Design of Control Algorithms for Automation of a
Full Dimension Continuous Haulage System

Aishwarya Varadhan

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Industrial and Systems Engineering

Dr. Michael P. Deisenroth, Chair
Dr. Robert H. Sturges Jr., Co-Chair
Dr. Charles F. Reinholtz

Volume I

April 18, 2000
Blacksburg, Virginia

Keywords: Automation algorithms, Mobile robotics, Sensor based navigation

Copyright 2000, Aishwarya Varadhan
Design of Control Algorithms for Automation of a Full Dimension Continuous Haulage System

Aishwarya Varadhan

(ABSTRACT)

The main theme of this research will be to develop solutions to the widely known 3-part question in mobile robotics comprising of “Where am I?” “Where should I be?” and “How do I get there”. This can be achieved by implementing automation algorithms. Automation algorithms or control algorithms are vital components of any autonomous vehicle. Design and development of both prototype and full-scale control algorithms for a Long-Airdox Full Dimension Continuous Haulage system will be the main focus. Automation is a highly complex task, which aims at achieving increased levels of equipment efficiency by eliminating errors that arise due to human interference. Achieving a fully autonomous operation of a machine involves a variety of high-level interlaced functions that work in harmony, and at the same time perform functions that mimic the human operator. Automation has expanded widely in the field of mobile robotics, thus leading to the development of autonomous robots, automated guided vehicles and other autonomous vehicles. An indispensable element of an autonomous vehicle is a navigation system that steers it to a required destination. The vehicle must be able to determine its relationship to the environment by sensing, and also must be able to decide what actions are required to achieve its goal(s) in the working environment. The goal of this research is to demonstrate a fully autonomous operation of the Continuous Haulage System, and to establish its potential advantages.
ACKNOWLEDGEMENTS

I would like to thank Dr. Deisenroth and Dr. Bob Sturges for being wonderful advisors, and for having provided me the opportunity to work with them. It’s always been a great pleasure discussing with Dr. Deisenroth, both my research and career goals. I am indebted to him for all his suggestions and help in guiding me to choose an appropriate advanced career. I owe my greatest thanks to him! And for all the valuable guidance that he has provided.

Every graduate student’s dream professor – this is what Dr. Bob is!. I would like to thank him for all his enthusiastic guidance. He has always been a great leader for this research effort, and I will always consider him as ‘the’ example for solving ‘any’ problem, and for being so modest and humble. His cheerful nature will linger forever, and he will always be my inspiration!

I would like to thank Dr. Reinholtz for being a valuable guide, and for serving on my thesis committee. Thanks, to the dynamic team at Long-Airdox, for they have always been great to work with. Special thanks to Mick, Dave, Chitra, John and Breno. I would also like to thank all the faculty members in ISE and ME for their academic encouragement.

A very special thanks to John (Dr. Tester), for being the most accessible, friendly, and valuable guide. Thanks to him for his timely guidance, and help in both academic and personal work. I would like to thank my teammates on this research for they have always put in their best, and those long nights would not have been possible without their jokes, and cheese crackers!. Greatest thanks to Bruce (the Master!), Mikey (Mike), Yos (Yodyot), Lu, Pavan, Pramod and Collin. Best wishes to all of them for all their future endeavors.

A special thanks to all my friends, to some of them who have encouraged me to stuff cheesy pizzas in (Sumo, Anush, PJ, PL, Mohan, Kalyan), and to all of them for just being good to me. Best wishes are provided as they progress towards completion, and a great career! The guidance of Lovedia Cole is greatly appreciated, and special thanks to Kathy.

And finally, I would like to thank all my family members. Countless, have their efforts been in encouraging me to pursue a graduate degree. Thanks to Satya, Aditya and Rajeev for being the best brothers. Greatest thanks to Krishnamoorthy Uncle and Aunt for being the best Uncle and Aunt in the world! I dedicate this work to them, and they will always continue to be the best!
## CONTENTS

**Table of contents**

<table>
<thead>
<tr>
<th>Chapter 1: Introduction</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1 Underground coal mining equipment</td>
<td>1</td>
</tr>
<tr>
<td>1.2 The LA CHS system</td>
<td>1</td>
</tr>
<tr>
<td>1.3 Uniqueness of the LA system</td>
<td>4</td>
</tr>
<tr>
<td>1.4 Problem statement</td>
<td>5</td>
</tr>
<tr>
<td>1.5 Research objectives</td>
<td>5</td>
</tr>
<tr>
<td>1.6 Outline of this document</td>
<td>6</td>
</tr>
</tbody>
</table>

**Chapter 2: Literature Review**

<table>
<thead>
<tr>
<th>2.1 Mining automation</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.2 Mobile robot navigation</td>
<td>9</td>
</tr>
<tr>
<td>2.3 Sensing strategies and sensor technologies</td>
<td>10</td>
</tr>
<tr>
<td>2.4 Simulation &amp; modeling</td>
<td>11</td>
</tr>
</tbody>
</table>

**Chapter 3: Introduction to the research**

<table>
<thead>
<tr>
<th>3.1 Bird’s eye view of the research</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.2 Research tasks</td>
<td>13</td>
</tr>
<tr>
<td>3.3 Introduction to the simulations developed</td>
<td>13</td>
</tr>
<tr>
<td>3.4 Sensor technologies and the interface</td>
<td>14</td>
</tr>
<tr>
<td>3.5 LabView and the control interface</td>
<td>16</td>
</tr>
<tr>
<td>3.6 Outline of control algorithm – Requirements and needs.</td>
<td>17</td>
</tr>
</tbody>
</table>

**Chapter 4: Simulation**

<table>
<thead>
<tr>
<th>4.1 Manual track speed control of a single MBC</th>
<th>19</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.2 Path Generation</td>
<td>22</td>
</tr>
<tr>
<td>4.3 Single MBC path following</td>
<td>24</td>
</tr>
</tbody>
</table>
4.4 Path feasibility analysis – Accuracy ellipse method 25
4.5 Dynamic Mine Map (DMM) 30
4.6 Dynamic Path Planning (DPP) 32

Chapter 5: SICK AND PMS 34
5.1 SICK Laser Scanner 34
   5.1.1 SICK data description 34
   5.1.2 Line finding techniques – feature extraction from raw data 36
   5.1.3 SICK Checksum code generation 38
5.2 PMS Laser Scanner 38
   5.2.1 PMS data description 39
   5.2.2 Comparison with the SICK 40
   5.2.3 PMS line finding 40

Chapter 6: Control Software development 42
6.1 CIN development 42
   6.1.1 The CIN interface 42
   6.1.2 CIN inputs and outputs 44
   6.1.3 Functioning of the CIN 45
6.2 LabView VI’s developed for the CHS automation 53
   6.2.1 Single MBC with the SICK scanner 54
   6.2.2 PMS Control through the interface 55
   6.2.3 Multi-MBC VI with PC-PC data sharing 56

Chapter 7: Control Algorithms 63
7.1 “Where Am I” – MBC Localization 64
   7.1.1 MBC localization using UT sensors 64
   7.1.2 Line-finding 66
   7.1.3 Line finding applied to the laser scanners – Simple Line finding 70
   7.1.4 Iterative end point line splitting 71
7.2 “Where should I be” – Path planning 74
7.2.1 Simple path planning – Geometric approach 74
7.2.2 Mine plan dependant path planning – 90 deg mine plan 74
7.2.3 Implementation of the DMM and DPP simulation results 75

7.3 “How do I get there” - Motion control 76
7.3.1 Simple Motion control 76
7.3.2 Path Following – SSS Algorithm 79
7.3.3 Analysis of dynamic motion planning 82

Chapter 8: Conclusion 83

Chapter 9: Future research 85

References 86
Figures

Figure 1: The Continuous Miner in attached configuration with the first MBC..................2
Figure 2: The CHS entering a mine for operation.........................................................3
Figure 3: A snap shot of the system in a 90 degree mine layout.................................3
Figure 4: Screen shot of the velocity control simulation.............................................21
Figure 5: Velocity profile from the velocity control simulation.................................21
Figure 6: A typical mine plan with dimensional parameters......................................24
Figure 7: Screen capture of the automatic MBC path following simulation..............25
Figure 8: The ‘Deviation Envelope’ for a single link system in a 90 degree mine plan.....30
Figure 9: Screen shot of the ‘Dynamic Mine Map’ simulation routine.....................31
Figure 10: DPP results with a Cubic Bezier curve for the path...................................33
Figure 11: The Dynamic Map creation technique used in the DPP simulation...........33
Figure 12: SICK® data structure..................................................................................35
Figure 12 a: A data plot from a single scan using the SICK® laser scanner.............36
Figure 13: Appropriateness of Line-finding to the raw SICK® data............................38
Figure 14: PMS® data structure..................................................................................39
Figure 15: A typical Virtual Instrument block diagram showing the Code Interface Node.....43
Figure 16: LabVIEW® CIN with inputs and outputs..................................................45
Figure 17: A default Code Interface Node with no object code.............................49
Figure 18: CIN with 3 input-output rows.................................................................49
Figure 19: A CIN with complete inputs and outputs wired.....................................50
Figure 20: CIN frame for a Single MBC control.......................................................54
Figure 21: Sub-VI block process for serial data communication.............................58
1.1. Underground coal mining equipment

Coal is the primary source of energy for the global economy, and coal mining has existed since the early eighteen hundreds. There has always been an increased demand for coal every year and coal-mining companies have been exploiting new technologies to increase the production volume to meet the rising demand. The equipment in this industry has undergone decades of changes and improvisation, to evolve from the early human powered machines to today’s fully autonomous equipment.

In a general sense, mining of coal refers to underground coal mining, and this process involves the removal of the coal from a seam layered between rock and earth. The thickness of the coal seam ranges from 24” to 70” and a variety of machines have been used to extract coal from this environment. The use of a continuous miner has resulted in a very high efficiency based on the volume of coal mined in a very short time. The traditional practice in coal mining for transferring the mined coal from the coalface to the exterior of the mine is via conveyors and dump trucks. These machines were sluggish in operation and there exists a time lag between the cutting operation and clearing due to the long distances of travel to transport the mined coal. To overcome this difficulty, Long-Airdox \(^{(1)}\) (LA) pioneered the development of the Continuous Haulage System (CHS) that combined the process of both cutting and transferring coal into a single process to be performed by a single machine. The use of this new technique has proven beneficial to coal mining companies with the quantum of coal mined increased multifold. This completely revolutionized the mining industry as the efficiency and speed of coal mining increased, with a substantial reduction in manual labor.

1.2. The Long-Airdox Continuous Haulage System (LACHS)

The Long-Airdox continuous haulage system is essentially a series of drivable belt conveyors linked together via tracked vehicle Sections. It consists of multiple units of Mobile Bridge Carriers (MBC) interconnected by Piggy Back Conveyors (PIG) that extend over a
length of 32’- 40’. The system in full configuration (5 units) extends to a length of 300’. At the foremost end of this MBC-PIG chain, is a continuous miner; a cylindrical cutter supported on tracks for mobility. The continuous miner is very similar to the MBC except for the additional rotary cutter. At the rear extreme end of the chain, the last PIG floats over another long conveyor called the ‘Rigid Frame Modular’ or the RFM. The miner may be either physically attached to the first MBC or may be linked via the feeder breaker. The former configuration is termed as “attached” and the latter as a “detached” system.

Figure 1, shows the continuous miner in an attached configuration. The continuous miner performs the operation of cutting coal at the face and the coal stream is continuously transferred through the MBC’s, via the PIG’s and finally to the exterior of the mine. Figure 2, illustrates the system entering a mine.

Thus, the coal transfer rate is the same as the cutting rate, and this reduces the time lag that existed in traditional machinery. The MBC’s are capable of independent control and are controlled by human operators. It is a tracked vehicle and can be compared in physical form and
structure to a military tank. Thus, there is an operator on each MBC and one on the
miner. The function of the MBC is to support guidance and navigation of the whole chain into a
mine entry and to perform cutting operation at the face. The lead MBC operator guides the
system into a mine entry and all the trailing MBC drivers follow this operation. Once the miner
has reached the face, the cutting operation begins and the coal is mined and transferred.

Figure.2. The CHS entering a mine for operation (1)

Figure.3. A pictorial view of the system in a 90 degree mine layout
This operation is called a ‘cut sequence’. After one such sequence, the whole chain is then moved to another entry or progressed down the same entry. Figure 3. shows the snapshot of the system in a mine layout.

1.3. Uniqueness of the CHS

The LA continuous haulage system is unique in many ways. Not only does one system replace multiple units of conventional machinery, but it also reduces the amount of labor and capital involved. The Long-Airdox CHS is the first of its kind among coal mining machinery that possesses the full chain concept, with multiple units tethered together. The bigger the system, the more complex it is in terms of control. Highly skilled MBC operators are required to drive the MBC’s to navigate this whole chain in the mine without colliding against the walls. This task is not trivial considering the massive length of the system and the mechanical constraints.

The system, when viewed from the perspective of mobile robotics, possessed various disadvantages. The length and the mechanical constraints of the joints in the system make it almost impossible to perform certain directional moves. Moreover, as the MBC’s are tracked vehicle units, an additional constraint was introduced in the guidance and navigation. Amnart Kanarat’s (2) detailed research on tracked vehicles and simulations of tracked vehicle dynamics illustrate the complexity of the CHS. The presence of pivot pins at both the ends of the MBC that connect with the front and the rear conveyor Sections introduces an additional constraint on the directional control of the MBC. The MBC drivers had to coordinate with each other to drive the chain into the mine, with no proper means of communication between them. They had to employ intuitive methods to navigate, either by looking at the side clearances or by observing the pattern of motion of the leading vehicle.

The only communication between the operators was an emergency stoplight that would instruct all the drivers to stop the chain in case of an extremity or emergency. Though mine safety has been greatly enhanced, the potential for accidents are very common with a very complex system.
1.4. Problem Statement

To address all these and other potential concerns described in the previous Section, Long-Airdox expressed the need for automatic control of the Continuous Haulage System to increase system efficiency and to eliminate human operators, thus reducing danger to both the operators and the equipment. The result of this proposal from Long-Airdox to automate the CHS was the initiation of a collaborative project between Virginia Tech and Long-Airdox (LAVT). The Virginia Tech research team’s responsibilities included the design and development of automation strategies for the complete automation of the CHS. This task involved reviewing both current and previous literature on various existing technologies for mine equipment automation and to develop new strategies for the CHS. The result of literature reviews indicated the need for a physical system that would allow experimentation of technology for the CHS. This task of developing a fully functional 1/10th scale prototype of the LA-CHS (Long-Airdox Continuous Haulage System) was accomplished by Bruce Wells (3). The fully functional prototype developed was designed to mimic most of the physical characteristics of the full dimension haulage system. MBC’s and PIG’s were fabricated from sheet metal and the prototype was mounted with sensors for control and navigation. The availability of this prototype served the main purpose of a test bed for the testing of interface software and control algorithms, the main focus of this research. A heavy emphasis in the discussion is placed on the development of path planning techniques and control algorithms for the CHS to enable a fully autonomous operation in any mine environment. The problem being addressed in this research is the design and development of control algorithms for the CHS.

