26.14</td>
<td>1</td>
<td>0.23</td>
<td>18.34</td>
<td>638.8</td>
</tr>
<tr>
<td>V1</td>
<td>59.65</td>
<td>2</td>
<td>0.08</td>
<td>23.11</td>
<td>964.7</td>
</tr>
<tr>
<td>V2</td>
<td>54.21</td>
<td>2</td>
<td>0.03</td>
<td>22.21</td>
<td>871.5</td>
</tr>
<tr>
<td>V3</td>
<td>53.57</td>
<td>1</td>
<td>0.10</td>
<td>20.65</td>
<td>1163.7</td>
</tr>
<tr>
<td>C1</td>
<td>66.65</td>
<td>0</td>
<td>0.20</td>
<td>18.97</td>
<td>773.4</td>
</tr>
<tr>
<td>C2</td>
<td>52.11</td>
<td>0</td>
<td>0.23</td>
<td>18.50</td>
<td>778.4</td>
</tr>
<tr>
<td>C3</td>
<td>60.85</td>
<td>0</td>
<td>0.20</td>
<td>18.13</td>
<td>909.5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Test</th>
<th>Average Velocity (m/s)</th>
<th>Maximum Velocity (m/s)</th>
<th>Average VEL (%)</th>
<th>Average RVEL (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>0.828</td>
<td>1.371</td>
<td>83.0</td>
<td>77.6</td>
</tr>
<tr>
<td>S2</td>
<td>0.578</td>
<td>1.394</td>
<td>73.7</td>
<td>70.3</td>
</tr>
<tr>
<td>S3</td>
<td>0.661</td>
<td>1.400</td>
<td>84.7</td>
<td>75.7</td>
</tr>
<tr>
<td>V1</td>
<td>0.319</td>
<td>1.314</td>
<td>61.1</td>
<td>70.0</td>
</tr>
<tr>
<td>V2</td>
<td>0.370</td>
<td>1.330</td>
<td>65.3</td>
<td>65.8</td>
</tr>
<tr>
<td>V3</td>
<td>0.350</td>
<td>1.311</td>
<td>65.9</td>
<td>72.9</td>
</tr>
<tr>
<td>C1</td>
<td>0.276</td>
<td>1.423</td>
<td>35.3</td>
<td>47.5</td>
</tr>
<tr>
<td>C2</td>
<td>0.343</td>
<td>1.330</td>
<td>36.5</td>
<td>43.4</td>
</tr>
<tr>
<td>C3</td>
<td>0.287</td>
<td>1.294</td>
<td>34.0</td>
<td>45.5</td>
</tr>
</tbody>
</table>

Table 4.4: Subject 4 Test Results
<table>
<thead>
<tr>
<th>Test</th>
<th>Time (s)</th>
<th>Collisions</th>
<th>Distance from Goal (m)</th>
<th>Total Distance (m)</th>
<th>Total Rotation (°)</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>38.50</td>
<td>0</td>
<td>0.15</td>
<td>18.86</td>
<td>634.2</td>
</tr>
<tr>
<td>S2</td>
<td>38.86</td>
<td>1</td>
<td>0.20</td>
<td>18.20</td>
<td>761.5</td>
</tr>
<tr>
<td>S3</td>
<td>31.08</td>
<td>1</td>
<td>0.36</td>
<td>17.60</td>
<td>565.7</td>
</tr>
<tr>
<td>V1</td>
<td>115.22</td>
<td>0</td>
<td>0.05</td>
<td>18.35</td>
<td>1368.6</td>
</tr>
<tr>
<td>V2</td>
<td>129.31</td>
<td>0</td>
<td>0.05</td>
<td>18.44</td>
<td>1277.9</td>
</tr>
<tr>
<td>V3</td>
<td>127.52</td>
<td>1</td>
<td>0.05</td>
<td>19.42</td>
<td>1639.7</td>
</tr>
<tr>
<td>C1</td>
<td>88.74</td>
<td>0</td>
<td>0.03</td>
<td>18.76</td>
<td>1024.8</td>
</tr>
<tr>
<td>C2</td>
<td>101.67</td>
<td>1</td>
<td>0.05</td>
<td>20.36</td>
<td>774.9</td>
</tr>
<tr>
<td>C3</td>
<td>94.84</td>
<td>0</td>
<td>0.00</td>
<td>18.16</td>
<td>909.9</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Test</th>
<th>Average Velocity (m/s)</th>
<th>Maximum Velocity (m/s)</th>
<th>Average VEL (%)</th>
<th>Average RVEL (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>0.472</td>
<td>1.385</td>
<td>48.6</td>
<td>52.9</td>
</tr>
<tr>
<td>S2</td>
<td>0.441</td>
<td>1.237</td>
<td>44.8</td>
<td>61.8</td>
</tr>
<tr>
<td>S3</td>
<td>0.546</td>
<td>1.200</td>
<td>48.6</td>
<td>52.6</td>
</tr>
<tr>
<td>V1</td>
<td>0.151</td>
<td>0.875</td>
<td>25.5</td>
<td>50.6</td>
</tr>
<tr>
<td>V2</td>
<td>0.135</td>
<td>0.987</td>
<td>28.8</td>
<td>52.9</td>
</tr>
<tr>
<td>V3</td>
<td>0.145</td>
<td>0.937</td>
<td>27.6</td>
<td>54.0</td>
</tr>
<tr>
<td>C1</td>
<td>0.204</td>
<td>0.685</td>
<td>25.6</td>
<td>35.5</td>
</tr>
<tr>
<td>C2</td>
<td>0.178</td>
<td>0.732</td>
<td>21.9</td>
<td>30.3</td>
</tr>
<tr>
<td>C3</td>
<td>0.185</td>
<td>0.824</td>
<td>22.1</td>
<td>36.0</td>
</tr>
</tbody>
</table>

