University of Southern Queensland

Faculty of Engineering & Surveying

Development of A Low Cost GPS Guidance System for use in Agriculture

A dissertation submitted by

N. Hayllor

in fulfilment of the requirements of

ENG4112 Research Project

towards the degree of

Bachelor of Engineering Majoring in Mechatronic

Submitted: November, 2006

Abstract

This project developed an accurate, cost effective GPS guidance system and developed conceptual designs of a hands-free guidance interface for use in the agricultural sector. The guidance system utilised two Garmin GPS 18-5 receivers to provide position data at a rate of 5Hz. The guidance system was built upon a previous incarnation that utilised Garmin GPS 35 receivers providing position data at a rate of 1Hz. The guidance system currently uses an indicating arrow on a computer screen to provide guidance information to operators.

The accuracy with which the guidance system could guide an operator in a straight line up a paddock was quantified by monitoring the path followed by the operators machine out and back along a defined trajectory and then compared to current industry standards. The accuracy of the guidance system to guide an operator along parallel paths offset a distance from each other was also quantified. The ability for the software to offset over different distances was also determined.

The information collected and conclusions drawn will be used to quantify the accuracy of the guidance system as a whole for agricultural applications.

It is hoped that the conceptual designs for the hands-free guidance interface will be used to develop a working prototype and be tested with the guidance system in the near future.

Certification of Dissertation

I certify that the ideas, designs and experimental work, results, analyses and conclusions set out in this dissertation are entirely my own effort, except where otherwise indicated and acknowledged.
I further certify that the work is original and has not been previously submitted for assessment in any other course or institution, except where specifically stated.

N. Hayllor

0050009687

Signature

Date

Acknowledgments

This thesis was typeset using $\LaTeX\,2_\varepsilon$.

This research was carried out under the principal supervision of Prof John Billingsley, Professor of Mechatronic Engineering at the University of Southern Queensland and I would like to thank him for his guidance.

I would also like to thank Mr Rudi Bartels and Mr Mark Dunn of the National Centre for Engineering in Agriculture for providing the Garmin receivers and many hours of technical assistance with Zigbee transmitters. Much thanks must also go to the technicians on Level 2 of Z-block for without the cable they supplied, testing would not have been a reality.

Finally I must thank my family for without their help, testing this project would have been infinitely harder.