1.5. Research Contributions.

As a result of the extensive research over a span of 2 years, the results achieved in this automation effort have proven to be satisfactory on both the prototype and full-scale systems. This document serves the purpose of both a reference and a base document for future work on this automation effort. The major contributions of this research can be summarized as follows.
i. Simulations in MATLAB® to analyze path-planning strategies developed for the guidance and navigation of the LACHS prototype system.

ii. Sensor data acquisition interface in LabVIEW® for both the SICK® and PMS® Laser scanners.

iii. A control interface in LabVIEW® that allows direct communication with the MBC hardware from a host PC. This interface also acts as a data communications link between the control algorithms and LabVIEW® via a methodology called as Code Interface Node (CIN).

iv. Control algorithms for the automation of both the 1/10th scale unit and the LA full-scale system.

1.6 Outline of this Document

The remainder of this document is divided into eight chapters as follows:

- Chapter 2 presents a detailed review of the literature on four different areas – mining automation, mobile robot navigation, sensor technologies and associated sensing strategies, simulation and modeling of mine equipment. This will act as an external reference to this research.

- Chapter 3 presents a bird’s eye view of the problem dealt in this research. An outline of simulations developed, the control algorithms and the requirements of these algorithms are discussed briefly to introduce the reader to the crux of the problem. An introduction to LabVIEW® is provided in this chapter to introduce the software development concept. This chapter also throws light on the various sensor technologies that were considered and tried for use on the CHS and their associated sensing strategies viz. the use of the sensor data to design control algorithms.

- Chapter 4 deals with the various simulation routines developed for this automation effort. MATLAB® simulation routines are discussed in detail and their application in developing the control algorithms are also dealt with in this chapter.

- Chapter 5 is dedicated to the different sensory systems used for navigation and guidance for the CHS prototype and the full-scale system. This chapter discusses the three main
sensor technologies viz., Ultrasonic, the SICK® laser scanner, and the PMS® laser scanner. The discussion spans the sensory system’s data structure, its interface with the control interface and the nature of the data.

- Chapter 6 is a detailed discussion on the control interface and the control software development phases. LabVIEW® Virtual Instruments created for the CHS automation are discussed in detail. This chapter also introduces the main interface engine between the LabVIEW® block diagram and the pre-compiled object codes written in C for implementing the control algorithms.

- Chapter 7 concentrates on the various algorithms developed for the CHS automation. The discussion introduces the classic 3-Question philosophy of “Where am I”, “Where should I be” and “How do I get there” in mobile robotics. The text in the discussion aims at providing answers to these questions, and thus resulting in automation algorithms for the LACHS.

- Chapter 8 briefly discusses the implementation of all the algorithms and the simulation results developed by the author as a part of the automation effort and touches upon the future tasks.

- Chapter 9 deals with future research that can be leveraged off the work in this research to generate generic automation control algorithms that can be modified to suit the needs of other similar systems resulting in either full or partial automation.
CHAPTER 2: LITERATURE REVIEW

Mine equipment automation is a potential area for automation due to the opportunities that exist among the mining equipment. In general terms, automation can be applied to systems to increase their operating efficiency, to remove the interference of humans in dangerous environment or to obtain high levels of accurate control of the operating machinery. Underground mining satisfies most of the above conditions and hence there has been a lot of work on mine machine automation and related areas. In this Section, some of the technical papers related to our main focus of mine equipment automation are discussed. This chapter focuses on four major Sections and discusses in brief, the technical literature reviews in these areas.

2.1. Mining Automation

The term “mining automation” is used to describe the automation of mine machinery or underground mine equipment. Peter I. Corke (4) et al, in their paper ‘Modeling and control of a 3500 tonne mining robot’ describes the automation techniques developed to control a Dragline. Draglines are extremely large machines that are widely used in open-cut coalmines for overburden stripping. This literature reviews their efforts in developing a computer control system capable of automatically driving a dragline into a mine. The use of an IR scanning laser range finder is motivated in this paper. The scanner acts as the main sensing element to measure the range of the bucket from the dragline.

One of the many interesting pieces of literature in areas of algorithm development is the discussion of the design and development of a motion control algorithm for a continuous mining machine by Hui-Min Huang (5) et al. The paper discusses the development of control architectures for the control of the movement of continuous mining machines (only tram control). The tramming control algorithm design is based on the hierarchical architecture design principles developed at National Institute of Standards Technology (NIST), referred to as the Real-Time Control System (RCS). The algorithm design allows for the control of both cutting and free space movement by a continuous mining machine and at the same time, allows for a
high degree of human operator interaction. Obstacle avoidance is assumed to be a human operator function.

Jonathan M. Roberts et al \(^6\) in their work describe in detail both the sensing and control techniques involved in the automation of underground truck haulage. Haulage trucks are dump trucks that transfer mined coal from the coalface to the outside. This publication focuses on reviewing the state of art techniques for haulage truck automation.

### 2.2. Mobile robot navigation

This Section briefly describes the content of a few technical literatures in areas of mobile robot navigation. As the CHS is viewed as a serpentine robot, review on mobile robot navigation will be of great value in designing the control algorithms.

Gary K. Shaffer \(^7\) et al describe in detail, position estimation techniques for underground mine equipment. The paper concentrates on the existing approaches and also discusses about estimation of position from sensors like beacons, gyroscopes and wheel counters that estimate position directly.

The paper by Hu \(^8\) et al presents an experimental mobile robot that is designed to operate autonomously within both indoor and outdoor environments. A sensor based autonomous navigation architecture for a dynamically changing environment is described in this work, and also various methods for robot path trajectory generation are discussed.

Doty and Seed \(^9\) present a review of the research work on a low cost autonomous robot that creates a landmark map of the environment through exploration and object identification. The robot is equipped with a short-range proximity IR detector and the results of the experimental work are also described.
Leo Dorst\textsuperscript{(10)} et al present a detailed description of the path-planning problem for arbitrary devices as a geometric problem. A path-planning algorithm based on flow and vector fields are described.

A control system for an autonomous robot cart designed to operate in a well-structured environment is the main focus of the paper by Nelson and Cox\textsuperscript{(11)}. They present a detailed description about the control system developed for this indoor robot and also the algorithms used for remote operation.

### 2.3. Sensing strategies and sensor technologies

Sensor based motion planning by the implementation of generalized voronoi graphs is described by Howie Choset\textsuperscript{(12)} et al. They present mathematical models for modeling sensors and dealing the path-planning problem by the use of voronoi graphs.

Charbonnier and Strauss\textsuperscript{(13)} present an efficient technique for interpreting laser range finder data. A suitable polygonal approximation for laser rangefinder data from a scanner mounted on an indoor mobile robot is presented. The concept of modeling the walls as straight lines and approximating the environment is the main focus.

Skrzypczynski\textsuperscript{(14)} describes in his paper the technique of ‘iterative end point line splitting’ for producing environment maps for a mobile robot equipped with a laser range finder. He also describes how to build a world model for the robot in an unknown environment.

Gonzalez, et al \textsuperscript{(15)} present the map building techniques for a mobile robot equipped with 2D laser range finder.

Ultrasonic sensor model and configuration simulations for mobile robot navigation in narrow aisles is the main focus of the technical review by Hogan\textsuperscript{(16)} et al. They present various optimal sensor locations for a mobile robot based on simulation results.
2.4. Simulation & modeling

This Section briefly discusses the simulation software that was explored for use in this research. One important technical review in this Section is the ‘accuracy ellipsoid’ method developed by Kotulski et al in their paper. They present a detailed simulation methodology in analyzing the problem of the positioning accuracy of robot manipulators. For any robot manipulator, it is important to know how far random deviation of the hand may be from the desired position if the joint positioning errors possess a normal distribution. They describe two methods for this process and the first technique is in determining the accuracy ellipsoids of the tip when the joint angles and joint positioning errors are known.

Other commercial and educational simulation packages for mechanism and robot simulation have been evaluated. Some of the packages that were explored as a preliminary work on this research include Popbugs – a simulation based environment for track driven robots developed by Chris Thornton at Sussex University in United Kingdom, DADS track module, ADAMS full simulation package, RobotAssist, Working Model 2D and 3D and 3DS Max 2.0. All of the above packages were of not much use in this research, as the complexity of the CHS, both dynamically and in terms of kinematics was not supported by the above simulation engines. Moreover, as the main focus on this research is to simulate the CHS just to evaluate the developed control techniques without much emphasis on 3D graphics, writing custom code seemed to be the best choice.
CHAPTER 3: INTRODUCTION TO THE RESEARCH

The content of this chapter aims at introducing the complete nature of the problem, the research efforts and other related tasks.

3.1. Bird’s eye view of the research

As discussed briefly in Chapter 1, the LAVT automation effort aims at achieving a fully autonomous Continuous Haulage prototype system. The main theme of the work is to develop a robust control hardware and software, test these systems on the prototype and upon successful and satisfactory results, port these systems to suit the full unit system. The LAVT team has spent a major portion of time working on developing and proving the systems on the 1/10\textsuperscript{th} scale prototype, and through proper documentation and technology transfer, will be handed over to company personnel at Long-Airdox for work on the full scale.

Although a small portion of this document will cite the field-testing of the algorithms on a single unit (Full scale) at the Long-Airdox field site, main concentration is on the prototype system only. The work done as a part of the LAVT automation effort involved the construction of a 1/10\textsuperscript{th} scale laboratory test system to create an experimental test bed. Bruce Wells\textsuperscript{(3)}, describes both the prototype mechanical system and the basic electronics on-board the system. The next phase was to review and exploration of current and previous literature on related topics as discussed in Chapter 2. As this research parallels any other research in a robotics field, the full picture of producing an automated system can be categorized as a robotics and control theory problem. As every robotics research beings with modeling the system to get a better understanding of the system kinematics and dynamics, efforts on the LAVT research also began with a similar process. Amnart Kanarat\textsuperscript{(2)} has described the detailed kinematics of the system and has substantiated his theory by developing simulation models.

A major portion of the following chapters of this document will concentrate on extending the base hardware interface system, initially developed by Bruce Wells, to suit modifications done as the research progressed, developing a new sensor interface for a home built Laser
scanner, extending the LabVIEW® Virtual Instruments for a multi-unit system, and developing and validating control algorithms.

3.2. Research Tasks

In continuation with the discussions in the previous Sections, the major tasks of this research document can be categorized as follows.

i. Simulations in MATLAB® - Path planning and path feasibility analysis
ii. Sensor Interface - A data acquisition interface on the host PC
iii. Control Interface - Interface between the host PC and MBC

The next few Sections briefly describe each of the above-mentioned categories.

3.3. Introduction to the Simulations developed for CHS Automation

Due to the complexity involved in the automation of the CHS, it was vital to have a thorough understanding of the system before embarking onto developing the control algorithms. Literature survey on modeling and simulation had revealed that simulation techniques are fast and accurate prototyping methods to study the behavior of a system. Thus, simulation became a very important tool in this research. The CHS was modeled, both mathematically and graphically and through simulation, a strong understanding of the system was obtained. The results of the simulation were used to build the ‘driving rules’, which upon further enhancement, led to the development of the actual control logic to be used on the system for automatic control.

The simulation performed for the CHS can be sub-categorized into two sub groups.

i. MBC simulations
ii. Sensor simulations.
This research focuses mainly on the MBC simulations. Simulation programs for MBC analysis were developed in MATLAB® with an emphasis on graphical screen output. Graphics are a very effective way of representing the system. They are an indirect medium of communicating the performance, with no prior exposure to the complex mathematical model. However, it should be noted that ‘Graphical representation is not just a blind animation sequence, but a structured collection of visuals that directly capture the results of the modeling stage.’ (18). The following MBC simulations were constructed and Chapter 4 describes each of these in greater detail.

i.  MBC Velocity control simulation
ii.  MBC Path following
iii.  Path validation and feasibility analysis
iv.  Dynamic Mine Map (DMM)
v.  Dynamic Path Planning (DPP)

The above listed simulation programs have the capability to control the MBC’s track speeds by a Graphical User Interface (GUI) and enable the control of both position and orientation of the MBC along the chosen path. Simulations were developed to validate the path used for navigation of the MBC.

3.4. Sensor technologies and the interface

Every autonomous vehicle requires the function of environment perception. This can also be termed as sensory perception. Sensors are required to aid the vehicle to identify and locate itself in a given environment. This is the “Where am I?” problem or the “Self localization” problem. The same applies to the automation of the LACHS. Eliminating human operators demanded the use of various sensor on-board the system. The sensory system of the LACHS can be classified into two main systems. One system dedicated for the guidance and navigation of the system, and the other to acquire analog data from the system.

The guidance and navigation system is a system that perceives the environment, or in other words acquires environment data. Once the data has been acquired in real time (when the
Aishwarya Varadhan  Chapter 3. Introduction to the Research

system is in the working environment), this data is then input to the algorithm via the sensor interface. After computation, the output from the algorithm is sent back to the system via the control interface. This cycle repeats until the system has reached its planned destination.

Sensors for the CHS were required mainly for the guidance and navigation of each MBC and the PIG. Apart from this primary function of navigation, sensors were also required for emergency routines and safety features. In this research, a higher emphasis is placed on the guidance and navigation system.

After a thorough review of the available sensors in the industry, the team with the expertise of Dr. Robert H. Sturges began work on two predominant technologies viz.

i. Ultrasonic
ii. Laser scanners – SICK® and PMS®

Both the above mentioned sensor systems are distance or range measuring devices. An Ultrasonic sensor is essentially a single point device, whereas laser scanners are sweeping range sensors. Both these technologies were explored and tested on the prototypes. Basic experimentation with the Ultrasonic sensors revealed that they were not appropriate for the CHS, as a single data point does not suffice for path planning. This work was done as a preliminary research on the 1/10th scale model to choose an appropriate sensor technology and develop control techniques for that technology. In addition, Amnart Kanarat (2) through software simulations, has validated the need for the SICK® as the main sensing element for the full scale CHS. The use of the LMS200 SICK® laser scanner for distance ranging by the Autonomous Vehicle Team at Virginia Tech, combined with preliminary data analysis was a convincing fact that the SICK® was a promising technology for the CHS. Further exploration on the use of SICK® combined with the simulation results, validated its need as the main sensing element on the CHS.