Table 4.5: Subject 5 Test Results
<table>
<thead>
<tr>
<th>Test</th>
<th>Time (s)</th>
<th>Collisions</th>
<th>Distance from Goal (m)</th>
<th>Total Distance (m)</th>
<th>Total Rotation (°)</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>26.31</td>
<td>1</td>
<td>0.23</td>
<td>18.11</td>
<td>515.0</td>
</tr>
<tr>
<td>S2</td>
<td>26.74</td>
<td>2</td>
<td>0.33</td>
<td>17.03</td>
<td>487.3</td>
</tr>
<tr>
<td>S3</td>
<td>26.16</td>
<td>1</td>
<td>0.13</td>
<td>18.52</td>
<td>505.5</td>
</tr>
<tr>
<td>V1</td>
<td>74.25</td>
<td>0</td>
<td>0.08</td>
<td>17.57</td>
<td>606.1</td>
</tr>
<tr>
<td>V2</td>
<td>54.12</td>
<td>1</td>
<td>0.10</td>
<td>19.26</td>
<td>592.7</td>
</tr>
<tr>
<td>V3</td>
<td>47.24</td>
<td>0</td>
<td>0.15</td>
<td>17.77</td>
<td>600.8</td>
</tr>
<tr>
<td>C1</td>
<td>66.00</td>
<td>0</td>
<td>0.25</td>
<td>18.07</td>
<td>843.4</td>
</tr>
<tr>
<td>C2</td>
<td>69.50</td>
<td>0</td>
<td>0.13</td>
<td>18.58</td>
<td>720.4</td>
</tr>
<tr>
<td>C3</td>
<td>60.78</td>
<td>0</td>
<td>0.05</td>
<td>18.12</td>
<td>845.2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Test</th>
<th>Average Velocity (m/s)</th>
<th>Maximum Velocity (m/s)</th>
<th>Average VEL (%)</th>
<th>Average RVEL (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>0.670</td>
<td>1.385</td>
<td>59.5</td>
<td>54.6</td>
</tr>
<tr>
<td>S2</td>
<td>0.617</td>
<td>1.278</td>
<td>57.6</td>
<td>45.5</td>
</tr>
<tr>
<td>S3</td>
<td>0.674</td>
<td>1.138</td>
<td>57.8</td>
<td>51.5</td>
</tr>
<tr>
<td>V1</td>
<td>0.230</td>
<td>0.632</td>
<td>20.9</td>
<td>32.1</td>
</tr>
<tr>
<td>V2</td>
<td>0.347</td>
<td>0.963</td>
<td>32.8</td>
<td>31.4</td>
</tr>
<tr>
<td>V3</td>
<td>0.367</td>
<td>1.273</td>
<td>31.7</td>
<td>34.6</td>
</tr>
<tr>
<td>C1</td>
<td>0.267</td>
<td>0.806</td>
<td>26.0</td>
<td>33.3</td>
</tr>
<tr>
<td>C2</td>
<td>0.259</td>
<td>0.729</td>
<td>26.1</td>
<td>31.6</td>
</tr>
<tr>
<td>C3</td>
<td>0.290</td>
<td>0.993</td>
<td>27.6</td>
<td>37.8</td>
</tr>
</tbody>
</table>