N. Hayllor

University of Southern Queensland

November 2006

Contents

Abstract **i**

Acknowledgments **iv**

List of Figures **xii**

List of Tables **xvii**

Nomenclature **xviii**

Chapter 1 Introduction **1**

1.1 Introduction 1

1.2 Project Objectives 3

1.2.1 Upgrade Hardware and Quantify Gains in Accuracy 3

1.2.2 Upgrade Software to Accommodate New Hardware, Add New Features and Quantify

1.2.3 Develop System for Autonomous Steering Operation 4

1.3 Overview of the Dissertation 4

1.4 Conclusions 4

Chapter 2 Literature Review and Assessment of Other Products 5

2.1 The Global Positioning System 5

2.1.1 Overview 5

2.1.2 Segments of GPS 6

2.1.3 GPS - The Basic Idea 7

2.1.4 Usages 9

2.2 The Global Positioning System in Agriculture 12

2.2.1 Usages 12

2.2.2 Methods of Guidance 14

2.3 Data transmission 17

2.4 Methods of Hands-Free Guidance Operation 18

2.4.1 Electrohydraulic Solenoid Steering Valves 18

2.4.2 Steering Wheel-based Actuators 20

2.5 Interfacing Computer Systems with Mechanical Devices 22

2.5.1 The H-Bridge 22

Chapter 3 Data Collection and Testing Methods 24

3.1 Introduction 24

3.2 Small-scale testing 24

3.2.1 Equipment 25

3.2.2 Testing and Data Collection Methods 26

3.3 On-Farm testing 27

3.3.1 Equipment 27

3.3.2 Testing and Data Collection Methods 28

3.4 Conclusion 30

Chapter 4 Review of GPS35 Receiver Software 32

4.1 Introduction 32

4.2 Data transmission 32

4.3 Software 33

4.4 Testing of System 34

4.4.1 Small-scale accuracy 34

4.4.2 On-farm accuracy 35

4.5 Usability of Guidance Software 38

4.6 Conclusions 40

Chapter 5 Development of Autonomous Steering Unit 41

5.1 Introduction 41

5.2 Design Requirements 42

5.3 Conceptual Design 42

5.3.1 Differential Type Planetary Gearbox 42

5.4 Other Ideas 49

5.5 Conclusions 50

Chapter 6 Development of GPS18-5 Compatible Software 51

6.1 Introduction 51

6.2 Baud rate and parity errors 51

6.2.1 Baud rate 52

6.2.2 Parity 52

6.3 Receiving NMEA 0183 Almanac Sentence 53

6.3.1 Downloading NMEA data to a text file 55

6.4 Converting NMEA 0183 to YUMA format 56

6.4.1 Variables of getnmea 56

6.4.2 Opening Data File 57

6.4.3 Retrieval and Computation of Data from File 58

6.5 Development of YUMA data file 62

6.5.1 Definition of YUMA Almanac 62

6.6 Useage of YUMA data 64

6.7 Adding Guidance Code to Software 65

6.7.1 Position data for use in guidance 66

6.7.2 The ‘dokeys’ subfunction 66

6.7.3 The ‘Set From’ and ‘Set To’ commands 67

6.7.4 The ‘makenormal’ subfunction 68

6.7.5 The ‘showheading’ subfunction 68

6.7.6 The ‘outward’ variable 69

6.7.7 Saving and Loading a Heading 70

6.7.8 The ‘model’ subfunction 70

6.7.9 ‘Nudge’ Ability 72

6.8 Simulation Programs 73

Chapter 7 Testing of GPS 18-5 Based System 75

7.1 Introduction 75

7.2 Data transmission 75

7.3 Testing of system 77

7.3.1 Accuracy of Orbit Data Calculations 77

7.3.2 Small-scale accuracy 78

7.3.3 On-farm accuracy 79

7.4 Usability of Guidance Software 84

7.5 Conclusion 84

Chapter 8 Conclusions and Further Work 85

8.1 Introduction 85

8.2 Hardware Upgrade 85

8.3 Software Upgrade 86

8.4 Develop Autonomous Steering 89

8.5 Discussion of Possibilities for Further Development 90

References 92

Appendix A Project Specification 95

Appendix B Autonomous Steering Unit Conceptual Design Information 97

B.1 Initial Conceptual Design 98

B.2 Adjusted Design with Offset 99

B.3 Detail Drawings 100

B.3.1 Planet Lock Detail Drawing 100

B.3.2 Steering Wheel Splined Shaft Mount 101

B.3.3 Extension Shaft 102

B.3.4 Steering Column Mount Boss 103

Appendix C Software Source Code 104

C.1 Introduction 105

C.2 The `change38400` VB Subfunction 106

C.3 The `getposrecord` VB Subfunction 107

C.4 The `askalm` VB Subfunction 108

C.5 The `NMEA2YUMA` VB Subfunction 109

C.6 The `steer1` VB Code 112

C.7 The `calcvel2` VB Subfunction 113

C.8 The `solvev` VB Subfunction 114

C.9 The `dokeys` VB Subfunction 115

C.10 The `makenormal` VB Subfunction 117

C.11 The `showheading` VB Subfunction 117

C.12 The `model` VB Subfunction 118

C.13 The `Motorsim` VB Function 121

Appendix D Satellite Orbit Data, Drift Testing and Guidance Trajectories123

D.1 Comparison of calculation of satellite orbit parameters 124

D.2 Small-scale testing data for GPS 35 Guidance System 126

D.3 On-farm testing data for GPS 35 Guidance System 128

D.4 Data transmission setup for GPS 18-5 System 130

D.5 Small-scale testing data for GPS 18-5 Guidance System 131

D.6 On-farm testing data for GPS 35 Guidance System 133

List of Figures

1.1 An example of parallel swathing 2

2.1 GPS Constellation 6

2.2 Location of GPS MCS and other stations 7

2.3 Pseudo Random Code Transmitted by GPS Satellite 8

2.4 Triangulation from Satellites 8

2.5 Method to Gain Near-Atomic Accuracy in GPS Receivers 10

2.6 Swath Control on John Deere GreenStar 2^{TM} 13

2.7 Timing errors with pseudo random code 16

2.8 Timing of pseudo random code with carrier signal 17

2.9 A MaxStream XBee Pro RF Module 18

2.10 Example of hydraulic steering on farm tractor 19

2.11 Complexity of interfacing electrohydraulic valve to tractor steering system 20

2.12 Steering column drive interface 21

2.13 Steering column mounting point for gearbox 21

2.14 The general arrangement of a H-Bridge 22

3.1 Data collection setup for small-scale testing 25

3.2 Data collection setup for on-farm testing of GPS system 31

4.1 XBee Pro Zigbee transmitter with PCB 33

4.2 Drift of GPS 35 system 35

4.3 Operation of guidance system during testing 36

4.4 Locating point for guidance accuracy testing 36

4.5 Log of trajectory of test rig 37

4.6 A trajectory where satellite lock was lost and then regained 39

5.1 Configuration of a planetary gearbox 43

5.2 Initial design with two planetary gearboxes mounted together 44

5.3 Schematic of Chosen Gear 45

5.4 Steering column mounting point for gearbox 46

5.5 Interface between steering wheel mount and gearbox 46

5.6 Conceptual design of gearbox 47

5.7 Conceptual design of offset system 48

5.8 Less complicated steering design 49

6.1 NMEA Sentence configuration 53

6.2 Code to generate file name and location 55

6.3 Code to store almanac data to text file 55
6.4 Code to close data file 56
6.5 Code required to generate array of nmeadata 57
6.6 Code to open almanac data file 58
6.7 Code to retrieve almanac sentence from file 59
6.8 Example NMEA 0183 Sentence 59
6.9 Code to retrieve data from file 60
6.10 Code for function ‘getalmdata’ 60
6.11 Code to use ‘getalmdata’ 60
6.12 Method of storing to separate dimensions of nd 61
6.13 Code to retrieve Hex data from sentence 61
6.14 Calculations on signed Hex data 62
6.15 Definition of YUMA almanac 63
6.16 Code required to write YUMA data to file 64
6.17 Comparison of NMEA 0183 String to YUMA Format 64
6.18 Screenshot of Guidance Screen 65
6.19 Code to gain ASCII value of pressed key 67
6.20 Code to imitate a key press 67
6.21 Guidance heading 69
6.22 Code to calculate side and Travel 70

6.23 Code to generate accurate swath widths 71

6.24 Code to display guidance arrow on screen 72

6.25 Code to implement ‘nudge’ ability 73

6.26 Code simulating swath width control 74

7.1 Transmission cable used to connect base station to machine 76

7.2 MaxStream XBee Pro RF Modem 77

7.3 Comparison of Garmin and Calculated Orbit Data 78

7.4 Drift of GPS 18-5 system 79

7.5 Motor configuration for testing hands-free guidance 80

7.6 Record of path travelled by tractor moving along trajectory in both directions 81

7.7 Guidance system swath-width offset testing 82

7.8 Path followed over 24m swath offset 83

8.1 Comparison of Garmin and Calculated Orbit Data 87

8.2 Record of path travelled by tractor moving along trajectory in both directions 88

8.3 Guidance system swath width offset testing 89

B.1 Exploded view of gearbox design 98

B.2 Alternative view of gearbox offset design 99

B.3 Detail design for part Planet Lock 100

B.4 Boss connecting steering wheel to gearbox via chain 101

B.5 Rod that extends column length adjuster 102

B.6 Boss to mount gearbox on steering column 103

D.1 Orbit Parameters of Satellites calculated by Garmin and from almanac data124

D.2 Software representation of comparison of satellite orbit calculations . . . 125

D.3 Drift of mobile receiver in GPS 35 system 126

D.4 Drift of mobile receiver in GPS 35 system over 11 minutes 127

D.5 Trajectory of path around top of farm ring tank 128

D.6 Trajectory of machine across swath widths 129

D.7 Set up of transmission cable on tractor 130

D.8 Trajectory of machine across swath widths 131

D.9 Trajectory of machine across swath widths 132

D.10 Tyre path of tractor under guidance 133

D.11 Tyre path of tractor under guidance 134

D.12 Screenshot of steering screen while guidance is operating 135

List of Tables

2.1 Different operational states of a H-Bridge 23

4.1 Table of swath widths along trajectory 38

6.1 Variables contained within nmeadata type 57

6.2 Extra variables used by getnmea 58

6.3 Scale Factors for YUMA calculations 59

7.1 Table of distances between logged trajectories 81

7.2 Table of swath widths along trajectory 83

7.3 Table of swath widths along trajectory 84

8.1 Drift Comparison of GPS 35 and GPS 18-5 Receivers 86

8.2 Table of distances between logged trajectories 87

8.3 Table of swath widths along trajectory 88

D.1 Table of ring tank length 128

Nomenclature

The following nomenclature has been used throughout this report

GPS	Global Positioning System
DoD	U.S. Department of Defence
MCS	Master Control Station - Data processing station that calculates orbit adjustments of satellites
DGPS	Differential Global Positioning System
Skip	The area between two consecutive implement passes that is not affected by the implement
Overlap	The area between two consecutive implement passes that is affected more than once by the implement
Swath	The area of paddock effectively covered by a farming operation as it moves from one end of the paddock to the other
Swath Width	The effective width of the implement used in the farming operation
Parallel Swathing	The process of reducing overlap and skip by ensuring that for consecutive runs through the paddock, the centre of the implement is exactly one swath width from the centre of the previous run
NMEA	National Marine Electronics Association
NMEA 0183	Standard defining electrical signal requirements, data transmission protocol and time, and specific sentence formats for a serial data bus.
YUMA	A standard of converting NMEA 0183 almanac data to a more easily understood format.
RS232	Interface standard for data communications equipment. Uses handshake protocols to minimise data loss.
SV	Satellite Vehicle.
CSV	Comma separated values. File format where the data is separated by a comma, and new lines separated by a semicolon. Can be stored as a text file and read into Excel as a full spreadsheet.
RTK	Real Time Kinematic. A technique used in GPS systems where the guidance is based upon the use of carrier phase measurements.