Continuous experimentation with the SICK® was required to develop a thorough understanding of its functioning and its appropriateness to the needs of guidance and navigation. Once the basic technology of “line-finding” was developed, more refinement was the immediate
need and this demanded a sophisticated data processing. Due to the heavy economic investment involved with these scanners, and as experimentation was carried over on the prototypes only, it was decided in the common interests of both the VT team and Long-Airdox that a new low-cost scanner be developed for use on the prototypes. The idea behind this new proposal was to utilize the SICK® scanners for experimentation with the full-scale equipment by company personnel.

This new proposal led to the development of a low-cost laser scanner by Dr. Robert H. Sturges and Todd Upchurch. The scanner essentially mimicked the SICK® laser scanner in most respects and was named as Poor Man’s SICK®. The task was then to develop “line-finding” techniques for this new scanner. The next chapter discusses the nature of this scanner and the algorithm requirements in detail.

Guidance and Navigation routines were developed for both these sensor technologies. Line-finding routines were developed for the SICK® LMS200, for use on the LA full-scale machine, and the PMS, for use on the 1/10th scale model.

3.5. LabVIEW® and the control interface

With the sensor technology chosen, and the hardware ready for testing, there was a pressing need for a powerful and an efficient software system to act as an interface between the MBC hardware and the control PC. Bruce Wells (3) discusses in great depth the reason for choosing LabVIEW®. In line with the discussion here, LabVIEW® was chosen due to its capability to accept pre-compiled code developed using any high-level language like C or Pascal. This research discusses in detail, the use of LabVIEW® as the main interface and its code interface modules with a standard C compiler for processing data.

The code interface module is responsible for requesting data from the scanner and other sensors, process this data and output track speeds to locate the MBC in a mine layout. Aishwarya Varadhan and Robert H. Sturges (19) describe a system that employs LabVIEW® for rapid development for mobile robot prototypes.
3.6. Outline of the control algorithm – 3 question philosophy

Simulation just mimics the functioning of the system, but does not actually control the system itself. The results of the simulation are very important to the researcher working on transferring its results to a logic structure that will enable real-time control of the physical system. This stage is the algorithm development stage.

Automation of any system involves the break down of the desired tasks and the goals to be achieved. This research will view the system from a robot builder’s perspective, taking advantage of the analogy between the CHS and an autonomous serpentine mobile robot. The problems involved in the automatic control of a mobile robot can be summarized by the classic 3-question philosophy:

i. Where Am I?
ii. Where should I be?
iii. How do I get there?

These can also be viewed as the necessary requirements that the algorithm has to handle in order to achieve autonomous control. The basic requirements for the algorithms based on the above three questions are

i. The algorithm must be accurate and easy for the MBC to estimate its current heading and position with reference to a fixed coordinate frame – Where Am I?
ii. It must be capable of deciding its next course and/or estimate its goal points – Where should I be?
iii. It must have an accurate control that will enable to track its course with high reliability – How do I get there?

All the algorithms developed for the CHS are based on the above theory. If the algorithm provides a satisfactory answer to these three questions, then it is sufficient to conclude the logic to be the basic ‘driving rules’ of the system.
The first and foremost task is to convert sensor data to useful data that enables the localization of the MBC in a mine environment. The second task is to use this MBC location data and determine a feasible path to follow. The final task is then to drive the unit along this path.

We can summarize the complete task of automating a system by the following stages in order of execution.

1. Initialization of software and hardware systems
2. Serial port initialization
3. Request for sensor activation
4. Sensor data acquisition
5. Self Localization – interpret sensor data useful form
6. Analysis – use the control algorithms to arrive at current MBC position
7. Path planning – plan a path for the MBC
8. Path verification – Check feasibility of the computed path
9. Motion control – issue track commands to drive MBC along this chosen path.
CHAPTER 4: SIMULATION

This chapter discusses the various simulation routines developed for the automation of the LACHS. Simulation is a powerful and rapid technique to obtain a thorough and near complete understanding of any system that can be mathematically modeled. Simulation begins with the process of mathematically modeling the system under analysis followed by a time-based or an event-based simulation to ‘simulate’ or mimic the functioning of the system. The second vital component of a simulation is the visualization or the graphical display. This research places an equal emphasis on both these processes, as graphical display is a very effective tool to obtain a very quick understanding without having to look at complex mathematical equations. Many techniques in robotic studies involve graphics and classic examples like the C-Space technique may be quickly understood by screen graphics than theoretical explanation. MATLAB®, a powerful modeling tool from MathWorks® was used to develop all the simulation routines. The selection of MATLAB® as the main simulation development tool can be attributed to its easiness in use and its effective graphical display capabilities. All the simulations developed utilize the mathematical model and employ these results to produce on-screen animations. Amnart Kanarat (2) in his documentation on the LACHS modeling describes in great depth, the kinematics and the dynamics of the system. This research employs some of these results as the base to develop the simulations.

4.1. Manual Track Speed control of a single MBC (Velocity control)

The automation of the LACHS system required a thorough understanding of the kinematic and dynamic characteristics of both the MBC and the PIG. In the conventional style manually driving the system, each MBC operator is supplied with a set of controls on the control console in the driver cab, which serves the purpose of navigating the MBC in the mine roadways. The two important controls were the left and right track speed controls coupled with other functional controls for raising the in-by and out-by booms. The MBC operator, with the aid of these two track speed controls, essentially a hydraulic valve to control the fluid flow, regulates the speed of the tracks. Directional turns are achieved by differential speeds between the two
tracks, and the advantage an MBC has over other similar wheeled vehicle systems is the ability to make zero radius turns (about its own center).

To introduce an automated system to control the track speeds, it was required to have a complete knowledge MBC’s reaction react to different types of curves, turns and track speeds. As the full scale MBC was not available on demand to conduct experiments, simulation was a logical choice to resort to. Moreover, due to the fact that the control algorithms were designed to be directly based on the simulation results, it was chosen to perform a track speed control simulation of the MBC.

The designed and intended purpose of this simulation was to obtain a visual sense of the reaction of the MBC to different types of curves, track speeds and turning radii. Manual track speed control was achieved by the use of GUI slider bars. A single MBC (modeled as a rectangle) following a sample pre-laid path was achieved by manually controlling the left and right track velocities via the velocity sliders. The MBC was modeled as a rectangle to keep the simulation low in detail, and at the same time to obtain an understanding of the path following capability of the system. A combination of piecewise segments of circular arcs and straight lines chained together constituted the complete path. Different combinations of arc radii were chained with straight-line segments to create a variety of paths, and with the aid of the velocity controls, the MBC was positioned along the path.

The simulation was time based, and the program allowed the alteration in the total running time. The program’s capability to write the MBC position, angle, and the left and right track speed data to a file made this analysis very important in the CHS automation. This ASCII data file was then used to arrive at velocity profiles (Time Vs Velocity curves) for the individual tracks and various profiles for various kinds of turns were obtained and plotted. The results from these velocity profiles were very crucial as these dictated the maximum and minimum working speeds, and the nature of the path, which the system would be able to follow.

After analysis, it was concluded that the ideal MBC path would be composed of segments of straight lines and arcs chained together. This simple path composition enables a reliably
simple path computation process without involving complex parameterization or higher degree curves. The idea behind this effort was to base the initial tests on a simple path and to observe the system’s performance on this sample path. Figure 4 below illustrate a sample screen from the simulation. Figure 5, illustrates a sample velocity profile of the MBC tracks when negotiating a curve. Appendix I (i) provides the source code listing for this simulation.

Figure. 4. Screen shot of the velocity control simulation

Figure. 5. Velocity profile from the velocity control simulation
It should be noted that this simulation does not take into account, the kinematics of the system, nor the ground conditions into consideration. The purpose of this is to obtain a feel of what the path would look like and how the system would react to that path. This essentially mimicked the MBC operators by replacing them by the GUI slider controls.

4.2. Path generation

As discussed in the previous Section, navigation of the automated system required a path for the MBC to follow. This path can be viewed as a virtual path laid on the mine floor. The algorithm will use this path to plan the set of moves required to position and orient the MBC. Some of the desired qualities for this path are that it should be a “collision free” path, (i.e.) a path that will always result in a 100% positioning of all the pivots (the pin joints) at all times without the PIG or the MBC colliding against the mine walls. The path has to feasible, (i.e.) that it should be negotiable by the MBC-PIG pair without resulting in extremities in the configuration.

Generation of this path requires both the geometry of the system and its working environment. For simplicity, it can be assumed that a single MBC-PIG pair will be sufficient to arrive at this path. The theme behind this assumption is that this MBC-PIG chain repeats itself resulting in a serpentine system. If one such link in the chain is able to transcribe a path, then it is logical to conclude that every link in that chain will also transcribe this path. Since the path is a function of the system’s geometry in combination with the mine environment, mine plans; plans that describe the mine parameters (pillar width, cut angle and roadway width) were required. To arrive at this path, various configurations (combinations of lines and arcs of different radii) were superimposed on a mine layout (a 90 degree mine plan to start with) and the manual track speed simulation (discussed in the previous Section) was used to choose the collision free path. The initial development of this simulation focused on a single mine layout (a 90 deg mine plan) and once confidence in results for this mine plan was obtained, paths for other mine plans were planned for development with the assistance of company personnel in providing all the mine plan configurations. This research describes the mine path only for the selected 90-degree mine plan. The same philosophy of superimposing the path on a mine layout combined with the manual
track speed control can be used to generate paths for the other layouts. The path for a 90-degree mine plan was developed initially as this mine layout was considered as the most difficult one to navigate the system through, due to the presence of square pillars (with filleted corners), and narrow aisle ways. Though Long-Airdox records revealed that the longest MBC-PIG pair was not used on this layout, this configuration was ideal to arrive at the path for the worst case, which once proved to be feasible and negotiable will prove to be feasible for all the other configurations.

**Path Terminology**

The following terminology will be used to describe a typical path generated for any given mine plan.

- **Pillar width (P)**: The width of the mine pillar
- **Mine angle (θ)**: The cut angle
- **X road (XW)**: The roadway width along the X-axis
- **Y road (YW)**: The roadway width along the Y-axis
- **X Center dist (XD)**: The distance from the center of the roadway to the arc center in X-axis.
- **Y center dist (YD)**: The distance from the center of the roadway to the arc center in Y-axis.
- **R**: The radius of the curve segment (a quarter circular arc)

Note: *All units for all the above parameters are in ‘feet’ to be inline with LA’s unit system.*

The path generated for the 90-degree mine plan is as shown in Figure 6. This Figure illustrates the above-discussed terminologies for arriving at the initial path. The drawing is an exact replica of the mine plan that has 40’ square pillars and 90 degree pillar angles. In comparison with the other mine patterns, the 90-degree plan posed various challenges to path planning due to its square corners and the narrow roadways.

Figure 6 also show a snap shot of the LACHS to depict the visual difficulty in positioning the system around a corner.
4.3. Single MBC path following

Once the path is computed from the technique described in the previous Section, and verified by the manual track speed control simulation, the next step is to analyze the automatic control of a MBC on this path. This is a more realistic test of path feasibility and verifies the system’s capability to traverse this path. Thus, the manual track speed control simulation was modified to create a new routine that would enable automatic control of the MBC position and orientation along the chosen path. This simulation is an extension of the above described “velocity control” simulation. The main theme was to model a single MBC in a 90 x 90 mine plan with 40’ centers and 20’ roadways (as shown in Figure 6). The path used for the simulation was the path computed in simulation described under Section 4.2. The program has the capability to automatically (unlike the manual positioning as in Section 4.1) position the MBC along this path and with the use of a graphical output (animation), the feasibility of the path was justified. A screen shot of this simulation is shown below in Figure 7. It can be observed that a MBC is shown to follow this path. The program also has the capability to change the length and widths of the MBC. All parameters used in this are real values obtained from LA.

Appendix I (ii) provides the source code listing for this simulation.
4.4. Path feasibility analysis – Accuracy ellipse method

In the above three steps, we computed a collision free path and studied the effects of positioning the MBC and a PIG along a chosen path. Based on these results, a prototype control algorithm was developed and tested on the 1/10\textsuperscript{th} scale model. Though the prototype system responded to the algorithm’s commands in tracing this virtual path, there existed some amount of invariability and uncertainty in trying to trace this path repeatedly. This raised concerns regarding the system’s abilities to trace this path in terms of space and joint constraints. Though the simulations in the previous Sections verified the feasibility of this path, a piece of information appeared to be missing. Computing the path did not imply that it is always 100% foolproof in terms of MBC/PIG positioning. It is required to measure qualitatively and quantitatively the response of the CHS to this algorithm that instructs it to follow the computed path. Intuitively it can be concluded that the accurate positioning on the path is not possible with
such a system due to the amount of slip that exists in the tracks, combined with possible sensory
noise. Amnart Kanarat’s (2) analysis on tracked vehicle dynamics and slippage clearly show that
it is a difficult task for the CHS to respond exactly to the commands due to kinematic and
dynamic principles of the system. From all these results, we can conclude that at any point in
time, there will always exist an error in positioning the system (the deviation from the ideal
path). So the question of “how much can the system deviate from the ideal path?” and still
remain ‘safe’ (collision free) remained to be answered. To address this problem, it was proposed
to develop another simulation based program to identify the allowable deviations (at every point)
from the ideal path, so that the system would be able to get back on the path without any adverse
effects (collision).

To solve this problem, the technique of “Accuracy ellipses” was used. The simulation is
based on the paper titled “On two methods of determining the ellipses and ellipsoids of
Kotulski (17) et al.

The Technique

The technique of computing the Accuracy ellipse is a classic robotic methodology in
which the accuracy of positioning the end-effectors or the tool is computed based on the position,
major and minor axis lengths of the accuracy ellipse. This technique assumes a standard error
distribution for the joint positioning, and also assumes that the tool hand traces a known
trajectory. Given these details, the methodology aims at computing the ‘deviation envelope’ for
this tool tip about its trajectory. The motivation for this kind of study is the error existing in
tracing a known trajectory. For the tool tip to trace a defined trajectory, the industrial robot
controller computes the joint trajectories (joint angles Vs time) for all its joints via the inverse
kinematics relationship. Once the joint angles are computed, the controller aims at controlling
the servo or motor drives at the joints to position the tool tip at the desired point on the path. In
most cases, due to kinematic constraints and errors in positioning, the joint angles are not always
100% accurate, and hence this error in the joint positioning migrates to the end-effectors thereby
causi ng a positional error at the tool tip. Thus, at every point in the path where positional
accuracy was desired at the tool tip, there existed some kind of deviation or error from this ideal
or desired position. Repeating this for the full length of the path, we can define an error envelope for the whole path, and the accuracy ellipse technique aims at computing this envelope. Once the error envelope or the deviation envelope is computed, this gives a complete description of how the tool tip can be positioned to minimize error.