Table 4.6: Subject 6 Test Results
<table>
<thead>
<tr>
<th>Test</th>
<th>Time (s)</th>
<th>Collisions</th>
<th>Distance from Goal (m)</th>
<th>Total Distance (m)</th>
<th>Total Rotation (°)</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>19.92</td>
<td>1.67</td>
<td>0.37</td>
<td>17.86</td>
<td>656.5</td>
</tr>
<tr>
<td>V1</td>
<td>54.40</td>
<td>2.00</td>
<td>0.43</td>
<td>20.60</td>
<td>1080.1</td>
</tr>
<tr>
<td>C1</td>
<td>37.51</td>
<td>0.67</td>
<td>0.32</td>
<td>19.26</td>
<td>720.1</td>
</tr>
<tr>
<td>S2</td>
<td>21.16</td>
<td>1.33</td>
<td>0.19</td>
<td>18.52</td>
<td>640.4</td>
</tr>
<tr>
<td>V2</td>
<td>31.99</td>
<td>1.33</td>
<td>0.34</td>
<td>20.55</td>
<td>660.4</td>
</tr>
<tr>
<td>C2</td>
<td>43.81</td>
<td>1.00</td>
<td>0.21</td>
<td>20.29</td>
<td>772.4</td>
</tr>
<tr>
<td>S3</td>
<td>46.54</td>
<td>1.00</td>
<td>0.41</td>
<td>18.23</td>
<td>885.5</td>
</tr>
<tr>
<td>V3</td>
<td>101.56</td>
<td>2.67</td>
<td>0.08</td>
<td>23.92</td>
<td>991.6</td>
</tr>
<tr>
<td>C3</td>
<td>108.28</td>
<td>0.33</td>
<td>0.29</td>
<td>18.17</td>
<td>754.0</td>
</tr>
<tr>
<td>S4</td>
<td>25.89</td>
<td>1.00</td>
<td>0.36</td>
<td>18.36</td>
<td>655.7</td>
</tr>
<tr>
<td>V4</td>
<td>55.81</td>
<td>1.67</td>
<td>0.07</td>
<td>21.99</td>
<td>1000.0</td>
</tr>
<tr>
<td>C4</td>
<td>59.87</td>
<td>0.00</td>
<td>0.21</td>
<td>18.53</td>
<td>820.4</td>
</tr>
<tr>
<td>S5</td>
<td>36.15</td>
<td>0.67</td>
<td>0.24</td>
<td>18.23</td>
<td>653.8</td>
</tr>
<tr>
<td>V5</td>
<td>124.02</td>
<td>0.33</td>
<td>0.05</td>
<td>18.74</td>
<td>1428.8</td>
</tr>
<tr>
<td>C5</td>
<td>95.09</td>
<td>0.33</td>
<td>0.03</td>
<td>19.09</td>
<td>903.2</td>
</tr>
<tr>
<td>S6</td>
<td>26.40</td>
<td>1.33</td>
<td>0.23</td>
<td>17.88</td>
<td>502.6</td>
</tr>
<tr>
<td>V6</td>
<td>58.54</td>
<td>0.33</td>
<td>0.11</td>
<td>18.20</td>
<td>599.9</td>
</tr>
<tr>
<td>C6</td>
<td>65.43</td>
<td>0.00</td>
<td>0.14</td>
<td>18.26</td>
<td>803.0</td>
</tr>
</tbody>
</table>