Chapter 1

Introduction

The overall aim of this project was to further develop and implement a GPS guidance system for use in agriculture. The guidance system utilised the use of GPS signals from two separate receivers to calculate the field position of the machine relative to a starting point. It used this information to ensure the position of the machine was not only inline with a predetermined heading, but also at the right offset from the starting point. The project was undertaken to continue current research in the hope that an effective system would one day be ready for commercial retail for a price well below that of current packages.

1.1 Introduction

The Global Positioning System (GPS) has become a part of modern-day life. Whether we know it or not, we all use GPS in our daily routines. Since its launch in the 1970's by the U.S. Department of Defence (El-Rabbany 2002) , the use of GPS has become immense. It is used to guide aircraft around the world, track emergency devices and guide us around our own countries in the form of in-car navigation.

Due to its ability to position vehicles, GPS has been adapted into the agricultural industry in the form of accurate, straight-line guidance systems. These systems have been in development for many years now, with accuracy down to an amazing $\pm$2cm

(Trimble 2006). However, such mainstream guidance systems are extremely expensive to install (upwards of $80 000) and thus there stands the opportunity for a new cheaper, but still accurate, method of guidance.

Enter carrier phase GPS. This system provides agriculture with the ability to have ±2cm GPS accuracy for a fraction of the cost of mainstream systems. It does so because the system does not provide an absolute position in the world - it guides via calculation of displacement from a starting point (Billingsley 2006).

A guidance program using this system has already been developed by Professor Billingsley, but requires further enhancement to make it commercially viable in the agricultural industry. Currently, the software can guide any machine in a perfectly straight line for however long is necessary, but it cannot provide parallel swathing (see figure 1.1), or guide autonomous of the operator. At the start of this year the system was operational but required user input to steer the machine via an arrow on a computer screen, or a lightbar depending on its configuration. It used two Garmin GPS35 receivers, and was able to produce position and guidance data at a rate of 1Hz.

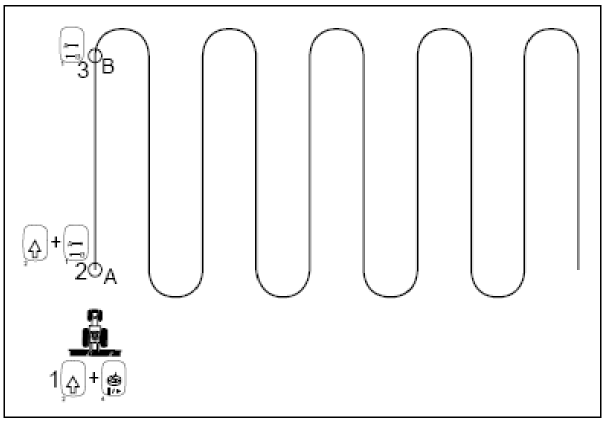


Figure 1.1: An example of parallel swathing

(Trimble 2002)

Research, testing and development is necessary to quantify the performance of this system, improve it for commercial use and compare its abilities with other systems on the market.

1.2 Project Objectives

1.2.1 Upgrade Hardware and Quantify Gains in Accuracy

New advances in technology have provided an ability to upgrade the receivers on the system to produce position and guidance data at 5Hz. The Garmin GPS1 8-5 receiver is an example of such new technology. The GPS 18-5 can output position data at a rate of 5Hz and is thus inherently more accurate than the GPS 35 receiver. Testing of the hardware will be undertaken to quantify the gains in accuracy provided by the new hardware. The software that 'talks' to the GPS35 receivers needs to be modified to communicate with the GPS 18-5's as some of the receivable information has changed between the models.

1.2.2 Upgrade Software to Accommodate New Hardware, Add New Features and Quantify Gains in Accuracy, Productivity and Usability

The software requires the GPS satellite position record (almanac) that can be downloaded from the GPS 18-5 receiver to calculate the location of every satellite in the sky. This is required as the GPS software requires the locations down to six decimal places and the 18-5 receiver only transmits this data as integers. The almanac data must be downloaded, converted and stored ready for use by the guidance program and code must be written to do this. Code must also be written to compare the software satellite location calculations to those coming from the GPS 18-5 receiver.

Code must also be added to provide the user with the ability to accurately offset over a required swath width.

Increases in productivity and usability will be quantified through the testing of the updated guidance system.

1.2.3 Develop System for Autonomous Steering Operation

A autonomous steering unit needs to be developed and interfaced with not only the software, but the steering column of a piece of farm machinery. This ‘hands-free’ steering ability is a major requirement as the main purpose of GPS guidance is to relieve the operator from the steering operation.

1.3 Overview of the Dissertation

This dissertation is organised as follows:

Chapter 2 Literature Review and Assessment of Other Products

Chapter 3 Data Collection and Testing Methods

Chapter 4 Review of GPS35 Receiver Software (Accuracy)

Chapter 5 Development of Autonomous Steering Unit

Chapter 6 Development of GPS18-5 Compatible Software

Chapter 7 Testing of New System

Chapter 8 Results, Conclusions and Further Work

1.4 Conclusions

Through analysis, modification and testing of the current software, improvement and testing of the new software, and development of a mechanical drive interface, it is hoped that a fully working prototype will be complete by the end of this project.

Chapter 2

Literature Review and Assessment of Other Products

2.1 The Global Positioning System

2.1.1 Overview

The Global Positioning System (GPS), a satellite-based guidance system, was developed by the U.S. Department of Defence (DoD) in the 1970's. The system was initially only available to the DoD but it was later made available to civilian users around the world (El-Rabbany 2002). The GPS system makes access to positioning and timing information available under any weather conditions throughout the entire world (Langley 1990).

GPS consists of a constellation of 24 satellites and their ground based monitoring stations (Trimble 2006). These satellites are arranged so that there is between four to ten satellites visible at any location on the Earth at any point in time (as four is the minimum amount of satellites required for accurate positioning). On the 17th July, 1995, worldwide coverage by the GPS system was confirmed - 24 fully operations satellites were operating in orbit (El-Rabbany 2002).