The paper discusses the technique of computing the accuracy ellipses for a robotic manipulator with ‘n’ links. Though the paper outlines the method for determining joint positioning errors in industrial manipulators, the author has modified this technique to suit the LACHS simulation needs by making the assumption that at any instant of time (t) along the path, a MBC-PIG pair can be treated as a 2-link manipulator, with the pivot joints on the path. The main idea behind this was to compute the accuracy ellipses (lengths of major and minor axis) at every point on the path by considering the MBC to follow this path.

The result of this simulation was a contour that identified the critical points on the path. A potential advantage of this simulation is that the result obtained can also be used for the PIG’s by just altering the length of the link. It must be noted that initially both the MBC and the PIG were be treated as lines (with no width), and in later stages of development, the width of the system was included. It should also be noted that in this simulation, only a MBC or a PIG is considered to arrive at the deviation envelope. The previous discussion about using an MBC-PIG pair is not used in this simulation, as there is no significant difference between using a single system as compared to a pair.

The mathematical formulation behind the computation of these accuracy ellipses is explained below.

Let ‘L’ be the length of the link under analysis, and in this case, either a PIG or a MBC. If this system is positioned at the start of a sample path, then it is desired to find the positional error of the other end of the link along the same trajectory. The simulation assumes a global coordinate frame and aims at positioning the near end of the MBC/PIG on this path. The path considered here for analysis is the path computed for a 90-degree mine plan. The aim of the
methodology is to position this single link on the path and to locate the errors at the far end of the link with respect to this path.

At any instant of time ‘t’, let us assume that the link be positioned at a point ‘P’ on the path. Let (X, Y) represent the position coordinates of the end of the link with respect to the fixed end. Let us also assume that there is a probability ‘p’ associated with the positioning of the joints so as to arrive at the exact end-effector position and let \(\sigma_0\) represent the standard deviation of the joint error (in radians). If \(\theta\) is the angle of inclination of the link with reference to the global frame, then the coordinate of the end of the link is given by...

\[
X = L \cdot \cos(\theta) \\
Y = L \cdot \sin(\theta)
\]

Taking the partial derivatives of the end-effector position w.r.t the joint angle \(\theta\), this gives the Jacobian of the link.

\[
\frac{\partial X}{\partial \theta} = -L \cdot \sin(\theta) \\
\frac{\partial Y}{\partial \theta} = L \cdot \cos(\theta)
\]

The elements of the matrix of second order moments \(^{(14)}\) are as follows.

\[
\lambda_{xx} = \left(\frac{\partial X}{\partial \theta}\right)^2 \cdot \sigma_0^2 = L^2 \cdot \sin^2 \theta \cdot \sigma_0^2
\]

\[
\lambda_{yy} = \left(\frac{\partial Y}{\partial \theta}\right)^2 \cdot \sigma_0^2 = L^2 \cdot \cos^2 \theta \cdot \sigma_0^2
\]

\[
\lambda_{xy} = \left(\frac{\partial X}{\partial \theta} \cdot \frac{\partial Y}{\partial \theta}\right) \cdot \sigma_0^2 = -L^2 \cdot \sin \theta \cdot \cos \theta \cdot \sigma_0^2
\]
Now, let us refer to the following...

\[ D = \lambda_{xx} + \lambda_{yy} \]

\[ |k| = \lambda_{xx} \lambda_{yy} - \lambda_{xy}^2, \ k < 0 \text{ or } k > 0 \]

Thus, the major and minor axis of the accuracy ellipse at the tool tip or the end-effector is given by...

\[ a^2 = -\ln|p - D + \sqrt{D^2 - 4|k|}} \]

\[ b^2 = -\ln|p - D - \sqrt{D^2 - 4|k|}} \]

As all the units of lengths are in FPS system of units, the major axis and the minor axis values are in the same units. Once these values are computed, the simulation routine graphs the ellipse. This process of computing the major and the minor axis of the accuracy ellipse is repeated for all the points on the chosen path by moving the MBC/PIG to those points.

When the accuracy ellipse is plotted for the complete path, it results in a contour or a boundary of the deviation (i.e.), the errors in the end-effector positioning. This in the case of the MBC/PIG results in the error contour or the ‘deviation envelope’.

Figure 8, following this description shows a typical screen output from the simulation. It can be observed that while the system is negotiating a left or right turn into a straight aisle way, the envelope thins down, thus predicting that the system cannot deviate any farther than this envelope. If at any instant the link travels beyond this envelope, it depicts that the system will be unable to make the next move on the path without resulting in collision against the mine walls.

Thus, the turns are very crucial for the system and the results of this graph are inline with the initial understanding of the navigational capabilities of the system. For a higher confidence, a tighter standard deviation in joint errors and probability of positioning will result in a thinner envelope. Thus, we can conclude that this is an appropriate technique for the LACHS system.
This research focuses only on experiments for a single link system, but the simulation routine developed has the capability to extend this technique to dual links or even more by very little modification. Appendix I (iii) provides the source code listing for this simulation.

Figure. 8. The ‘Deviation Envelope’ for a single link system in a 90 degree mine plan

4.5. Dynamic Mine Map (DMM)

In all the previous Sections, the path computation, feasibility analysis and the accuracy ellipse techniques for path validation and deviation envelope calculation have been performed on an unique mine layout. The example layout used in all previous discussions is a 90-degree mine plan. This acted as a starting point for the algorithm development so as to keep the initial work rapid and simple. As the algorithms were developed based on this mine plan and once tests on the 1/10th scale model proved to be satisfactory, it was required to compute similar paths for all the mine plans. From Long Airdox’s records, it was evident that there was about 50-60 different
mine layouts with varying pillar widths, varying cut angles and different aisle widths. To compute a feasible path for all these layouts, required the regeneration of the mine layout repeatedly, and this was a tedious and repetitive task. Hence, it was decided that a virtual mine environment can be in-built into the simulator, and thus be capable of changing its parameters to transform into a new layout. To accommodate these varying mine patterns, a new simulation routine was developed. This essentially re-created the mine environment for the navigation of the CHS. The simulation had the capability to change the mine parameters dynamically and create a new environment. This added in a great flexibility to the program as the path computation can be carried out for a lot of different layouts using just one simulation model. Appendix I (iv) provides the source code listing for this simulation. Figure 9 illustrates a sample screen shot of this simulation. All units are to scale as in the real world data.

Figure. 9. Screen shot of the ‘Dynamic Mine Map’ simulation routine
4.6. Dynamic Path Planning (DPP)

The previous Section discussed the aspect of creating varying mine patterns to apply the path computation routine. This simplified the process to a great extent by adding in the mine plan variability, but at the same time it should be noted that computing a feasible path is a tedious and time-consuming process. The process begins with the iteration of various combinations of arc and line segments and then tethering these pieces together to create a piecewise continuous curve for the system to follow. Due to the presence of a large number of mine patterns, and also due to the very fact that the system will have to depend on a ‘value’ for the mine parameter that is subjective to continuous change (while cutting of coal takes place), the methodology was not robust due to its direct relation with the mine plan parameters. Instead, it would be a much better and improvised method if every MBC could create a dynamic map of its surrounding world at its current location. This is the main theme behind the dynamic path planning routine. This simulation is built on top of the DMM simulation model. The DMM simulation model creates the mine plan dynamically and the DPP uses this data to come up with a feasible path for the system to follow.

Continuing with the discussion of creating a local map, the model proposes the idea that every MBC have a complete knowledge of the environment of its immediate neighbors – the immediate leading and trailing unit. With the aid of this world information, every MBC controller is expected to create a local map (representing a 3 unit chain) and then compute a path. Once this local map is created, the simulation has the capability to iterate line and arc segments to create a feasible path, or to use more complex continuous curves (like a Quadratic Bezier spline) to arrive at the path.

The author has just developed this new system, but not carried any experiments in creating a path using the DPP. It is anticipated that the team with the aid of this simulation tool will be able to create a dynamic path for the system, and that this will be employed as the map-making strategy in the control algorithms for both prototype and full scale testing. Appendix I (iv) provides the source code listing for this simulation. The screen shot of the simulation world with a cubic Bezier curve is shown below in Figure 10. Figure 11, is the methodology employed behind creating this dynamic mine map.
Figure. 10. DPP results with a Cubic Bezier curve for the path

Figure. 11. The Dynamic Map creation technique used in the DPP simulation
CHAPTER 5: SICK AND PMS

This chapter discusses the two main laser sensors used on the prototype and the full-scale system. The commercial laser scanner, the SICK® LMS 200 from SICK Optik, and the home ground version of this, the PMS® (Poor Man’s SICK) are used as navigation sensors on the LACHS. These sensors are distance-measuring devices and with one full scan, spanning 180 degrees, return the distance information of the environment in which the MBC is operating. These scanners are mounted on the MBC and each MBC is mounted with two of these sensors. This gives a total 360-degree view of the working environment.

5.1. SICK Laser Scanner

5.1.1. Sick data description

From the results of the simulations, a very good knowledge of controlling the system was obtained. The results of the simulation now had to be transferred to the full-scale system. This required that the MBC have some sort of sensor that can perceive the mine environment. From the discussions in the Dynamic Path Planning simulation, there arises a need to map the world local to a 3-unit chain. In order to perform this task of environment perception or mine mapping, distance or range measuring devices are required. Thus, the CHS needed sensors that would enable it to locate itself with respect to the mine walls in order for the “path planner” to compute the path for its navigation in the mine. After a thorough analysis of all the available sensors in the market, the “SICK® LMS200”, a laser scanner from Sick Optic, Germany, was chosen due to its robustness in the mine environment. Amnart Kanarat’s MATLAB® simulations clearly mimic the functioning of this scanner and his simulations use one scanner on each MBC and attempted to position the MBC in a straight-line between two mine walls. This approach was inline with the final goals of control and the main aim of using a navigation sensor is to perceive the world as it is and transfer this data to interpretable form for the MBC controller.

The SICK® is a 180 deg sweeping laser scanner that scans at the rate of 2 Hz and for every scan, it returns a bi-dimensional array of length 181 data points. Every data point in this 2 dimensional data array is the sweeping angle (in deg) and the corresponding distance (in inches)
the sensor picks up for every degree it scans. The SICK has provisions to change its sweeping resolution (every 0.5 deg). Bruce Wells (3) (1999, MS thesis in ME) has performed detailed research on the SICK® and his work reveals the options available for configuring the SICK® through the control software. The document composed by Wells discusses the SICK’s control elements in detail and also deals with the scanner’s control protocol.

The use of the SICK® scanner required an interface to be built for data acquisition. This was designed and built by Bruce Wells using LabVIEW®. This interface enabled the link between the host computer and the LMS 200. The interface also allowed the operation of all the necessary operations (initialization, serial port handshaking, and data acquisition) for obtaining the scanner data.

Figure 12, shows an example of the SICK data structure for one full scan. The scan sequence is a 1-degree increment scan spanning 180 degrees.

<table>
<thead>
<tr>
<th>Angle (deg)</th>
<th>Range (inches)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>12.6</td>
</tr>
<tr>
<td>1</td>
<td>12.4</td>
</tr>
<tr>
<td>2</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
</tr>
<tr>
<td>180</td>
<td>24.78</td>
</tr>
</tbody>
</table>
As the SICK was chosen as the main sensing device for use on the MBC, the next step was to study the nature of the data and to develop strategies for positioning the system using this data. To confirm the robustness of the SICK data, Dr. Robert H. Sturges and Todd Upchurch conducted a “data acquisition” test on the SICK in a mine to evaluate its performance in that environment. The data collected from this test was used to develop basic control strategies. A MS Excel data plot of the SICK data is shown in Figure 12.

![Figure 12. A data plot from a single scan using the SICK® laser scanner](image)

### 5.1.2. Line-finding techniques – feature extraction from raw data

Choosing the SICK® as the main sensing element was then followed by the exploration of techniques to use the data acquired by the SICK to generate useful data for the algorithms. The process of converting raw sensor data into an interpretable form and using this converted data form in navigating an autonomous vehicle is called ‘Sensor Based Navigation’. Sensor based navigation is a vital component of any control algorithm of an autonomous vehicle. It
immediately follows the task of sensor or environment perception. In the context of the LACHS, the need to develop a sensor based navigation strategy was very important for accurate location of the MBC in the mine environment. Developing such a strategy is the development of an algorithm that would process the raw data from the SICK, filter the noise and result in a mathematical form that can represent the mine walls. Reviewing literature on sensor-based navigation revealed that most researchers used techniques like terrain-mapping using sophisticated cameras, 3D world mapping, grid formation, and other complex techniques. The existing level of complexity in the CHS discouraged the LAVT team from using such complicated techniques unless there was a need. The goal was not to add any more complexity into the system. The approach was to develop a simple, but robust approach that would perform the efficient role of environment mapping, and at the same time keep the computation to a minimum level.

By observing the mine patterns, it was intuitive and obvious to approximate the mine walls by a series of straight lines connected to each other. Filleted corners would be interpreted as sloped lines connecting two other line segments straight, thereby forming a diagonal Section of a mine pillar. This idea combined with the sensor’s capability to scan over a 180-degree span appeared to be the simplest way to represent the world as a combination of different line segments. Thus the word “line-finding” was associated with the SICK.

To expand on this term, line-finding deals with the filtering of the SICK data and generating an equation for the line (that represents the mine wall) in the form \( Y = aX + b \), which would be later employed to compute the position and the orientation of the MBC at that location in the mine. This process is the heart and soul of the MBC driving algorithm. The details of various line-finding techniques developed for the CHS automation are described in Chapter 7.

Figure 13, compares the raw data plot obtained from a mine test, with the illustration of the appropriateness of ‘line-finding’ applied to this raw data. Data in this Figure are real-world data acquired in a data acquisition test of the SICK laser scanner in an underground mine. The flat portions show the presence of a mine wall, and the ragged edges of the data plot illustrates the undulations in the mine wall surface.
5.1.3. SICK checksum code generation

The SICK® laser scanner required a CRC16 Checksum for it to be configured in different modes. Detailed discussions of this can be found in the discussions of the SICK® scanner in Bruce Wells’s thesis document. To assist the computation of the SICK® checksum, a C-program was written (with the assistance of Dave Mayhew and Bruce Wells) to compute the CRC 16 Checksum for the algorithm used by the SICK®. Appendix II (i) lists this code. This checksum generation code is essentially an implementation of the code from the SICK® user manual.