Table 4.7: Test Results for All Subjects Grouped by Subject, Part A
<table>
<thead>
<tr>
<th>Test</th>
<th>Average Velocity (m/s)</th>
<th>Maximum Velocity (m/s)</th>
<th>Average VEL (%)</th>
<th>Average RVEL (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>0.843</td>
<td>1.379</td>
<td>88.3</td>
<td>82.0</td>
</tr>
<tr>
<td>V1</td>
<td>0.323</td>
<td>1.424</td>
<td>64.3</td>
<td>79.4</td>
</tr>
<tr>
<td>C1</td>
<td>0.493</td>
<td>1.420</td>
<td>73.3</td>
<td>76.8</td>
</tr>
<tr>
<td>S2</td>
<td>0.843</td>
<td>1.410</td>
<td>86.7</td>
<td>70.9</td>
</tr>
<tr>
<td>V2</td>
<td>0.579</td>
<td>1.365</td>
<td>57.8</td>
<td>52.2</td>
</tr>
<tr>
<td>C2</td>
<td>0.443</td>
<td>1.251</td>
<td>43.2</td>
<td>44.6</td>
</tr>
<tr>
<td>S3</td>
<td>0.386</td>
<td>1.097</td>
<td>36.2</td>
<td>39.9</td>
</tr>
<tr>
<td>V3</td>
<td>0.192</td>
<td>0.941</td>
<td>26.2</td>
<td>30.9</td>
</tr>
<tr>
<td>C3</td>
<td>0.161</td>
<td>0.530</td>
<td>15.9</td>
<td>22.4</td>
</tr>
<tr>
<td>S4</td>
<td>0.689</td>
<td>1.388</td>
<td>80.5</td>
<td>74.6</td>
</tr>
<tr>
<td>V4</td>
<td>0.347</td>
<td>1.318</td>
<td>64.1</td>
<td>69.6</td>
</tr>
<tr>
<td>C4</td>
<td>0.302</td>
<td>1.349</td>
<td>35.3</td>
<td>45.5</td>
</tr>
<tr>
<td>S5</td>
<td>0.486</td>
<td>1.274</td>
<td>47.4</td>
<td>55.7</td>
</tr>
<tr>
<td>V5</td>
<td>0.144</td>
<td>0.933</td>
<td>27.3</td>
<td>52.5</td>
</tr>
<tr>
<td>C5</td>
<td>0.189</td>
<td>0.747</td>
<td>23.2</td>
<td>33.9</td>
</tr>
<tr>
<td>S6</td>
<td>0.654</td>
<td>1.267</td>
<td>58.3</td>
<td>50.5</td>
</tr>
<tr>
<td>V6</td>
<td>0.315</td>
<td>0.956</td>
<td>28.5</td>
<td>32.7</td>
</tr>
<tr>
<td>C6</td>
<td>0.272</td>
<td>0.843</td>
<td>26.6</td>
<td>34.2</td>
</tr>
</tbody>
</table>

Table 4.8: Test Results for All Subjects Grouped by Subject, Part B
<table>
<thead>
<tr>
<th>Test</th>
<th>Time (s)</th>
<th>Collisions</th>
<th>Distance from Goal (m)</th>
<th>Total Distance (m)</th>
<th>Total Rotation (°)</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>19.92</td>
<td>1.67</td>
<td>0.37</td>
<td>17.86</td>
<td>656.5</td>
</tr>
<tr>
<td>S2</td>
<td>21.16</td>
<td>1.33</td>
<td>0.19</td>
<td>18.52</td>
<td>640.4</td>
</tr>
<tr>
<td>S3</td>
<td>46.54</td>
<td>1.00</td>
<td>0.41</td>
<td>18.23</td>
<td>885.5</td>
</tr>
<tr>
<td>S4</td>
<td>25.89</td>
<td>1.00</td>
<td>0.36</td>
<td>18.36</td>
<td>655.7</td>
</tr>
<tr>
<td>S5</td>
<td>36.15</td>
<td>0.67</td>
<td>0.24</td>
<td>18.23</td>
<td>653.8</td>
</tr>
<tr>
<td>S6</td>
<td>26.40</td>
<td>1.33</td>
<td>0.23</td>
<td>17.88</td>
<td>502.6</td>
</tr>
<tr>
<td>V1</td>
<td>54.40</td>
<td>2.00</td>
<td>0.43</td>
<td>20.60</td>
<td>1080.1</td>
</tr>
<tr>
<td>V2</td>
<td>31.99</td>
<td>1.33</td>
<td>0.34</td>
<td>20.55</td>
<td>660.4</td>
</tr>
<tr>
<td>V3</td>
<td>101.56</td>
<td>2.67</td>
<td>0.08</td>
<td>23.92</td>
<td>991.6</td>
</tr>
<tr>
<td>V4</td>
<td>55.81</td>
<td>1.67</td>
<td>0.07</td>
<td>21.99</td>
<td>1000.0</td>
</tr>
<tr>
<td>V5</td>
<td>124.02</td>
<td>0.33</td>
<td>0.05</td>
<td>18.74</td>
<td>1428.8</td>
</tr>
<tr>
<td>V6</td>
<td>58.54</td>
<td>0.33</td>
<td>0.11</td>
<td>18.20</td>
<td>599.9</td>
</tr>
<tr>
<td>C1</td>
<td>37.51</td>
<td>0.67</td>
<td>0.32</td>
<td>19.26</td>
<td>720.1</td>
</tr>
<tr>
<td>C2</td>
<td>43.81</td>
<td>1.00</td>
<td>0.21</td>
<td>20.29</td>
<td>772.4</td>
</tr>
<tr>
<td>C3</td>
<td>108.28</td>
<td>0.33</td>
<td>0.29</td>
<td>18.17</td>
<td>754.0</td>
</tr>
<tr>
<td>C4</td>
<td>59.87</td>
<td>0.00</td>
<td>0.21</td>
<td>18.53</td>
<td>820.4</td>
</tr>
<tr>
<td>C5</td>
<td>95.09</td>
<td>0.33</td>
<td>0.03</td>
<td>19.09</td>
<td>903.2</td>
</tr>
<tr>
<td>C6</td>
<td>65.43</td>
<td>0.00</td>
<td>0.14</td>
<td>18.26</td>
<td>803.0</td>
</tr>
</tbody>
</table>