2.1.2 Segments of GPS

There are three main segments to the GPS system: the space segment, the control segment and the user segment (Wells 1987). The space segment consists of the satellite constellation mentioned previously and shown in Figure 2.1. A GPS satellite transmits a signal made up of five different components: two sine waves (carrier frequencies), two digital codes and a navigation message. The carrier frequencies and the digital codes are used to calculate the distances between the GPS satellites and the receiver; the navigation message contains data about the position of the satellite as a function of time (El-Rabbany 2002).

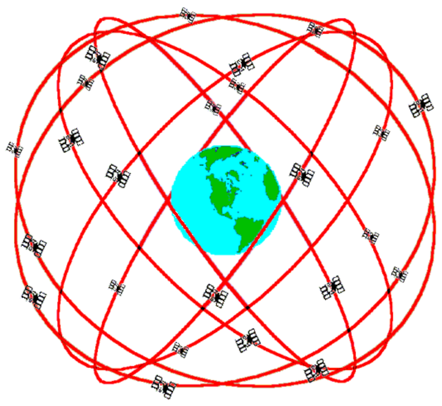


Figure 2.1: GPS Constellation

(Dana 2006)

The control segment of the GPS system is a network of ground-based tracking stations, control stations and a master control station (MCS) (see figure 2.2) situated at Colorado Springs, Colorado, U.S. This segment is used to track the satellites to determine and predict their locations, the behaviour of their on-board atomic clocks, the satellite almanac and other items. The coordinates of the tracking stations are known extremely accurately and each is equipped with a accurate GPS receiver. The data collected through these receivers is transmitted to the MCS where it is analysed and data such as the satellite positions, clock parameters and almanac are modified. This is then transmitted back to the satellites through a control station to ensure the accuracy and integrity of the GPS system (El-Rabbany 2002).

The user segment includes all of the end-users of the GPS system. From the military

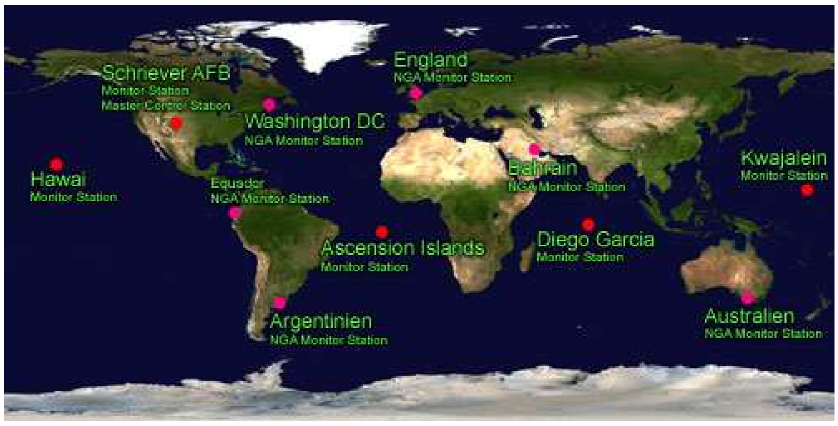


Figure 2.2: Location of GPS MCS and other stations

(Kowoma 2006)

to civilian users, anyone with a GPS receiver and antenna can use the GPS signal to calculate their position on the planet to within 8.5m (Shaw, Sandhoo & Turner 2000). The system is currently available to all users without charge, which is why it has become so readily adopted into industries such as aviation and agriculture.

2.1.3 GPS - The Basic Idea

The basic idea behind GPS is quite simple. If we know the distances between a GPS receiver on Earth, and three satellites in the atmosphere, and also the locations of those satellites, then we can compute the location of the receiver using triangulation (Trimble 2006). That is all well and good, but how do we go about calculating the distances between the receiver and the satellites?

Each satellite transmits a different code signal to the next. One part of this signal consists of positioning data of the satellite relative to time. This data updates the almanac (record of satellite positions) in the receiver and allows the receiver to pinpoint the location of every satellite it is talking to (El-Rabbany 2002). Another part of this signal is a pseudo random code (Figure 2.3) and it 'plays' from the satellite and in the receiver. Each piece of equipment starts 'playing' this sequence from the same point in time thus, due to the distance between them, a short delay will occur in the comparison of the signals. The receiver calculates the delay between its signal and the received signal and then multiplies this difference by the speed of light to calculate the distance to the satellite (Trimble 2006).

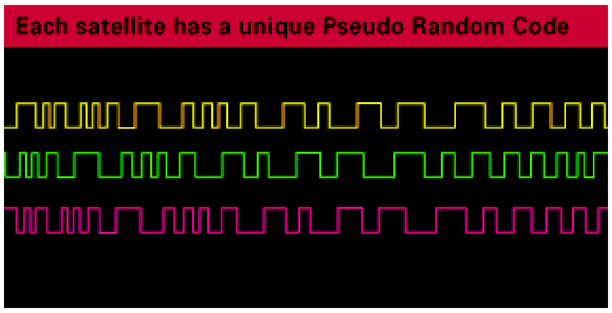


Figure 2.3: Pseudo Random Code Transmitted by GPS Satellite

(Trimble 2006)

Once we have calculated the distances from the receiver to three different satellites, we can calculate our position on the planet. By imagining that the three satellites are the centre of three spheres, each with a radius of the distance between the receiver and the satellite, we can narrow down our location to simply two points in space. It would be useful to have a fourth satellite to finalise our position, however it is easier to simply reject one of the answers as it is too far from earth or moving at an extreme velocity (Trimble 2006). This concept is shown in Figure 2.4.

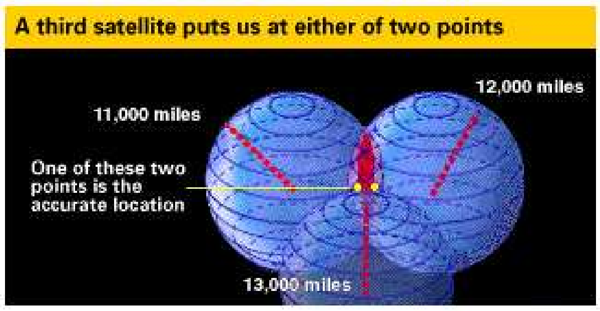

A third satellite puts us at either of two points
12,000 miles
11,000 miles
One of these two points is the accurate location
13,000 miles

Figure 2.4: Triangulation from Satellites

(Trimble 2006)

The calculation of position not only relies on good satellite positioning data, but also the accuracy of the clocks in the satellites and the receivers. The timing in the satellites is kept accurate by atomic clocks based upon Rubidium and Cesium (El-Rabbany 2002). Due to the cost of atomic clocks (between $50 000 to $100 000) it would not be conceivable to place them in receivers. Thus receivers clocks cannot be as accurate as the satellites' and this is a big problem because if the timing of a receiver is off by even a thousandth of a second, it translates to an error of over 300km (Trimble 2006). Luckily, a method was developed to provide receivers with near-atomic accuracy.