5.2. PMS Laser Scanner

As discussed in the previous chapters, the use of the SICK® scanners on the 1/10th scale prototype prevented the simultaneous use of these scanners for LA personnel to explore its usage on the full-scale machine. Moreover, the heavy economic investment associated with each of these scanners discouraged the procurement of more units of the scanners before the technology was fully confirmed. This need for simultaneous usage of the scanners on the full scale and the prototype initiated the approval of a new proposal to build a low-cost scanner that would mimic the SICK® in most respects, but at the same time would keep the cost to a minimum. Dr. Sturges and Todd Upchurch built a new laser video scanner that mimicked the SICK® in its functioning.
This scanner was named as ‘Poor Man’s SICK’ after the original SICK®. The PMS® also scanned over a 180-degree circular span with a maximum range of 25”. The return data structure of the PMS® was the same as the SICK®, with a bi-dimensional array between the angle and the scanned distance.

5.2.1. PMS data description

The data output from the PMS® controller essentially imitated the SICK® data structure. The raw data from the scanner was a 2-dimensional array. The first column recorded the angle scanned (in degrees) and the second column, the corresponding range or distance. The only major difference between the SICK® data structure and the PMS® data structure is the difference in the angle increment during the scan. The SICK® performs a scan over a full 180-degree range in increments of either 0.5 degree or 1 degree, but the PMS® scanned over this full 180-degree range in angular increments of 3.75 degrees. This number is associated with the functioning of the scanner. Due to this 3.75 angle increments, the total number of data points recorded per scan was reduced to 49 from the original 181. Hence, the PMS® data array length was only 49 data points, with each point corresponding to the angle increment. Figure 14, illustrates the PMS® data structure.

<table>
<thead>
<tr>
<th>Angle (deg)</th>
<th>Range (inches)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>2.6</td>
</tr>
<tr>
<td>3.75</td>
<td>2.4</td>
</tr>
<tr>
<td>7.50</td>
<td></td>
</tr>
<tr>
<td>11.25</td>
<td></td>
</tr>
<tr>
<td>15.0</td>
<td></td>
</tr>
<tr>
<td>18.75</td>
<td></td>
</tr>
<tr>
<td>180</td>
<td>4.78</td>
</tr>
</tbody>
</table>

Figure. 14. PMS® data structure
5.2.2. Comparison with the SICK

The PMS® is similar to the SICK® in most ways. The scanning range is the same for both these scanners, except for the fact that the maximum scanning range of the SICK® is 25’ and for the PMS® it is 25”. This also coincided with the scaling factor of the model (1/10th scale). As the test bed at the laboratories was all scaled to 1/10th of the original dimension, the maximum range of 25” on the PMS’s seemed a very logical range. The scanning rate is also very much different for both these scanners. The following table illustrates the comparison between the PMS® and the SICK® laser scanners.

<table>
<thead>
<tr>
<th>Function</th>
<th>SICK</th>
<th>PMS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum scanning range</td>
<td>25’</td>
<td>25”</td>
</tr>
<tr>
<td>Time for one full scan</td>
<td>0.5 sec</td>
<td>3 sec</td>
</tr>
<tr>
<td>Scanning range</td>
<td>180 degrees</td>
<td>180 degrees</td>
</tr>
<tr>
<td>Data structure</td>
<td>Bi-dimensional array</td>
<td>Bi-dimensional array</td>
</tr>
<tr>
<td>Data length/scan</td>
<td>181</td>
<td>49</td>
</tr>
<tr>
<td>Scan angle increment</td>
<td>1 or 0.5 degrees</td>
<td>3.75 degrees</td>
</tr>
<tr>
<td>Control Interface</td>
<td>In-built control cards</td>
<td>M68HC11</td>
</tr>
<tr>
<td>Data transfer interface</td>
<td>RS232/RS422 serial transfer</td>
<td>RS232 serial transfer</td>
</tr>
<tr>
<td>Power</td>
<td>24 VDC</td>
<td>12 VDC</td>
</tr>
</tbody>
</table>

5.2.3. PMS Line-finding

The reduced data size of the output data from the PMS® directly reflected on the processing of the line-finding routine. As the original routine was designed for the SICK® data (181 points), the same routine was not appropriate for use on the PMS®. The application of the line-finding routine for the PMS® data required that changes be made in the line finding routine. The major changes that were made to effect this transfer are the change in the number of data points. Moreover, as the PMS’s were set to read from a different start angle (other than 0-180),
changes were made to process this modified angle inputs. Appendix III (iii) describes this modified code in detail.
CHAPTER 6: CONTROL SOFTWARE DEVELOPMENT

This chapter introduces the interface software and the various stages of software development.

6.1. CIN Development

To enable communication between the algorithms and the prototype MBC hardware, it was desired to employ a software interface that will have the capability to link the control logic with the physical system. After exploring various existing interfaces techniques, the use of a commercially available code or program seemed appropriate than to spend time writing dedicated code.

LabVIEW®️, a commercially available data acquisition software from National Instruments®️ was chosen to be the main hardware interface. LabVIEW®️'s capabilities were studied in comparison to other existing automation packages and as a collective effort from the team, LabVIEW®️ was chosen for the interface with the prototype hardware. Moreover as LabVIEW®️ has software libraries that allowed the integration of custom code to be called during execution, it acted as a quick prototyping tool for this research enabling the testing of the control algorithms without involving high levels of complexity. This research discusses only the code module of LabVIEW®️, (i.e.) only the CIN and not the software itself. More details on the use of LabVIEW®️ for this automation effort can be found in the thesis document of Bruce Wells (3).

6.1.1. The Code Interface Node (CIN) interface

CIN is the acronym for “Code Interface Node”, a G language (a Graphics based programming language) routine in LabVIEW®️ that allows the use of external functions or programs (like C, Pascal, Fortran) to be associated with LabVIEW®️. The use of this methodology introduced an additional flexibility to the user in terms of using high-level language routines to be directly interfaced with LabVIEW®️, thus offloading most of the
The use of LabVIEW® for rapid development of mobile robot prototypes has been illustrated by the author and Dr. Sturges in their paper titled ‘A Hierarchical Real Time Control System for Rapid Development of Mobile Robot Prototypes’. In the development of control and interface software for the CHS, LabVIEW® was used as the low level interface (between the hardware and the host system) whereas, all the algorithms and other track commanding routines are processed through custom C code interfaced with LabVIEW® through the CIN.

The screen snapshot in Figure 15, shows a typical LabVIEW® VI frame with the code interface nodes, wired to various inputs and outputs. The CIN is enclosed in the highlighted area.

Figure. 15. A typical Virtual Instrument block diagram showing the Code Interface Node
The CIN allows the embedding of pre-compiled custom high level language code in LabVIEW®. As the author and the team were very comfortable with programming in C, and as C was considered as the foundation for future developments, the LAVT team chose to use C to embed the object codes into the CIN for data processing. The C code, essentially the line finding and the motion control algorithms were compiled using a standard Makefile that included a special National Instruments® header file. The programming interface used to create the pre-compiled code resource was MS Visual C++ 6.0, and this Makefile on compilation would result in an ‘.lsb’ file that can then be loaded in the CIN.

6.1.2. CIN Inputs and Outputs

The CIN accepts different forms of inputs and process them in different ways. Though the base compilation file is a ‘C’ file, the programming environment requires an external LabVIEW® header called as ‘extcode.h’. This header file, supplied by National Instruments® supports the processing of the CIN inputs and other CIN elements. The LabVIEW® Code Interface Manual provides a detailed description of the various types of inputs that the CIN can process. The source code manual included with this research document also provides a detailed documentation of the CIN inputs and its functioning. Typical CIN inputs and outputs used in the CHS automation algorithms include variables of types ‘int’, ‘float’, ‘bool’, ‘array’ and ‘clusters’. The corresponding type conversions for these variables are as follows.

<table>
<thead>
<tr>
<th>C Style variables</th>
<th>LabVIEW – CIN variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>Int</td>
<td>Int32</td>
</tr>
<tr>
<td>Float</td>
<td>Float64</td>
</tr>
<tr>
<td>Bool</td>
<td>LVBoolean</td>
</tr>
</tbody>
</table>
| Array[10]         | Typedef struct {
                     Int32 dimSizes[1];
                     Float64 arg[1];
                 }array; |
The following Figure illustrates the various types of input variables being wired into the CIN and the outputs. Labels are provided to identify each of the above-discussed types.

![Diagram of CIN with inputs and outputs]

Figure. 16. LabVIEW® CIN with inputs and outputs

6.1.3. Functioning of the CIN

A CIN is a block diagram node associated with a Section of source code written in C. The source code is first compiled and linked to form an external executable code. LabVIEW® calls this executable code when the node executes, passing input data from the block diagram to the executable code, and returning data from the executable code to the block diagram.

‘CIN’s execute synchronously, which means that LabVIEW cannot use the execution thread used by the CIN for any other purpose. When the VI executes, LabVIEW monitors the menus and keyboard actions. When running multi-threaded, there is a separate thread for each of these tasks. Thus, when a CIN object code executes, it takes control of its execution thread. If LabVIEW has only one thread of control, then all of the LabVIEW is stopped until the CIN
object code returns back to the block diagram. In multi-threaded environments like Win98/NT, CIN’s do not stop the other actions of the VI while execution’ (quote reference – excerpts from CIN manual)

While a multi-threaded application can control other actions via the free threads even while the CIN object code is running, it must be noted that the VI cannot be reset until the CIN is complete.

The Makefile

The following is the template for the CIN Makefile.

```makefile
name = 'name_of_c_file'
type = CIN
CINTOOLSDIR = path to LabVIEW CINTOOLS directory
!include $(CINTOOLSDIR)\ntlvsb.mak
```

The ‘name’ tag corresponds to the name of the C file – the source file that will be loaded in the CIN. The type may be either CIN or an external subroutine. The discussion of the external sub routine is beyond the scope of this research. The third line in the Makefile corresponds to the path where the LabVIEW \Cintools directory is located and the last line supports the compilation of the source code via a link to the standard WIN OS Makefile – ntlvsb.mak.

Thread safe and Multi-threading

The CHS CIN makes use of the multi-threading capabilities of the CIN and when defined with the standard CIN C file, will ensure in a multi-threaded application. Multi-threading transfers control to LabVIEW via other free threads to control other synchronous routines. Multi-threading corresponds to defining the VI as “Re-entrant”.

The procedure for a Re-entrant CIN object code is discussed in the next Section.
**Error handling and Re-setting**

Most source code development environments has inbuilt error trapping routines or a debugging program that will enable program step through for debugging operations. Though the LabVIEW® CIN routines are compiled using the MS® VC++ compiler, the MS® debugger could not be used due to the external linking of the source file. It should be noted that the CIN source file cannot be compiled as a standalone file and the user does not have the flexibility of stepping through the program.

To circumvent this problem, LabVIEW® allows the use of a ‘debug’ statement that will force the CIN source code to temporarily stop execution and display the current value onto the front panel. The CIN source codes developed for the CHS automation do not make use of this feature, but is explained here as a future task. Apart from the debug statement, LabVIEW also allows the use of standard error handling functions. Error traps found in the standard C++ library – the Try, Catch and Throw routines can be used in the CIN to process errors. Moreover, every CIN Manager function returns a NULL value (NoErr) to the block diagram. If this value is not returned, it depicts the presence of an error in the source code and the execution of the CIN halts.

When the CIN execution halts, the front panel becomes inactive and the user has no control over the execution of the VI. This stage is called as the “Re-setting” stage. Various causes are attributed to this error in execution. One of the most common errors, which cause re-setting of the VI, is error in the source code either in the form of invalid pointers or infinite loops. Common cases are when a conditional loop never executes or infinite re-cycling occurs. During the development of algorithms for the line finding, invalid pointers and numerical errors have caused this error. Hence, it is the complete responsibility of the programmer to ensure that the source code is bug free, and if at any point of time the user encounters this error, the only way to stop the VI is to free the current thread.

Freeing the current thread is beyond the user’s capability to manipulate in an operating system, and hence the best way to restart the VI is to use ctrl-alt-del keys in combination to close LabVIEW® with improper referencing. Though this is not a suggested method to do as it might
result in data loss, there are no other error traps that can handle this error. Proper commenting of code and simulating the logic in another form can avoid the occurrence of this error.

**Steps to create a CIN and load its code resource**

The following steps illustrate the procedure for creating a simple LabVIEW CIN. The example illustrated here performs the addition of two numbers ‘a’ & ‘b’ and returns the result ‘c = a+b’ to the block diagram, and finally this result is displayed on the front panel. Screen shots of the process are included for additional clarity.

**Creating the CIN block diagram**

1. Open LabVIEW® and select ‘New’ from the File menu
2. Choose the ‘Advanced’ tab from the ‘functions’ menu with the block diagram currently active. Block diagram can be activated by using Ctrl + E keys.
3. Press and hold the ‘Advanced’ tab and choose the ‘Code Interface Node’ function from the list of different sub-tab icons.
4. This will now activate the CIN. Place the CIN at any preferred location in the block diagram.
5. The top portion of the CIN will be colored in orange, depicting that the CIN is devoid of its re-entrant source code. Careful observation will reveal that the CIN, below its orange header, has two columns with one row. The left column corresponds to the inputs and the right corresponds to the outputs. Figureure 17 illustrates a default CIN.
6. This illustrates that the CIN has neither inputs nor outputs wired to it. The default is only one row allowing either only one input or output.
7. Assuming that the controls for the two input numbers ‘a’ and ‘b’ and an indicator for the result ‘c’ are already present in the block diagram, right clicking the mouse on the first column of the CIN will reveal a pop-up menu with menu options. Choose ‘Add Parameter’ and add one more two-column row to the CIN. This is an additional input-output row for the CIN. Repeating this step will add one more input-output row. The CIN will now look like in the illustration in Figure 18.
Figure. 17. A default Code Interface Node with no object code

Figure. 18. CIN with 3 input-output rows

8. As the CIN has to accept two inputs and send one output, the input-output row has to be configured accordingly. Wire the ‘a’ and ‘b’ control terminals to the first two
columns of the CIN (on the left hand side). Wire the indicator ‘c’ to the last column. It must be noted that the indicator ‘c’ is an output, and the CIN has to be configured accordingly. Right clicking on this column will reveal a pop-up menu with the sub-menu option ‘Output Only’. Choosing this option will blank out the left cell of this row and this procedure will configure the last row to be an output row. Complete the wiring of the CIN by wiring the last output. The CIN will now look like the illustration in Figure 19.

![Image of a CIN with complete inputs and outputs wired]

Figure 19. A CIN with complete inputs and outputs wired

9. The next stage is the creation and loading of the source code for the CIN.

**Creating and loading the CIN source code**

The following procedure may be followed to create the CIN source file and to load the CIN with its pre-compiled code resource.

1. After complete wiring of the CIN, right-clicking on the CIN and choosing the sub-menu ‘Create C file’ will now open a new dialog box with the title ‘Choose a code
resource.c file’. Type in a file name, say ‘add.c’ for the source file name in the dialog box.