Table 4.9: Test Results for All Subjects Grouped by Test, Part A
<table>
<thead>
<tr>
<th>Test</th>
<th>Average Velocity (m/s)</th>
<th>Maximum Velocity (m/s)</th>
<th>Average VEL (%)</th>
<th>Average RVEL (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>0.843</td>
<td>1.379</td>
<td>88.3</td>
<td>82.0</td>
</tr>
<tr>
<td>S2</td>
<td>0.843</td>
<td>1.410</td>
<td>86.7</td>
<td>70.9</td>
</tr>
<tr>
<td>S3</td>
<td>0.386</td>
<td>1.097</td>
<td>36.2</td>
<td>39.9</td>
</tr>
<tr>
<td>S4</td>
<td>0.689</td>
<td>1.388</td>
<td>80.5</td>
<td>74.6</td>
</tr>
<tr>
<td>S5</td>
<td>0.486</td>
<td>1.274</td>
<td>47.4</td>
<td>55.7</td>
</tr>
<tr>
<td>S6</td>
<td>0.654</td>
<td>1.267</td>
<td>58.3</td>
<td>50.5</td>
</tr>
<tr>
<td>V1</td>
<td>0.323</td>
<td>1.424</td>
<td>64.3</td>
<td>79.4</td>
</tr>
<tr>
<td>V2</td>
<td>0.579</td>
<td>1.365</td>
<td>57.8</td>
<td>52.2</td>
</tr>
<tr>
<td>V3</td>
<td>0.192</td>
<td>0.941</td>
<td>26.2</td>
<td>30.9</td>
</tr>
<tr>
<td>V4</td>
<td>0.347</td>
<td>1.318</td>
<td>64.1</td>
<td>69.6</td>
</tr>
<tr>
<td>V5</td>
<td>0.144</td>
<td>0.933</td>
<td>27.3</td>
<td>52.5</td>
</tr>
<tr>
<td>V6</td>
<td>0.315</td>
<td>0.956</td>
<td>28.5</td>
<td>32.7</td>
</tr>
<tr>
<td>C1</td>
<td>0.493</td>
<td>1.420</td>
<td>73.3</td>
<td>76.8</td>
</tr>
<tr>
<td>C2</td>
<td>0.443</td>
<td>1.251</td>
<td>43.2</td>
<td>44.6</td>
</tr>
<tr>
<td>C3</td>
<td>0.161</td>
<td>0.530</td>
<td>15.9</td>
<td>22.4</td>
</tr>
<tr>
<td>C4</td>
<td>0.302</td>
<td>1.349</td>
<td>35.3</td>
<td>45.5</td>
</tr>
<tr>
<td>C5</td>
<td>0.189</td>
<td>0.747</td>
<td>23.2</td>
<td>33.9</td>
</tr>
<tr>
<td>C6</td>
<td>0.272</td>
<td>0.843</td>
<td>26.6</td>
<td>34.2</td>
</tr>
</tbody>
</table>