This method works in both 3-dimensions and 2-dimensions so for less confusion, we will work in 2-dimensions for the moment. As the ranges between the receiver and the satellites are calculated from time, we can simplify things by talking in terms of time.

If we calculate the time between the receiver and two satellites, we can calculate the intersection point between them (Figure 2.5 (a)). But what happens if our receiver timing is out by one second? Our calculated position is in fact a fair distance from where we actually are (Figure 2.5 (b)). To fix this problem we can make a timing measurement to a third satellite. In a perfect world, the three timing spheres would line up perfectly, however we know this is not true. The three spheres will not intersect at a point, and the software in the receiver detects this. The receiver calculates an offset that it can apply to the three signals to cause an intersection of the spheres, and as such, calculates its timing error (Figure 2.5 (c)). This error is used to update its internal clock, and from that point on the receiver is working on near-atomic accuracy timing. Of course, this calculation must be repeated often to keep the signal as accurate as possible.

As a GPS guidance system operates in 3-dimensions, the number of satellites required to obtain an accurate time and hence position fix is actually four; they are required to solve for the four dimensions x, y, z and t.

This is the basic operation of the Global Positioning System. The next sections will describe its usages and limitations and errors before moving to applications of GPS in agriculture.

2.1.4 Usages

The GPS System has more uses throughout the world than most people would care to imagine. They range from the global to the regional to the local level.

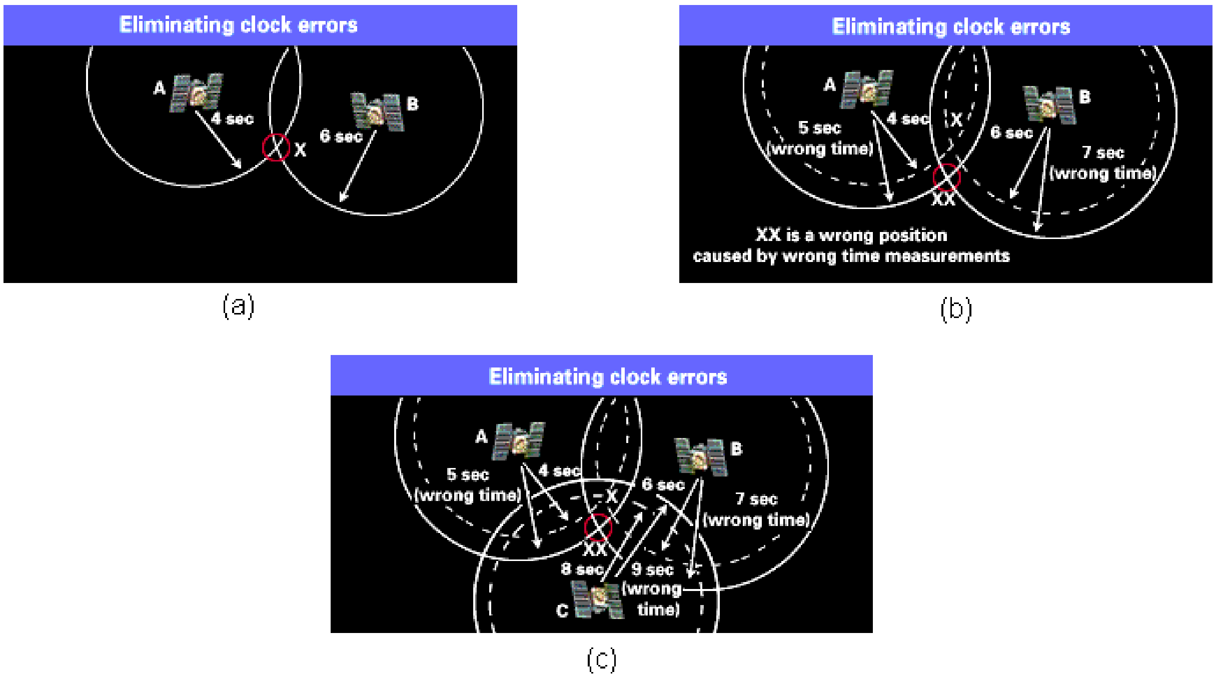


Figure 2.5: Method to Gain Near-Atomic Accuracy in GPS Receivers

(Trimble 2006)

Globally, GPS is used for navigation, surveying and timing and communications (Hofmann-Wellenhof, Lichtenegger & Collins 2001). For navigational purposes, the main users of GPS are the military and civilian operations such as air travel. It is a fact that todays jetliners and other aircraft can fly themselves from takeoff to landing using the GPS system (Emirates Airline Captain, 2005). Surveyors use the system to monitor global changes in geodynamical phenomena such as volcanic uplift and plate tectonics (Hofmann-Wellenhof et al. 2001). Timing and communications benefit from GPS through the accuracy of the satellite on-board atomic clocks. A receiver using just one satellite can provide a timing accuracy of just 40 nanoseconds. GPS should also help to increase the efficiency of communications due to its ability for very precise coordination (Hofmann-Wellenhof et al. 2001).

Regionally, GPS is used for navigation and surveying. Such navigational uses include: exploration, transportation management and various types of automation (Hofmann-Wellenhof et al. 2001). By using differential GPS, and placing fixed GPS receivers along the U.S. coastline, the U.S. Coast Guard has been able to develop an accurate guidance system to assist vessels approaching the coast (Hofmann-Wellenhof et al. 2001). Other uses are:

- Construction;
- Law Enforcement;
- Mapping and GIS;
- Mining; and
- Survey and Infrastructure.

  Source: (Trimble 2006)

In construction terms, GPS has greatly improved operational efficiency. You are no longer required to continuously survey an area to ensure accurate construction is occurring. GPS allows the final design to be uploaded into construction equipment so that the on-board computer can control the blade, bucket etc. to a precision unmatched by previous methods (Trimble 2006).

This same precision can be used for such operations as law enforcement. It is not uncommon to fit GPS receivers to high-risk (paroled) criminals who are banned from going near certain locations. Such a system spares man-power for other operations while still keeping a very watchful eye on the movement of these parolees (Hollingsworth 2006).

Locally, GPS is used the following applications:

- Agriculture;
- Emergency Services;
- Personal Guidance Systems; and
- Tracking of vehicles.
  Source: (Trimble 2006)

The emergency services use GPS to monitor the location of all their vehicles, as well as guide them along the quickest route to the emergency (Hofmann-Wellenhof et al. 2001). Everyday people use hand-held GPS locators during bushwalks and other outdoor activities. Not only do they tell you where you are (and provide an exact location to emergency services if you get lost), but some are also capable of logging your movements and as such allow you track your trip for later reflection (Trimble 2006, Broida 2003).