2. Create a Makefile as follows using a standard text editor (preferably the DOS edit) and save the file as ‘add.mak’.

```plaintext
name = add
type = CIN
CINTOOLSDIR = c:\nation~1\LabVIEW\Cintools
!include $(CINTOOLSDIR)ntlvsb.mak
```

Note: Ensure that the two files ‘extcode.h’ and the ntlvsb.mak files are available in the directory of the development environment. In MS VC++ 6.0, the extcode.h file has to be copied to the ~\include\directory.

3. Open this Makefile by clicking File-Open in the Microsoft Visual C++ 6.0 compiler.
4. Choose ‘Yes’ to the dialog message and create a Workspace by name “Add1”.
5. Open the created C file and this file will look similar to the following.

```c
/*
* CIN source file
*/
#include “extcode.h”
CIN MgErr CINRun(float64 *a, float64 *b, float64 *c);  
CIN MgErr CINRun(float64 *a, float64 *b, float64 *c) {  
    /*ENTER YOUR CODE HERE*/
    return noErr;
}
```

6. This is the template of the resource file. It can be observed that all the variables, both input and outputs are being passed as pointers rather than values. This is due to the fact that LabVIEW CIN uses the ‘Pass by reference’ technique.

7. To compute the addition of the two numbers and store the result in the output variable ‘c’, add the following lines of code to the existing template. The code will now look as follows..
A Re-entrant CIN

It is required that in developing the code resources for the CIN, that the CIN be re-entrant. This is an indirect error-handling scheme. If an error occurs while processing the object code, the CIN will halt execution and Re-enter once this error has been handled. Though error handling is not discussed in detail in this document, modifying the CIN to be re-entrant has other potential advantages. More documentation on this can be found in the LabVIEW® Code Interface Manual. Creating a re-entrant CIN requires the addition of a new function to the code.
and loading this re-modified code will change the ‘Orange’ color of the CIN header to ‘Yellow’. For example, the re-entrant version of the add.c file appear as follows…

```c
/*
 * CIN source file
 */
#include "extcode.h"
CIN MgErr CINRun(float64 *a, float64 *b, float64 *c);
CIN MgErr CINProperties(int32 prop, void *data);
CIN MgErr CINProperties(int32 prop, void *data) {
    switch(prop) {
        case kCINIsReentrant :
            (Bool32*)data = TRUE;
            return noErr;
        }
    return mgNotSupported;
}
CIN MgErr CINRun(float64 *a, float64 *b, float64 *c) {
    *c = *a + *b;
    return noErr;
}
```

6.2. LabVIEW VI’s developed for the CHS automation

The following Sections describe the CIN frame of the LabVIEW® VI’s developed for the CHS automation. Detailed documentation on creation of these VI’s can be found in the documentation provided by LabVIEW®. This research will concentrate only on the Code Interface Node frame and will illustrate changes in this frame during the various phases of development and modification. The code resources for all these nodes are not discussed here, as they are dealt in great detail in the next chapter. Though the control interface software underwent
long periods of development and refinement, only the final developed control VI is discussed here to avoid repeatability.

6.2.1. Single MBC with the SICK scanner

This VI was initially developed to control a single MBC with analog inputs (the two angle sensor and one dolly sensor). The sensor used for guidance and navigation is the SICK® laser scanner and the simple line-finding algorithm was implemented using this VI. The VI processed the data from the scanners and output this data to the CIN to generate the velocity outputs. The screen shot of the Code Interface Node Block Diagram is shown below in Figure 20.

![Figure 20. CIN frame for a Single MBC control](image)

As it can be observed from the Figure above, the two main inputs to the CIN are the data from the left and right SICK® scanners and the start and end angles for the line-finding restriction. The VI uses two CINs, with the first node processing the line finding on the SICK data and the second node performing the motion control strategy to navigate the MBC. The
outputs of the motion control CIN node are the left and right track speeds along with the MBC’s position and orientation.

The VI was implemented on the 1/10\textsuperscript{th} scale prototype and proved to be successful in producing confident results. The driving scenario for the test portrayed a simple 90-degree mine plan with a U turn. The MBC was programmed to start at one corner of the U turn, navigate around this bend to the roadway, and progress down the other bend before it finally stops. The algorithm’s robustness was verified and this VI was re-configured for full-scale MBC testing, which again yielded results to the complete satisfaction of the LAVT.

6.2.2. PMS control through the interface

The initiation of the proposal to develop the PMS\textsuperscript{®} for use on the prototype demanded modification of the LabVIEW\textsuperscript{®} VI. The LabVIEW VI discussed in Section 6.2.1 that used the SICK\textsuperscript{®} as the navigation sensor was required to be modified for use with the PMS. Though the control of the SICK\textsuperscript{®} laser scanners appeared to be complex, the control of its successor, the PMS\textsuperscript{®} was relatively an easy task. To initiate a scan sequence to the left and right PMS’s the control software was required to send an ‘initialization’ string to the controller onboard the PMS\textsuperscript{®}. Lu Yufeng and Bruce Wells developed the assembly codes for the PMS controller.

In discussion with the developers of the PMS\textsuperscript{®}, it was agreed to send an initialization string “06h” to the PMS controller. This string would command the MBC controller to initiate a sensor scan sequence. The controller on receipt of this initialization string would initiate a scan sequence on the left and right scanners and would send the data back to LabVIEW\textsuperscript{®}. It must be noted that the PMS VI uses only 1 serial RS232 port for communication unlike the SICK\textsuperscript{®} control interface that requires dedicated RS232 ports for data communication from the two SICK\textsuperscript{®} scanners. No other changes other than the addition of a ‘scan request’, was required for the PMS\textsuperscript{®}. Changes were also made in the object codes to handle the reduced data set from the PMS laser scanners.
6.2.3. Multi-MBC VI with PC-PC data sharing

When the tests on a single MBC and a Dual MBC system with ‘Follow the leader’ techniques proved to be satisfactory, development was more directed towards a Multi-unit system. Multi-unit refers to more than 2 automatic units that follow a manually driven lead MBC. More details about the different algorithms for a multi-unit system is described in Chapter 7. In all the previous developments, the trailing MBC - the automatic unit followed the manually driven lead MBC and this trailing unit was controlled through a dedicated host PC running the interface and the object code. To meet the control needs of a multi-unit system (3 units - 2 automatic units with a manual leader unit), the requirement of an additional PC was recognized, and hence in this new configuration, each automatic unit was designed to be controlled by an individual host controller.

A LabVIEW® VI to meet this new requirement was developed by the author as an extension to the base VI’s developed earlier. Moreover, as in this stage of development, the SICK® scanners were no longer being used on the 1/10th model, the VI’s used the PMS® scanners for navigation. The new VI was designed to meet the requirements of a 3 unit system and the two rear units were controlled by the LabVIEW VI’s from two individual host PC’s.

In this configuration, achieving automated control was a more complex task. In addition to each host PC controlling the MBC, it was identified that the MBC data would not suffice to locate and orient a unit with respect to its peer units. This was not an issue in a 2-unit system, as the lead unit was manual and the rear unit just performed the task of following the lead unit. Whereas in the configuration described above, the two automated units had to coordinate with each other in addition to the task of following the manually driven lead unit. This task of coordinating the 2 automatic units demanded some level of interaction between these units. Each of these units required data from its peer units for self-localization with respect to each other. This triggered the development of the concept of ‘data sharing’.

This need for data sharing was substantiated by the DMM simulations, which clearly identified the need for inter-MBC communication. Hence, the author spent a major portion of
time developing this new VI. The VI was designed with more capabilities and functions than actually required for a multi-unit control. The motivation to add redundancy was that the time and effort involved in this extensive development would finally lead to the developments of control algorithms, and this latter stage of development would require a robust and flexible user interface.

Some of the key functions performed by this VI are as follows…

1. Control of dual PMS laser scanners.
2. Data acquisition and dynamic scanner data plotting.
3. Analog data acquisition from the angle sensors and the sliding dolly sensor.
4. MBC Configuration selector – allows the selection of the MBC type for which the VI will be used – First, Last or an Intermediate system. This selection was vital due to the change in sensor configurations in the first and last units. The first unit lacked a sliding sensor, and the last unit had an additional RFM angle sensor. The MBC configuration selector allowed the use of the same VI for all these systems.
5. Read data routine - Request for data from peer units (*).
7. A dual CIN frame with inter-MBC data to process the line finding and the motion control object codes.
8. A closed loop PID frame for PID control of the tracks.
9. Send data routine – A serial data transfer routine that allowed data communication from current MBC to its peer units (*).

* These functions employ a Sub-VI.

**PC – PC serial Data Transfer**

The PC-PC data transfer function for MBC data sharing was performed through a RS422 communications port on the host PC’s. A National Instruments® dual RS422 PCMCIA cards were used for this function. These two additional ports on the host PC performed just the function of transferring data. One port was designated as ‘Configure UP’ port and the other one as ‘Configure DOWN’. The former was designated to communicate (both send and receive) data
to all the units in-by the current unit and the latter port was designed to communicate data to the units trailing the current unit.

Due to the similar port configuration on the host PC’s, the send and receive lines were of the same category (both acting as a DCE – Data Communications Equipment), and hence direct cable communication was not possible. For a direct port-port communication to be enabled, one of the ports must be configured as a DTE (Data Terminal Equipment) or a NULL MODEM cable has to used. The option of building a NULL MODEM cable seemed an easier and logical choice and NULL MODEM cables were designed for this purpose. The port pin-out diagram and the Null Modem specifications are included in Appendix II (iv).

**The Sub VI’s**

The read data routine and the send data routines employed two custom built Sub VI’s to perform the task of receiving and sending serial data. A Sub VI is a LabVIEW® VI that can be included in the block diagram and can be called like a function, thus establishing a hierarchical link. The calling VI is termed the parent and the Sub VI’s are termed child VI’s.

![Sub-VI block process for serial data communication](image)

Figure. 21. Sub-VI block process for serial data communication

The function of these two Sub VI’s is to convert the floating-point data to a hexadecimal string (with sign bytes) when sending data and to convert the incoming hexadecimal string to a floating-point value. Each floating-point value is represented by a 5-byte packet. The first byte is a sign byte followed by 4 data bytes. If the sign byte is ‘F’, then the value is interpreted as a
negative value, else if the sign byte is ‘0’, the value is a positive value as show in Figure 21. The block diagram of the send_data and rec_data Sub VI’s are included in Appendix II (iii).

**The CIN Object code handling**

This Section briefly describes the object code handling of the various types of inputs wired to the CIN. The multi-unit VI uses two Code Interface Nodes. This Section discusses only the CIN that processes the line-finding object code to avoid repetitive texts due to the reason that both these nodes process the same types of inputs.

The discussion is directed on the handling of different input types and not the object code itself. Chapter 7 is dedicated to the CIN object codes. The CIN processes the following types of inputs…

i. A double dimensional floating point array – for the scanner data.

ii. Single valued floating point inputs.

iii. A cluster input – converted to a single dimensional array – For the inter-MBC data.

iv. Boolean variables for the TRUE/FALSE variables.

**Example Object codes**

The object codes for each of the above type of inputs are presented below.

**i. A double dimensional floating point array**

Input variable name : var1  
Type : length X 2 floating-point array  
C style : float var1[size_1][]

LabVIEW CIN object code :
typedef struct
{
    int32 dimSizes[2];     // rows X cols -- > rows = len
    float64 arg1[1];   // col 1 = angle, col 2 = range
} TD1;

typedef  TD1 **TD1Hdl;  // Create a 'handle' to the data structure
TD1Hdl PMS_data;       // Temporary Variable assignment
PMS_data = var1;       // Incoming variable assignment

Data extraction :

    N_rows = (*PMS_data)->dimSizes[0]; //The number of rows in the array – depth
    N_cols  = (*PMS_data)->dimSizes[1]; //The number of columns = 2 (as bi-dim)

    // To access the element in the ith row and jth column of a bi-dimensional array
    // with N_cols number of columns, the following statement has to be used.

    Elemnt = (*PMS_data)->arg1[i*N_cols + j];

ii. Single valued floating point inputs

Input variable name   : cutoff
Type                  : floating point
C style               : float cutoff

LabVIEW CIN object code  :

    Float64 temp;        // Temporary Variable assignment
    temp = *cutoff;      // Incoming variable assignment

Data extraction :

    value = temp;
iii. A single dimensional array

Input variable name : var2
Type : length X 1 floating-point array
C style : float var1[size_1]

LabVIEW CIN object code :

```
typedef struct {
    int32 dimSize;      // no of rows
    float64 arg1[1];   // no of data columns = 1
} TD2;

typedef TD2 **TD2Hdl;   // Create a 'handle' to the data structure
TD2Hdl MBC_data;        // Temporary Variable assignment
MBC_data = var2;        // Incoming variable assignment
```

Data extraction :

```
N_rows = (*MBC_data)->dimSize;  // The number of rows in the array – depth

// To access the element in the ith row.....

Elemnt = (*MBC_data)->arg1[i];
```

iv. Boolean inputs

Input variable name : Inter_MBC
Type : Boolean
C style : bool Inter_MBC

LabVIEW CIN object code :

```
LVBoolean temp;       // Temporary Variable assignment
temp = *Inter_MBC;    // Incoming variable assignment
```
Data extraction:

\[
\text{value} = \text{temp};
\]

**Handling multiple data sharing VI’s on multiple host PC’s**

The control VI for a multi-unit system was developed with the motive of running this VI on multiple host PC’s at the same time for the control of different MBC units. In a two-unit bench test configuration (both automatic units), the issue of data clogging between the PC transfer ports halted the execution of the VI. The data clogging was due to the fact that both the host PC’s were running the same interface VI and when data read/write was activated on both the control interfaces, the two PC’s would poll the serial ports simultaneously expecting data from each other. This simultaneous serial port polling interfered with the further execution of the VI as the ports would stall waiting for the serial data.

To solve this problem, the concept of ‘data time-out’ was introduced in the read/write routines. The goal of this addition was to set a time-out value for the host PC’s so that they would poll the serial port for data within the time-out value, and if the time-out is exceeded, normal execution of the VI would continue with no incoming data. When different time-out values are used on both the PC’s, there will be no simultaneous polling of serial port data and hence, the data clog would be removed. Thus the goal was to set serial time-outs for the different host PC’s to avoid this problem. It should be noted that with this introduction, the first execution of the VI will always be devoid of incoming data, as in the first occurrence, the VI’s would timeout at different instants on both the PC’s and from the second execution onwards, due to the in-built time difference, data transfer will be enabled. The modified VI was tested and yielded successful results on a 3-unit system with two host PC’s for each of the two automatic units.
CHAPTER 7: CONTROL ALGORITHMS

The previous chapters discussed the interface software development, the CIN object code structure, the SICK® and PMS® scanners, all with respect to the data structure and the interfacing aspects. This chapter will discuss the various algorithms that were embedded in the interface software via the techniques previously described. These algorithms are termed ‘Control Algorithms’ due to their function of controlling a MBC.