Table 4.10: Test Results for All Subjects Grouped by Test, Part B
4.4 Testing Analysis

When the control station is logging data, it is recording the robot state and the control inputs at a rate of about 8.4 Hz. With more than 50 tests performed, this produces a monumental amount of data. Choosing which elements of the data to analyze is dependent on what one wants to prove or refute. Since the intent of the HVTR system is to use high speed travel to shorten mission times, the logical beginning for analysis will be how long the operator took to maneuver from the starting location to the goal location.

4.4.1 Time

The control method that allowed all test subjects to finish most quickly was the one using only their sight. In most cases, the operators were more than twice as fast at reaching the goal when using this control method as compared to using video or the computer. This is not surprising because this method provides the operators with the best understanding of the relationship between their control inputs and the resultant actions of the microrover.

For four of the six test subjects, the video interface was their next fastest, leaving two test subjects for which the computer interface was their next fastest. For most operators, these two control methods had very similar times. In fact, for all operators, the difference in time between the sight test and their next fastest is less than the time difference between the video interface and the computer interface.

Based on these tests, neither the video interface nor the computer interface show much superiority over the other. The video interface appears marginally faster, but the computer interface better allowed operators to avoid obstacles. On average, the number of collisions that occurred when each operator used the computer interface was less than or equal to the number of collisions that the operator had when using the video interface.

4.4.2 Driving Technique

Since two subjects performed better with the computer interface than with the video interface, it would be interesting to see if there is some similarity between their driving techniques. If there is a similarity, it would then be useful to see if their technique differs from those subjects who performed better on the video test.

To compare the driving techniques of Subject 1 and Subject 3 (the two subjects who performed better on the computer tests than on the video tests), the control in-
puts for velocity during the first twenty seconds of a test were analyzed. Surprisingly, Subject 1 and Subject 3 had the most different driving styles of any two operators. As can be seen in Figure 4-3, Subject 1 was very jerky with his control inputs, continuously jumping from full throttle to no throttle. Figure 4-4 shows Subject 3’s vastly different control style. She preferred to drive the robot slower with smoother control inputs. Because Subject 1 and Subject 3 differed so greatly in their techniques, it is impossible to say that a jerky or smooth control style influences if a person is better with the computer interface than with the video interface.

![Velocity Input for Subject 1, Test C2](image)

Figure 4-3: Velocity Inputs of Subject 1

### 4.4.3 Summary

In summary, the tests that were performed on the HVTR system showed that the computer interface had little or no advantage over the video system in the small, obstacle-rich environment used for the tests. Directly controlling the robot by sight greatly outperformed both of the other techniques.
Figure 4-4: Velocity Inputs of Subject 3
Chapter 5

Conclusion

This chapter concludes the study of the HVTR project. It contains

- a summary of what this thesis has covered,
- a discussion of what can be done to further develop the HVTR system, and
- some conclusions regarding the objectives of the HVTR project.

5.1 Thesis Summary

Unlike most microrover interfaces, the central theme of the HVTR project was to
develop a system for controlling a microrover at high speeds. Unique difficulties were
faced in implementing and testing this design.

Chapter 2 explored what tools could be used to achieve the goal of a high speed
microrover system. Several microrover control methods were analyzed for their viability
at high speed, and a teleoperated system using external reconnaissance imagery
was chosen as the basis of the HVTR design. Design issues were then explored using
a theory of supervisory control and some guidelines for building human control
interfaces.

An in-depth description of the HVTR implementation was given in Chapter 3.
All hardware and software elements were described in detail. The evolution of the
computer interface was also provided.

Chapter 4 described the tests performed on the HVTR system to assess its capa-
bilities in comparison to two common control techniques. Several tables detail the
results of these tests.
5.2 Future Possibilities

There is still a lot of development that needs to be done on the HVTR project. To improve the position recalibration system, the video transmitter needs to be upgraded and the distortion of the camera lens needs to be characterized. More positioning sensors should be added and the Kalman filter should be further developed to improve the synchronization between the onscreen robot icon and the microrover’s true position.