The tracking of vehicles is also useful for taxi's. GPS allows companies to track the movements of their vehicles to ensure that they are not wasting time and money or have been hi-jacked by a passenger.

2.2 The Global Positioning System in Agriculture

For many years now, GPS guidance has been finding many uses in the agricultural sector. Due to its ability to drive a piece of equipment perfectly straight (and now around corners and in circles), it has become an invaluable management tool for efficient farming. GPS helps to prevent operator error and fatigue, and through the use of controlled traffic farming, its ability to reduce overlap and skip provides savings on many levels such as fertilizer application, fuel and chemical application as well as placing wheeltracks consistently in the same location (GRDC 2005). Using GPS increases the efficiency of farming operations while maximising profit (Deere & Company 2006*b*).

2.2.1 Usages

The usage of GPS in agriculture is not limited to simply guidance of machinery. Due to its ability to pinpoint locations, it can also be used to map out a farm, including roads and other obstacles such as powerlines, and for yield monitoring purposes.

GPS technology has provisioned the development of technology such as variable rate seeding, spraying and fertilising. When a crop is harvested, a computer on-board the harvester logs the yield throughout the paddock, and this data is then analysed on PC to produce a map of optimal fertiliser application. This, combined with variable seed-

and spray-rate application helps to optimise the farming operation. Figure 2.6 shows a screenshot from a GPS system that systematically switches off spray nozzles on a sprayer along angled headlands. This technology can ultimately save spray as well as prevent overspray into other crops and spray drift (Deere & Company 2006*b*).

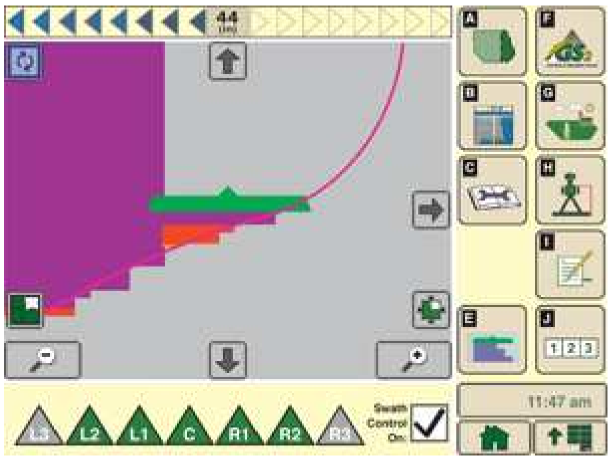


Figure 2.6: Swath Control on John Deere GreenStar 2^{TM}

(Deere & Company 2006*a*)

GPS can also be used in the operation of ground levelling, commonly called "laser levelling". Laser levelling is a process that irrigation farmers use to level their paddocks so that water will run along the paddock. Before GPS, this was accomplished by using a rotating laser set to a certain grade as a signal and a receiver on the machine. The machine would move up and down according to where it detected the laser on its receiver (Deere & Company 2006*b*). This method is quite successful, however it does not easily account for variations in paddock height and natural grade and is susceptible to interference by dust and other objects. In GPS levelling, a survey of the paddock is taken, then a contour map of the desired paddock grade is developed. This map is then entered into the on-board computer to automatically raise and lower the grader depending on location. The advantages of this method is that dust does not interfere with the signal, and that all variations in natural paddock geometry are easily accounted for (Trimble 2006).

2.2.2 Methods of Guidance

As there is more than one method of GPS guidance, it is necessary to distinguish between them so that comparisons between accuracies can be fully understood.

Manual Manual guidance is by far the cheapest, and sometimes most efficient method of guiding farm machinery. Cheapest because you do not need to invest in any further hardware or software to begin you guidance and most efficient because its chances of downtime are low. There are however many disadvantages to this system of guidance, mainly that operators cannot focus on driving straight 100% of the time: there are too many other operations that also need constant monitoring. Experience has shown that when manually steering machinery, turning around to look at the following implement causes you to twist the steering wheel without realising it, causing the machine to drive off line.

Standard Non-differential Standard, non-differential GPS is a guidance method that is by no means as accurate as differential GPS, but is still plenty good enough for many aspects of agriculture. This system uses only one GPS receiver to gain sub-metre position information. While you would never use this system for planting or furrowing of a paddock, it would be quite suitable for picking spray rows in the middle of the night (so you don't have to count and then try to pick the right place to drive down). As Trimble says, "Basic GPS can help you find the airport, Differential GPS can put you in the centre of the runway" (Trimble 2006).

Differential Differential GPS guidance uses two receivers to correct the errors in the satellite transmission caused by timing, environmental effects, and signals bouncing off tall objects. By placing a reference (immobile) receiver at a point that's location is known very accurately, the reference station can correct for any errors in the transmission.

Due to the relatively small distance between most mobile and reference receivers, you can quite safely assume that the signals are coming through the same part of the atmostphere, and hence will have the same errors.

The reference receiver corrects any timing and signal errors by attacking the positioning equations backwards. It knows its position very accurately, so instead of using timing signals to calculate its position, it uses position signals to calculate timing. If its calculated timing data is different to what it is receiving, then its sends this difference between the signals to the mobile receiver as an "error correction" factor (Trimble 2006)

A study into the efficiency advantages of GPS nagivation conducted by (Schrimpf 2005) produced the following results:

- Manual Guidance (lightbar) increased field speed by 13%;
- DGPS and RTK systems increased speed by 20%;
- Mean time in field was reduced by 11% using manual guidance; and
- DGPS and RTK systems reduced this by another 6%.

(Schrimpf 2005)

Code Phase vs Carrier Phase A GPS satellite 'speaks' more than one type of signal. There are in fact six different signals transmitted by each satellite, however most are not required for agricultural guidance.

The signals are:

- L1 Carrier Phase;
- L2 Carrier Phase;
- Coarse acquisition (C/A) Code;
- P-Code;
- Y-Code; and

- NAV Signal

  Source: (Kowoma 2006)

The only signals that are of interest to agricultural guidance are the L1 and L2 carrier phase signals and the C/A code.

Remember that a GPS receiver determines the travel time for a signal by comparing the pseudo random code its generating to that which it is receiving from a satellite. The receiver calculates the time by sliding its code along until both the generated and received codes line up. This type of guidance is called code phase.

The main problem with code phase systems is that because the bits of the pseudo random code are so wide, there is a fair amount of ambiguity in the resulting timing calculation.

Consider figure 2.7. You can see that the two signals are matched (i.e. when one is high, the other is high etc.), but can you also see that the signals are not exactly in phase? It is possible for one signal to change slightly before the other, and in doing so, generate large errors. This is because the pseudo random code is modulated upon the C/A signal which runs at 1.023MHz and as such has a cycle width of almost 1 microsecond. At the speed of light, 1 microsecond is almost 300m of error.