The automation of the CHS was based on the theory of mimicking the human operator’s function in driving the MBC, so that there would exist an initial platform to start the algorithm development. The algorithms were based on the classical mobile robot principles and aimed at answering the 3 questions Viz.

Where am I? – The current location/orientation of the MBC with reference to a mine wall.

Where should I be? – The position/orientation computed by the algorithm (based on the initial path developed).

How do I get there? – The motion control techniques required for commanding track velocities to trace the computed path.

Every autonomous mobile robot requires the function of environment perception or sensory capabilities that will allow it to measure its working environment. This is the “Where Am I” part of the algorithms. This aims at computing the system’s current location (both position and orientation), at some time instant ‘t’ based on the input sensory data with reference to a fixed or moving coordinate frame. This process is usually termed “Self Localization”.

Once self-localization is established, the next stage involves the computation of the location in the environment where the system should be located at some time instant, say ‘t + δt’. Different values for the increment parameter ‘δ’ may be used, but smaller the values are, higher the resolution of the computation. This process computes the location of the system in the future,
and essentially computes a path for the system to travel to. This is the “Where Should I be” part of the algorithms.

Once the path has been computed, the final stage involves the direction of the vehicle’s control system to position the vehicle on the computed path. This is the “How Do I get there?” stage in the algorithm process.

The line-finding provides answers to the ‘Self Localization’ problem by computing the current location & orientation of the MBC with reference to a mine wall. The ‘path planner’ then computes the target distance and orientation for the next move (along the path already computed) and the velocity commanding routine (the Simple Motion control and the SSS algorithm) performs the function of motion control.

7.1. “Where Am I” – MBC Localization

7.1.1. MBC localization using UT sensors

The initial developments for the guidance and navigation algorithm began with the use of Ultrasonic sensors for navigation. Each MBC was designed to have 4 UT sensors, two on each side for mine wall detection. The initial developments were targeted towards using the single point UT sensor for localization and was followed by the implementation of the logic via the CIN object code. Though the 1/10th scale prototype was never tested with this UT localization algorithm for various reasons pertaining to reduced data set and accuracy issues, this acted as the base technology for the algorithm development for the SICK and the PMS laser scanners.

Since the UT sensors were a single point range measuring device, there were only two data points on each side of the MBC. Initially, only one side of the sensor data from a MBC was considered for implementation as this was sufficient to locate the MBC with reference to the mine walls. The assumption behind neglecting the data from the other side was to maintain simplicity and was also motivated by the fact that the initial tests were targeted only on a
standard 90-degree mine plan. Hence, the mine parameters were all fixed, and there existed no requirement for the inclusion of mine plan variability.

Thus, with only two data points on one side of the MBC, it was direct and easy to apply a very simple localization technique. Figure 22 illustrates the UT sensor data from a MBC in a simple mine environment.

![Figure 22. MBC Localization with UT sensors](image)

From the Figure it is evident that localization is just based on the two distances ‘r1’ and ‘r2’, and this is sufficient to compute the location (x,y) and the orientation (θ) of the MBC with reference to the walls.

The following equations describe the Self Localization for this system...

\[
\theta = \sin^{-1}\left(\frac{r_1 - r_2}{l_{mbc}}\right)
\]

where,
\( r_1 = \) range measured by the lead UT sensor (inches)
\( r_2 = \) range measured by the trailing UT sensor (inches)
\( l_{mbc} = \) length of the MBC (inches) (this is assuming that the sensors are placed at extreme ends of the MBC)

and the distance \( D \) is given by

\[
D = r_1 - \sin(\theta) \times \frac{l_{mbc}}{2} \quad \text{(assuming that } r_1 > r_2) \]

where,
\( D = \) Distance from the center of the MBC to the mine wall (inches)

Note: It is assumed that the distance of separation between the sensors is the length of the MBC \( (l_{mbc}) \) (i.e.), that the sensors are placed at the extreme end of the MBC. Moreover, the computation of \( D \) does not take into account the actual width of the MBC.

Since this self-localization algorithm involving the UT sensors was not tested on the 1/10\(^{th}\) scale prototype, no further developments were directed towards improvising the technique. The major reason for not applying this technology was its inappropriateness to the varying and harsh mine conditions and its reduced data set combined with inaccuracies in range measurement. Moreover, the data from the sensors were a single point data and hence, any small change in the MBC orientation would reflect to a very large amount on the working of algorithm. And as the algorithm was entirely dependant on this single point data, the effect of using this technology on the system would result in heavy oscillations and this remained a convincing fact to reject this technology.

### 7.1.2. Line-finding

The absence of satisfactory results from the localization routine developed for the UT sensors prompted the need for a more robust technique for position estimation. The thought now was to use the full sensory data rather than just use 2 data points to self localize. Thus, the UT’s were replaced by more advanced sweeping laser range finders – the SICK\(^{\text{®}}\) and then the PMS\(^{\text{®}}\). These sensors returned a complete array of data points unlike the single range data from the UT
The proposal for the use of the sweeping range finders for environment perception gave birth to the term ‘Line-finding’. This idea originally was conceived by the author, Amnart Kanarat and Bruce Wells. The aim was to develop a unique and easy way to represent the mine wall data for it to be used with the self-localization routine. After a thorough review of the mine plan patterns supplied in the Long-Airdox documents, a commonality between all these mine plans could be observed. All the plans were prismatic, except for the varying radii fillet around the corners and from a plan view, the mine plan represented an array of repeating patterns. The MBC operators would drive between the gaps in these repeating patterns, called the roadway or aisle way, thereby navigating the system along for every cut sequence. This driving strategy combined with the mine plan structure initiated the idea that when a MBC drives in a roadway, all that is visible are the two mine walls on either side and the vision of the corners when the system approaches a left or a right turn.

Thus it is both logical and intuitive to identify the mine walls by a straight line, and use this straight line to locate the MBC, and hence, the term line-finding came into existence. The theory was to interpret the sensor data as a sequence of points, and use these points to compute a straight line that would represent the mine environment in the MBC’s current location. The method of least squares for fitting a line was applied to the incoming sensor data and an equation of the line in the form ‘Y = aX + b’ was computed. This line was computed with reference to a fixed coordinate frame on the MBC – The MBC’s dimensional center.

Two lines for each MBC (one for each scanner) were computed and the position and orientation of the MBC was computed using this line data. The system of equations below Figure 23, illustrate the appropriateness of line-finding to solve the problem of self-localization. It must be noted that the line-finding technique is a generic term, and can be applied to both the UT and the sweeping laser scanners.
Let \( \theta \) be the angle of inclination of the MBC with respect to the computed line, and let ‘D’ represent the distance of the MBC’s physical center from this line.

If \( P(\theta, r) \) are the data points returned by the scanner, then these points are converted to Cartesian form by the following set of equations. It should be noted that the frame of reference is on the scanner itself.

Thus, the Cartesian form of the scanner data points is given by \( P(x_i, y_i) \), where….

\[
\begin{align*}
x_i &= r_i \cdot \cos(\theta_i) \\
y_i &= r_i \cdot \sin(\theta_i)
\end{align*}
\]

From these series of points, the method of least squares is applied to arrive at the equation of the line \( L \).
If \( L = aX + b \), then the values of ‘a’ and ‘b’ (the slope and the intercept) are given by…

\[
a = \frac{n \sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} y_i}{n \sum_{i=1}^{n} x_i^2 - (\sum_{i=1}^{n} x_i)^2}
\]

\[
b = \frac{1}{n} \left[ \sum_{i=1}^{n} y_i - a \sum_{i=1}^{n} x_i \right]
\]

Thus, we have the line \( L = aX + b \).

Now, if ‘\( x \)’ and ‘\( y \)’ represent the X and Y coordinates of the MBC’s physical center with reference to the line \( L = aX + b \), then

\[
x = \frac{-b \ast a}{a^2 + 1}
\]

\[
y = \frac{b}{a^2 + 1}
\]

The position and orientation of the MBC with reference to the mine wall are…

\[
\theta = \tan^{-1}(a)
\]

\[
D = \sqrt{x^2 + y^2}
\]

Once the position and orientation of an MBC is computed with reference to its environment, the task of ‘Self Localization’ has been completed. Thus, line-finding was confirmed as the simple and effective means of self localization for the MBC.
7.1.3. Line-finding applied to the Laser scanners (SICK LMS 200 & PMS)

When confidence in the above-developed line-finding routine was established, the stage following this was the implementation of this methodology for the Laser scanners. As both the SICK and the PMS are sweeping range finders, example of the SICK is considered for explanation. The only change in the line-finding routine for the PMS is the reduced data set (49 as compared to the 181 from SICK). This application is termed as ‘Simple line-finding’.

Applying the line-finding to the SICK data was very realistic and simple. Standalone data samples from multiple single scans were analyzed in MS Excel and the line-finding yielded satisfactory results. The line-finding initially applied to the scanners was a very simple and easy technique. To begin with, two parameters controlled the quality of the line produced and its accuracy in locating the MBC. These parameters were two extreme values for the sweeping angle. When a range for the sweeping angle is specified, this indirectly creates a window from which the data is used. Hence, all the data points from a full scan were not used. The line-finding would use the data from the window created by specifying an upper and lower bound for the angle. Figure 24, below illustrates this technique.

Figure 24. Simple line-finding applied to the SICK data
On experimentation, the range from 70 degrees – 140 degrees seemed satisfactory for both the prototype and field tests on a full scale MBC (in a 90 degree mine plan). Though there is no hard theory to explain these ranges, the window created was fairly wide, and the algorithm development was simplified as there was no requirement to isolate sensory noise or external objects present over the full range of 180 degrees. Field tests of this algorithm yielded successful results and the MBC was able to navigate autonomously a U curve, in both forward and backward directions. The target was to keep the MBC at a specified distance from the wall and maintain this distance throughout the course of the curve. Though not a classic ‘path planning’, the algorithm achieved the results of a path planner due to the effectiveness of the line-finding routine.

### 7.1.4. Iterative end point line splitting

Though the simple line-finding by creating a usable window yielded very successful and satisfactory results, there were some extreme cases, which the developers had overlooked while arriving at the algorithm. One of the major glitches was the presence of the PIG while negotiating a corner. To expand on this, if the window is a constant (a fixed start and end angle), while the MBC negotiates a sharp turn, the in-by PIG negotiates the curve much ahead of the following MBC and thus, the window created would include a Section of the PIG and the line-finding would still treat this as a mine wall and come up with a false representation. This problem did not arise during the previous field tests as all the tests were conducted on a single MBC only. Thus, when the system was extended to a multi-unit system, the presence of the PIG required that the algorithm be modified.

Review of literature on mobile robotics and sensor-based navigation exposed a variety of methods, but not all of them seemed appropriate for the application to the CHS. One technique described by P. Skrzypczynski (14), in his paper titled ‘2D and 3D World Modeling Using Optical Scanner Data’, at the Technical University of Control, Poland employs a new technique termed “Iterative End point line splitting”. This new technique was aimed at developing a 2D environment map for a mobile robot using laser range finder data and this seemed a very close
match to the research on the CHS. Detailed exploration of this technique revealed its appropriateness for use on the CHS.

This new technique was again based on the original-line-finding methodology, but used the entire range of the scanner data without the interference of the PIG’s. Thus it is evident that a dynamically changing window was created and the values for the start and end angles were governed by the PIG angles, both at the in-by and the out-by end of the MBC. Thus, the scanner data used by the algorithm would vary for every scan based on the dynamic window, which in turn was a function of the PIG angles. Angle sensors at the PIG joints were used to measure this angle.

Amnart Kanarat performed the implementation of the iterative end point line splitting. Though the author did not extensively take part in implementing this technique due to the completion of his course work, a brief overview of the method is presented to introduce the methodology. The source code listing is provided in Appendix III (iii).

The iterative end point line splitting, as the name suggests aims at recursively splitting a single line into sub groups. ‘Once the scanned data is received from the scanner, the method creates a group of data points, according to a certain algorithm, and fixing the point corresponding to boundary line between the objects in the robot environment.’\textsuperscript{(14)} This was done choosing a maximum admissible distance between the consecutive scanned points. This approach does not guarantee exact splitting of groups, but assumes and is based on the fact that the distance between the scanned points is proportional to the distance between the range finder and the obstacle. Due to variability in the above-described assumption, another algorithm was developed by the author of the paper. This algorithm examined the distance between consecutive scan points.

The result of this step was a list of groups. Every group in this list is an array of scanned points. The list is verified with intention to delete the groups that contain too few data points.
This seemed very appropriate for the CHS in situations where the mine wall would have deep undulations, and the application of this methodology would treat the undulation as not recognizable and identify a single long line to represent the mine wall.

Figure. 25. Distance between measured points

‘Now, the points in each group were used to create a line using the least squares method as in the simple line-finding. An iterative end point algorithm was applied to find single elements of lines in these groups. According to this algorithm, a temporary line is drawn between the first and last points in the group. Next, the point, which is the farthest away to this temporary line, is searched for. After this point was found, each group of points is divided into two parts. So long as there were points which lie further than a given prescribed distance $r_{\text{min}}$ to the temporary line, the algorithm ran recursively’.

‘The result of this algorithm was an array of points-of-division for every group. A straight line was computed for every line between two dividing points using the array of these points’. 
This method was implemented on the 1/10\textsuperscript{th} scale model and it yielded successful results. Though the algorithm is currently undergoing changes, it is anticipated that this might be the final technology for navigation due to its robustness and performance in extreme wall conditions.

7.2. “Where Should I be” – Path Planning

Once self-localization is performed, the autonomous vehicle then requires commands for its further motion. This required a pre-defined or a dynamically computed path for the robot to trace. Thus, the word ‘path planning’ is associated with this research.

7.2.1. Simple Path planning

The initial stages of path planning for the CHS research began with the process of tethering different simple curves (straight lines and circular arcs) to produce a simple and easy path. The main theme behind the selection of simple path geometry like straight lines and circular arcs was to maintain simplicity and to establish confidence and study the response of the CHS to these types of simple curves. As explained under Section 4.2 of Chapter 4, the path generation simulation was targeted at this process of finding a traceable path for the CHS to follow.

7.2.2. Mine plan dependent path

The discussion under this Section continues from the discussion in Section 4.2 of Chapter 4. The mine plan dependant path conveys the meaning that the path developed will be a function of the mine plan parameters. A path for a 90-degree mine plan was developed and the worst case of a ‘S’ curve (a left turn followed by a straight segment and an immediate right) was considered for development.