If the video system and positioning system are improved, a better suite of tests can be done to analyze the performance of the project. Once the system is working at its full potential with the Pioneer AT platform, it will be time to upgrade to a faster platform. In theory, the system should scale well to greater speeds, but the theory can only be proved with a working high speed robot.

Many interesting modifications can be made to the control interface as the HVTR project proceeds. For the initial prototype, no autonomy was used because the performance of autonomy is often speed dependent, but some simple autonomy could probably be integrated into the control interface without sacrificing high speed performance. A user supervised waypoint navigation system would be the first choice for this development. The operator would input multiple waypoints defining a complete path that the microrover should traverse from its starting location to its goal location. The operator would then actively control the robot’s velocity while steering inputs would be controlled by the operator station. The operator would still be required to watch the navigation window and video window for unforeseen obstacles and discrepancies. The advantage is that the operator would have to perform less micro-management of the robot’s heading. If at any time the operator notices that the robot’s programmed path would cause it to collide with something, the operator could quickly modify the planned path by manipulating the position of the waypoints.

5.3 Conclusion

The HVTR system is meant for navigating large obstacle-free areas, but the tests detailed in Chapter 4 were analyzing performance in a small obstacle-rich environment. Because it was not possible to construct experiments that accurately reflected the mission that the robot was designed for, it is difficult to draw conclusions based on the test results.

The HVTR system is designed to be used in situations where the operator cannot see the microrover, so the HVTR systems’ poor performance in comparison to
the sight-based test should not be too much of a concern. What is of importance is the similarities in performance between using the HVTR system and controlling the microrover by only video. If the HVTR system cannot outperform the control technique which solely uses video, then there is no point in using the more complex HVTR system.

The HVTR system does have two distinct advantages over the video-only approach that could not be explored in the tests run to date.

1. The video image will be much less stable when the microrover is traveling quickly on terrain that is not as smooth as the carpet floor used in the recent tests. This will affect the video-only control interface more than the HVTR system.

2. Integrated into the HVTR system are all of the elements necessary to successfully complete a mission in complex environments such as cities. The operator can begin by planning a route for the microrover to travel by using the onscreen reconnaissance imagery. Throughout the mission, the progress of the microrover can be noted onscreen. In an environment such as a city, a video only interface does not provide any planning abilities, nor does it give any indication of the robot’s location in relation to its goal.

In summary, the HVTR project shows great promise, but its implementation has yet to be fully realized. The fusion of downward looking imagery and live video is a unique method for controlling microrovers that offers many advantages over traditional techniques.
Bibliography


Appendix A

Unmanned Vehicle Laboratory

The Unmanned Vehicle Laboratory (UVL) was established in 1990 as the Planetary Rover Baseline Experiment (PROBE) laboratory [15]. Students from MIT, Tufts, Boston University, and Northeastern University have worked in the UVL in cooperation with employees from the Charles Stark Draper Laboratory on a variety of robotic platforms.

A.1 The MITy series

The mission of the MITy project was to build several microrovers exploring the design and implementation of robots for extraterrestrial environments. They are all six-wheeled flexible frame designs, capable of high maneuverability.

A.1.1 MITy-1

The goal of MITy-1, the initial prototype, was to prove that it was possible to build a microrover capable of limited autonomy. Its functionality was limited, but it was very successful [5].

A.1.2 MITy-2

MITy-2 built upon the success of MITy-1 by improving the autonomous capabilities of the onboard software, and exploring a wider variety of sensors. For instance, a sun sensor was used as a positioning device because it would be an accurate method of heading determination on the moon or Mars [8].

A.1.3 MITy-3

MITy-3 was the last in the series of MITy robots. It experimented with an alter-
native drive train designed to prevent the microrover from getting stuck when driving over obstacles.

A.2 The EOD series

Building on the experience learned from the MITy series of autonomous robots, the EOD series focused on an explosive ordinance disposal mission for microrovers. The six-wheeled, flexible-frame platform is similar to the platform used in the MITy robots, except with stronger drive motors and a more complex Ackerman steering system [15]. EOD-1 was teleoperated, and EOD-2 was autonomous. More information on the EOD-2 system can be found in Appendix B.

A.3 Companion

Companion was a large mobile robot built on a motorized wheelchair platform. It had multiple processors, twenty-four sonars, and a laser range-finder that gave it the ability to maneuver autonomously through many situations. The goal of the Companion project was to build a robot that could coordinate the control of multiple autonomous vehicles, thereby allowing groups of robots to be controlled by a single human operator [3].