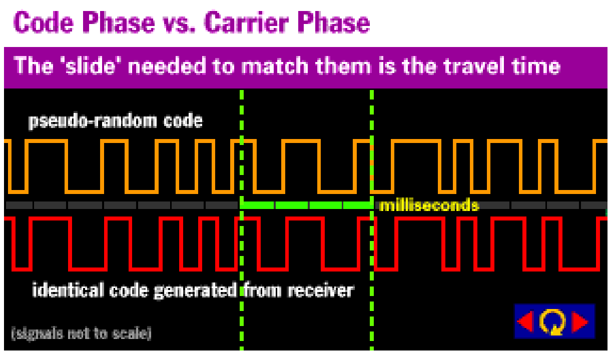

Code Phase vs. Carrier Phase
The 'slide' needed to match them is the travel time
pseudo-random code
milliseconds
identical code generated from receiver
(signals not to scale)

Figure 2.7: Timing errors with pseudo random code

(Trimble 2006)

There are however a few tricks that the receiver designers use in their software to ensure that the pseudo random numbers are almost perfectly in phase. But even when the accuracy is within a percent or two, the error is still in the magnitude of 3-6m.

The pseudo random number is also modulated upon the carrier phase signal and it is this signal that provides the real accuracy of the GPS system. As the carrier phase signals are transmitting at a frequency of approx. 1.57GHz, they are over 1000 times more accurate than the C/A code and hence much greater accuracy can be obtained. As we are modulating the 1MHz pseudo random number over the carrier phase, if we can line up our signal to within one percent of perfect phase, (see figure 2.8) we are able to generate an accuracy of between 3 to 4mm. Such is the accuracy required for parallel swathing and agriculture in general.

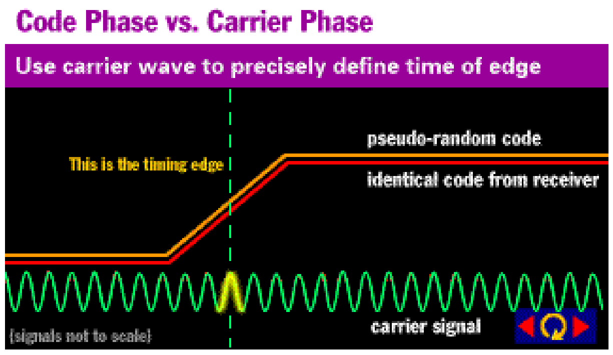

Code Phase vs. Carrier Phase
Use carrier wave to precisely define time of edge
This is the timing edge
pseudo-random code
identical code from receiver
(signals not to scale)
carrier signal

Figure 2.8: Timing of pseudo random code with carrier signal

(Trimble 2006)

The only problem with this system is because once carrier phase cycle is indistinguishable from the next, it is hard to count the number of cycles that have passed. This is where the code phase comes back into play. By getting 'close' with the code phase, you then only need to look at a few cycles of the carrier phase signal to calculate the proper timing and develop excellent accuracy (Trimble 2006).

2.3 Data transmission

A GPS guidance system requires that a stationary base station and a mobile receiver station be used. The data collected at the base station cannot be transferred to the guidance computer on the farm machine via a cable and hence must be transmitted via RF transmitters.

An RF transmitter than can be used for this is the MaxStream XBee Pro RF Module (shown in figure 2.9).

Figure 2.9: A MaxStream XBee Pro RF Module

(MaxStream n.d.)

The XBee Pro has an outdoor line-of-sight range of up to 1.6km and its transmit power output is 100mW. It has an operating frequency of 2.4GHz and a maximum RF data rate of 250kbps. It requires a supply voltage between 2.8V and 3.4V and has the dimensions 2.44cm by 3.30cm ((MaxStream n.d.)). The size of the XBee Pro makes it extremely useful for this type of application as it needs to be mounted on a tractor, and not take up large amounts of room.

The XBee Pro's RF range can be increased with the addition of a high quality aerial.

2.4 Methods of Hands-Free Guidance Operation

2.4.1 Electrohydraulic Solenoid Steering Valves

The most commonly used method for implementing autonomous steering in agricultural machinery, namely tractors and sprayers, is a system that uses an electrohydraulic solenoid to steer the tractor left and right. In today's modern society, there would not be one tractor capable of large-scale farming operations that does not have some sort of hydraulic power-assisted steering. There are two main types of hydraulic steering, one being separate hydraulic rams on each wheel hub, and the other one a single steering motor (or ram) in the middle of the machine connected to each hub with a control arm.

An example of hydraulic steering utilising a central ram and control arms is shown in figure 2.10.

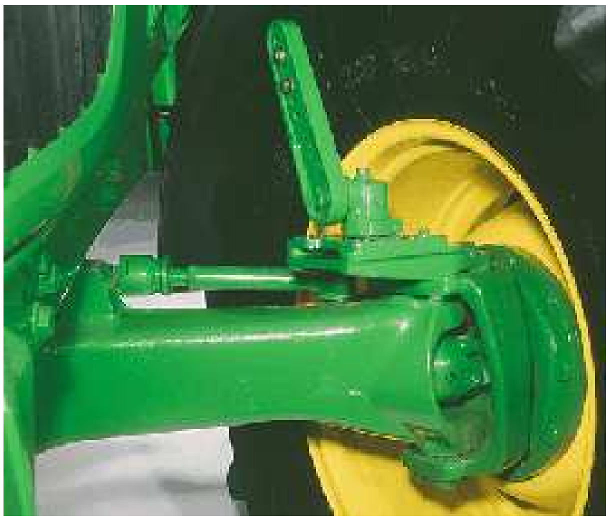


Figure 2.10: Example of hydraulic steering on farm tractor

(Deere & Company 2006*c*)

For autonomous guidance purposes, it is easiest to plum an electronic control valve into the oil system that supplies the steering. The actual hydraulic system is not simple however, and needs to be installed by a professional. The main reason for this is that the system must not turn the steering wheel instead of the wheels when activated, and must still allow the steering wheel access to the wheels when user input is required.

An example of the complexity of interfacing an electrohydraulic valve into a tractor's steering system is shown in figure 2.11.

When operating, the steering is being provided by the electrohydraulic valve, which is in turn controlled by the guidance computer. There should be no reason for the operator to touch the steering wheel unless the guidance system needs to be overridden (e.g. to avoid an obstacle unseen by the GPS).