From Long-Airdox material, it was evident that the geometry of this mine plan in combination with the geometry of the MBC made it a worst possible scenario for driving. Thus,
this was chosen for initial development with the motive that if the path for this mine plan proved to be satisfactory, then it was just a matter of relaxing some constraints.

Figure 26, below illustrates the path for a 90-degree mine plan.

![Figure 26. The path plan for a 90 degree mine plan](image)

This path was programmed into the algorithms and laboratory tests on the 1/10\textsuperscript{th} model yielded very good results although the path some changes due to the length of the PIG.

### 7.2.3. Implementation of the DMM and DPP results

As previously discussed in Chapter 4, the CHS system would be used in a variety of mines with changing mine patterns, and hence, dependability on a parameter that is subjective to change is not a good methodology to adopt. The implementation of the results from the DMM and DPP simulations will benefit the navigation of the system in any mine plan irrespective of its changing parameters. It is anticipated that experimentation with the DPP simulation and the implementation of its results will be performed by the LAVT team.
7.3. “How do I get there” – Motion Control

Once the robot has been localized with reference to its environment, and the path planner has computed the feasible path, the next immediate step is to drive or navigate the system along this path. This process is also termed as ‘Motion Control’ in robot literature. This Section discusses the two main motion control strategies developed by the author for the CHS.

7.3.1. Simple motion control

This algorithm developed, was used only for the navigation of a MBC along a constant width roadway with zero orientation (no change in MBC orientation – always pointed towards the direction of the . Hence the name ‘Simple Motion control’. The algorithm uses the data from the line-finder and attempts to drive the MBC, offset at a specified distance from the mine wall. The algorithm iterates for every scan of the scanner and computes the left and right track velocities based on the position at that instant of time.

The algorithm can be illustrated with a simple example. It employs a simple IF-ELSE logic for navigation, and is iterative in nature. The algorithm starts off by initializing the system by supplying a constant value for the left and right rack speeds irrespective of the configuration at that instant. For instance, if we desire to drive the MBC always at a constant distance ‘d’ from the wall for every time instant, and for the full course of the path, the algorithm uses the MBC’s location data (D) combined with this parameter to compute the offset if any. D is the distance of the center of the MBC from the mine wall. The offset may be a positive value, negative value or zero based on the MBC’s location from the wall at that instant of computation. In a similar manner, the orientation of the MBC is also categorized into different sub-categorizes, viz., positive (counter clockwise), negative (clockwise) and zero (no orientation).

Now, based on the values of D and d, we can compute the offset as follows…

Offset = d – D (inches).
Thus, we can categorize 9 different cases based on the values of Offset and $\theta$, the angle of the MBC with reference to the mine wall. Based on each case, the track speeds are changed by a constant value. For instance, if the offset is positive (the MBC is towards the right of the desired path) and if $\theta$ is positive (counter clockwise), navigating the MBC in the same manner would get it back on the path. Figure 27, illustrates a typical scene with the algorithm’s parameters.

Figure. 27. Simple Motion Control implemented on a single MBC

Thus we use the terminology ‘L+ and R+’ for this case, referring that the Left and Right track speeds are to be the same and maintained until the MBC reaches its desired distance from the wall. In short, the algorithm just attempts at maintaining the set offset from the wall at every point on the path, and hence, cannot be categorized as a ‘Motion planning’ strategy. Though the final goal of driving the MBC on a path is achieved, the main aim of the algorithm is always to null out any errors in orientation and maintain the wall offset. The following table lists the 9 possible cases that the algorithm has to check for in order to issue track commands.
<table>
<thead>
<tr>
<th>S.No</th>
<th>Offset (O)</th>
<th>Angle (θ)</th>
<th>Right Track speed (V_R)</th>
<th>Left Track speed (V_L)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Zero</td>
<td>Zero</td>
<td>R</td>
<td>L</td>
<td>An ideal case – maintain track speeds.</td>
</tr>
<tr>
<td>2</td>
<td>Zero</td>
<td>Positive</td>
<td>R-</td>
<td>L+</td>
<td>On the path, but with positive orientation. Thus increase Left speed and reduce Right speed to correct this orientation back to zero degrees</td>
</tr>
<tr>
<td>3</td>
<td>Zero</td>
<td>Negative</td>
<td>R+</td>
<td>L-</td>
<td>On the path, but with negative orientation. Thus decrease Left speed and increase Right speed to correct this orientation back to zero degrees</td>
</tr>
<tr>
<td>4</td>
<td>Positive</td>
<td>Zero</td>
<td>R+</td>
<td>L-</td>
<td>No orientation, but on the right side of the desired path. Hence decreasing the Left track speed and increasing the right will bring it back to desired path</td>
</tr>
<tr>
<td>5</td>
<td>Negative</td>
<td>Zero</td>
<td>R-</td>
<td>L+</td>
<td>No orientation, but on the left side of desired path. Hence, increasing left and decreasing right track speed will restore desired condition.</td>
</tr>
<tr>
<td>6</td>
<td>Positive</td>
<td>Positive</td>
<td>R+</td>
<td>L+</td>
<td>Maintain speed</td>
</tr>
<tr>
<td>7</td>
<td>Positive</td>
<td>Negative</td>
<td>R+</td>
<td>L-</td>
<td>Differential speed.</td>
</tr>
<tr>
<td>8</td>
<td>Negative</td>
<td>Positive</td>
<td>R-</td>
<td>L+</td>
<td>Differential speed</td>
</tr>
<tr>
<td>9</td>
<td>Negative</td>
<td>Negative</td>
<td>R+</td>
<td>L+</td>
<td>Maintain speed</td>
</tr>
</tbody>
</table>
Figure 28, illustrates all the above cases discussed in the table for additional clarity by graphically describing the algorithm.

Figure 28. Simple Motion Control – 9 possible cases handled by the algorithm

This algorithm when tested on the prototype and full-scale systems, yielded satisfactory results.

7.3.2. Path following – SSS Algorithm

From the description of the Simple Motion control strategy, it can be observed that this algorithm would perform very well in cases where there is no variability in the mine parameters. Once variability is introduced, the algorithm would produce unsatisfactory results. Moreover, the algorithm does not actually navigate the MBC on a path as desired by a path planner, but just tries to null out the orientation and to drive the MBC at a constant distance from the mine walls. Hence, a need for a more robust technique was required. This prompted the development of a new algorithm called the ‘SSS’ algorithm. The algorithm is a two-step algorithm that tries to correct the orientation prior to the linear distance and the SSS is an acronym for ‘STEER-Straight-STOP’.
The algorithm employs a two-step method to position the MBC on the desired path. To accommodate the SSS algorithm, the path planner has to provide the Motion planner with the current MBC location $S(X_s, Y_s, \theta_s)$ and the desired or target location $F(X_f, Y_f, \theta_f)$. These values are a function of the path. Given these two parameters, the algorithm first computes the difference between the two orientations $(\theta_f, \theta_s)$ to estimate the correction angle $\delta \theta$ so that the MBC could approach the final desired angle. The next step is to correct this angle by providing corresponding track velocities. This step is the STEER step. Once steering to the desired angle is performed, then the linear distance between the start and final points is computed and the tracks are commanded with this computed velocity so that the MBC reaches the target location. This is the STRAIGHT step. This is then followed by the STOP wherein the system halts at the final stage. The algorithm runs iteratively for the full course of the planned path.

Hence, it is clear that the algorithm is a direct motion control technique and actually aims at guiding the MBC along the desired path unlike the simple motion control. The two steps of angle correction and linear distance are described below in Figure 28.

![Figure 28. Illustration of the SSS algorithm](image-url)
The following summarizes the algorithm...

i. Given the points S and F (both functions of X, Y and $\theta$), the algorithm first computes $\delta\theta = \theta_f - \theta_s$ and $\theta_f = \tan^{-1}(\delta Y/\delta X)$. This is the change in angle required to reach the desired orientation.

ii. Now, the track speeds are computed by the simple relation, $V = r\omega$, where $V$ is the linear velocity, $r$ the radius of turn, and $\omega = (\delta \theta / \delta t)$, or the angular velocity. Thus the modified expression for the track speed is now given by...

$$V = \frac{B \frac{\partial \theta}{\partial t}}{2} = \frac{B \frac{\partial \theta}{\partial t}}{2 \, 0.5},$$

where $B$ is the wheelbase for the MBC or the distance between the track centers. A value of $t = 0.5$ seconds is chosen due to the fact that the operating frequency of the SICK is 2 Hz. Now the left and right track speeds are just the addition of either a positive or a negative sign to $V$ depending on the value of $\theta$.

iii. Once the angle is corrected, then the following relation computes the linear distance between the current and goal points.

$$d = \sqrt{\partial X^2 + \partial Y^2}$$

iv. The next step is to compute the track speeds for this linear distance correction. Thus the track speeds are now given by...

$$V = \frac{d}{\partial t} = \frac{d}{0.5}.$$ The same 0.5 sec for the time interval is assumed.

v. After this correction, the MBC would have reached its final target point with the desired orientation.

vi. Repeat steps i through v until the MBC navigates the full course of the path.
The SSS algorithm was implemented on the scale model and it has shown to yield successful test results.

7.3.3. Analysis of Dynamic Motion planning

Though the SSS algorithm performed satisfactorily on both laboratory and field-tests, there were shortcomings in the overall performance. Due to the heavy mass of the CHS and the presence of the PIG joint constraints, positioning the MBC accurately on the path required a feedback loop that would continuously monitor the system’s performance. The SSS algorithm was not designed as a continuous loop, but as an open loop corrector algorithm. This open loop nature of the algorithm induced oscillations in the system due to overshoot and undercuts, in trying to position the MBC back on the path. Moreover, the algorithm was unable to locate the MBC exactly on the desired point on the path without resulting in configuration extremities or collision against the walls. Thus, a better algorithm of motion control was desired. The LAVT team is currently exploring the possibilities and feasibilities of applying Neural Networks towards the control of the MBC and will analyze the feasibility of this new technology in developing state space models for a more robust motion control algorithm.
CHAPTER 8: CONCLUSIONS

The development of both interface and control software involved numerous challenges and a variety of interlaced tasks. Though the time involved and the cumbersomeness in developing the interface software development was reduced multifold due to the use of LabVIEW®, the establishment of a communications linkage between LabVIEW® and the control algorithms written in C via the Code Interface Nodes proved to be the most challenging task.

All the control algorithms described in the previous chapters were extensively tested and the LabVIEW® interface developed is currently employed as the main control interface between the MBC hardware and the host PC. The serial data communication techniques that were developed were thoroughly tested and verified on the field by running time tests on a multi-unit system. Though not much effort has been spent on detailing the simulation codes, it must be noted that the simulations developed for the CHS acted as the base platform for the control algorithm development and the results of the simulations were directly converted to object code for the Code Interface Nodes.

The algorithms were developed with a strong theoretical base and their appropriateness for use on the CHS and their robustness under extreme conditions have been verified by both the laboratory tests on the 1/10th scale model and field tests on the full scale MBC. It is anticipated that there will be no requirement for further developments on the Interface software as the most recent VI developed has in it embedded, more functions than actually needed for a multi-unit system. Likewise, there will exist no requirement for extensive simulations and the only area that has been left untested are the DMM and DPP simulations. The current developments in the control algorithms will tend to focus more towards the refinement of the SSS algorithm and the iterative end point line splitting technique.

It is also anticipated that the results of the DMM simulation will be robust enough to identify the appropriate path planning strategy for the SSS algorithm, and thereby generate track velocities. Thus, the author’s research has focused more on the initial stages of development and
in establishing the right techniques and methodologies for future extension in the CHS automation effort.

In addition, a Quality Assurance Plan (QAP) has been compiled by the author to qualitatively and quantitatively measure the performance of the PMS laser scanner and the control algorithms. The purpose of this test will be to ascertain confidence in the methodology. In brief, the QAP measures the robustness of the data and control algorithms by performing repeatability tests. The results of this test will aid the team in refining the development process if need be. Though the QAP test results are not discussed in this report due to the fact that the plan is still in the infancy stage, the author will work closely with the team in implementing the test.

It is with hope that this document will act as a complete reference documentation for both the VT team members and Long-Airdox personnel. The thoroughness of explanation and illustrations in Chapters 7 & 8 will completely introduce the author’s work to a new team member who will extend this work in producing the fully autonomous CHS. This documentation maintains a balance between the description of intermediate tasks performed and the final goals of achieving a fully autonomous system without having overlooked any vital information.
CHAPTER 9: FUTURE RESEARCH

Generic automation algorithms

The author’s contribution in this research has focused on developing interface and control strategies for mine equipment automation. Though the algorithms that were developed are highly specific and customized for use on the Multi-unit Continuous Haulage System, they can be massaged for use on other similar robotic systems.

Hence, the term Generic automation algorithms is suggested in the context of developing automation algorithms that can be applied to a variety of electromechanical systems. The results of this work can be modified to generate generic automation control algorithms that can meet the automation needs of other similar systems at Long-Airdox or elsewhere, resulting in either full or partial automation. The sensor technologies, and the sensor based navigation strategies can be ported over to a variety of mobile robot applications. The concept of line-finding can be applied to almost any kind of sensor system that is a range measuring device. One of the potential advantages is that the control techniques developed as a part of this work will be used to automate LA’s other systems like the Battery Haulage, the continuous miner and the Shearer.

From a control’s perspective, the usage of LabVIEW® and C is a rapid and improvised method of building interface software. The author and Dr. Sturges have summarized the potential advantages of this new hybrid system for developing mobile robot prototypes in their paper titled “A hierarchical Real Time Control System for Development of Mobile Robot Prototypes”.

One other area for future developments is the direct linkage of the simulation models to the physical system. The potential advantages of this amalgamation are that the simulation results can be directly put into test on the physical system, thus avoiding the intermediate steps of writing system specific code. Such a system will have the capabilities to graphically simulate the CHS and will dynamically transfer all the simulation results to the physical system.
REFERENCES


VITA

Aishwarya Varadhan was born in 1977 in a small ‘temple city’ close to the southern shores of India. He attended schooling in his hometown and then moved to another small ‘textile city’, about 200 miles away from home, for his undergraduate education. As an undergraduate, Aish decided to specialize in Mechanical Engineering, as this was the closest match to his interests and passion for building robots, and autonomous machines that will replace us humans! After 4 great years in undergraduate school, he graduated with a B.S.M.E, and decided to further his education with the goal of building mean robotic machines. In 1998, he joined the ISE department at Virginia Tech, and spent the best two years of his academic and research life. Being a part of Tech was much more than just being a Hokie! He enjoyed his courses, research and of course the Scuba Dives in the New River and the long bike rides. Graduate school has really had a great impact on his decisions to pursue advanced research. Thus, Aish will be looking forward to an advanced degree or a research career leading to specialization in Robotics and Visualization. He still dreams about building an underwater robot for use as a dive buddy while Scuba diving!