A.4 DSAAV

The Draper Small Autonomous Aerial Vehicle (DSAAV) is an autonomous helicopter platform designed as a stable reconnaissance platform. It can take off, hover, navigate, and land without human intervention. Its navigation system is built around a highly accurate differential GPS system that provides better than one meter accuracy in absolute position [12]. In a recent test, two DSAAVs were operated beyond the line-of-sight of the operator, each performing a different mission.

A.5 WASP

The Wide-Area Surveillance Projectile (WASP) is a reconnaissance aircraft that is launched within a 5-inch artillery shell. Once the shell is 11 nautical miles downrange of the cannon that launched it, a parachute deploys to decelerate the WASP to about 300 ft/s. Once it reaches this speed, it deploys its wings, tail fins, and a propeller. A small engine then starts to provide about 10 minutes of powered flight
followed by 5 minutes of a gliding descent. While in flight, the WASP provides a camera platform for reconnaissance as well as GPS coordinates [9].

A.6 VCUUV

The Vorticity Control Unmanned Underwater Vehicle project was built to study the efficiencies of fish-like propulsion. It mimics the swimming motion of an 8 ft bluefin tuna. Tests to date show that the VCUUV exhibits better maneuverability and efficiency than traditional propeller-based underwater vehicles [4].
Appendix B

EOD-2

Much of this appendix is excerpted from Micro-Rover Operator Station [11], a paper which overviewed the EOD-2 project and first introduced the HVTR concept.

B.1 EOD-2 Robot

The EOD-2 vehicle is designed to be an autonomous explosive ordinance disposal robot. To command the vehicle, the operator assigns the microrover abstract tasks such as search an area or navigate to a specific destination. The microrover is expected to execute these tasks or signal the operator that they cannot be performed. A group of three sonars allows EOD-2 to detect the location of nearby obstacles, which are saved for later path calculations.

B.2 EOD-2 Control Station

The EOD-2 control station is written in C, and makes use of the X and Motif libraries for its graphics. It runs on a Pentium desktop PC using Linux for its operating system.

B.2.1 Interface

The main interface to EOD-2 is the mission control window. This window displays a two-dimensional model of the microrover's surroundings, the current task stack, and the current position, heading, and velocity of the vehicle. The two-dimensional model has multiple overlays that can be hidden if necessary to reduce screen clutter. These overlays include:

- Obstacles
• Sonar Hits
• Path History
• Planned Tasks
• Map Grid

Tasks can be pushed onto or popped from the task stack from the mission control window. When a task is being added to the stack, a task-creation window opens and displays the task parameters. If the planned tasks overlay is visible, the mission control window will display the proposed task as the parameters are changed. The available tasks include:

• **Segment Follow**—Attempt to follow a straight-line path between two points as closely as possible.

• **Waypoint Follow**—Execute multiple connected segments in succession.

• **Transit**—Plan a series of waypoints to a desired location while avoiding known obstacles.

• **Area Search**—Traverse an area using a search pattern and attempt to detect unexploded ordinance.

• **Simple Control**—Allow the operator to take direct teleoperated control of the vehicle.

### B.2.2 Path Planning

Using the control station, the operator can submit a transit task to the micro rover which commands the robot to travel to an approximate location. To determine how to reach this location, the control station uses an \( A^* \) search algorithm to plan the most efficient path to that position given the control station’s current knowledge of obstacle locations.

### B.2.3 Navigation

Navigation to an operator declared destination is accomplished using a dead reckoning scheme involving the motor encoders for the drive and steering motors in combination with a rate gyro.
B.2.4 Map Building

During transit, the control station maintains the updated coordinates of the microrover as well as the locations of known and newly discovered obstacles. This information is continuously updated and allows for the real-time construction of a map of the mission area.

B.2.5 Multiple Robot Simulation

The EOD-2 control station has the ability to control multiple microrovers. Individual sensor data from each microrover is assimilated into the global map of the whole microrover community. When researching cooperation between multiple robots, the EOD-2 has the additional feature of allowing one or more physical microrovers to interact with a number of simulated microrovers. Each simulated microrover is programmed to behave like a real microrover navigating in a true environment.