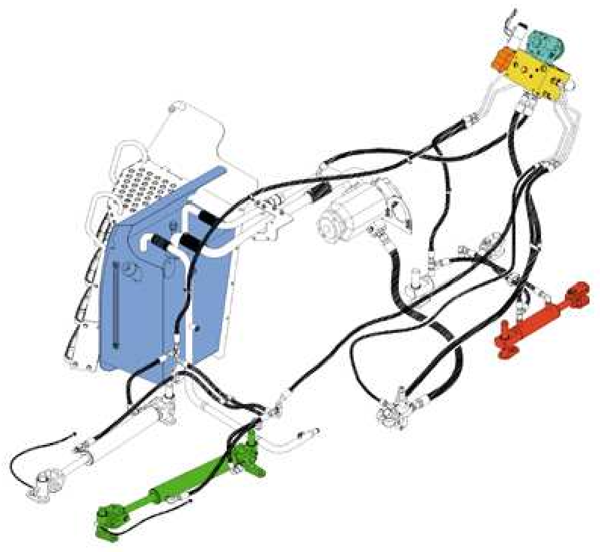


Figure 2.11: Complexity of interfacing electrohydraulic valve to tractor steering system

(Bruce Wiebusch 2001)

2.4.2 Steering Wheel-based Actuators

Actuators that are mounted on the actual steering wheel (or more precisely, steering column) have only just entered the market as a cheap and versatile way to implement autonomous guidance. Due to the fact that they mount directly to the shaft inside the steering column, there is no definitive requirement for a hydraulic steering system. However, such systems may not be capable of providing enough power to turn machinery that does not have power-assisted steering, but this is beyond the scope of this document.

Versatility is a very large requirement for modern farming technology. As mention in section 2.4, most common systems use an electrohydraulic valve to control steering, and such valves are not easily interchanged between machinery. A steering column mounted drive system could not only be installed by the operator, but is much much easier to shift between machinery. All it requires is the removal of a few bolts, and perhaps a change of adapter to fit different steering columns. An example of a steering column interface is shown in figure 2.12, and an example steering column mounting point shown in figure 2.13.

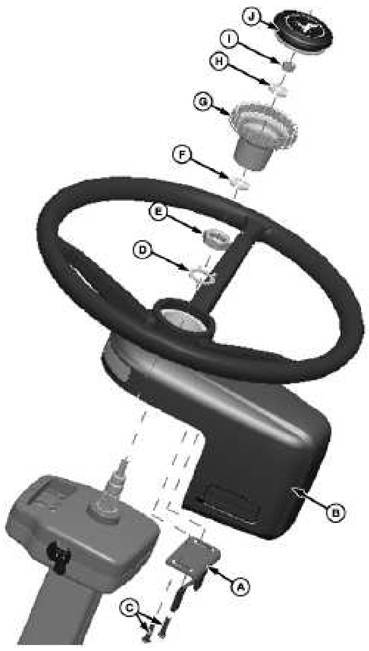


Figure 2.12: Steering column drive interface

(Deere & Company 2006*b*)

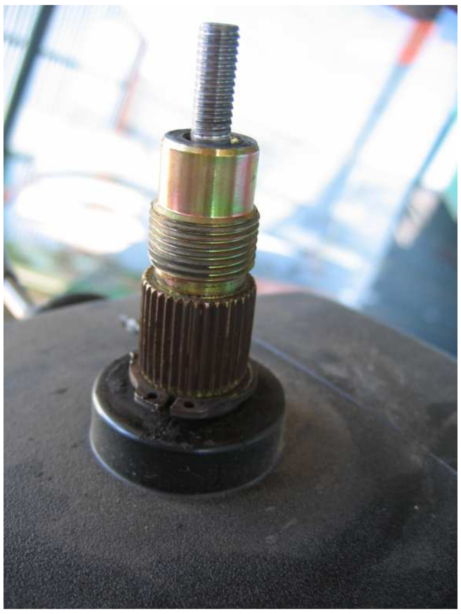


Figure 2.13: Steering column mounting point for gearbox

Personal Image

2.5 Interfacing Computer Systems with Mechanical Devices

2.5.1 The H-Bridge

A H-Bridge is so named because of the way the components on the board arrange to make the shape of a ‘H’. The four corners of the H comprise switching elements (generally transistors) and the centre horizontal bar is the location of the motor being driven.

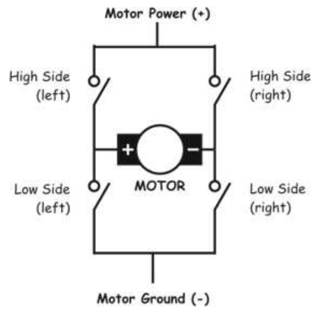


Figure 2.14: The general arrangement of a H-Bridge

(McManis 2003)

When in operation, the switches (or transistors) at each corner of the bridge operate in pairs to provide current flow through the motor in both directions. If you activate the ‘high left’ and ‘low right’ switches, then current flows from high left, through the motor, to low right and the motor turns. If ‘high right’ and ‘low left’ are activated, then the current flow is in the opposite direction and the motor direction is reversed (McManis 2003)

There are four main useful states of an H-Bridge, and they are shown in table 2.1.

One of the most useful parts of an H-bridge is that the switches on it can be activated by any number of methods. It is possible to simply switch them on using some toggle switches, however the main purpose of using a H-bridge is to provide an interface between a computer and an electric motor and hence toggle switches would not be of much use.

To control an H-bridge with a computer, you require a set of four field effect transistor’s

High Left	Low Left	High Right	Low Right	Operation Description
On	Off	Off	On	Motor rotates Clockwise
Off	On	On	Off	Motor rotates counter-clockwise
On	Off	On	Off	‘Brakes’ applied
Off	On	Off	On	‘Brakes’ applied

Table 2.1: Different operational states of a H-Bridge

(McManis 2003)

(FET’s). FET’s, unlike normal transistors, do not require any input current to activate, and hence operate correctly when used in conjunction with a parallel port of a computer. This is so because parallel ports indicate data by setting a high voltage on the output pin and a voltage is all that is required by a FET.

Using a computer to operate an H-Bridge leads to many advantages over normal toggle switches; the main being pulse-width modulation. In pulse-width modulation, the mark to space ratio of the output voltage is modified to switch the motor on and off at different rates thereby effectively controlling the speed of the motor.

Chapter 3

Data Collection and Testing Methods

3.1 Introduction

This section describes the methodology used, and the apparatus required for accuracy testing of the GPS guidance system. A clear view of the sky and a good amount of room will be required for small-scale testing and a machine (preferably a tractor) and a large area of land will be required for on-farm testing. Images of the testing set up and results will be required as well as drift and displacement data logging to calculate the accuracy of the trajectories produced by the guidance system.

3.2 Small-scale testing

Small-scale testing consists of mounting the two receivers stationary in a clear area so that they have good satellite coverage and logging the drift of the modelled position. This method of testing also includes dynamic testing where the mobile receiver is moved around over a known area and the position logged.

3.2.1 Equipment

The equipment required for small-scale testing is as follows:

- GPS receivers
- PC or Laptop capable of communicating via RS232 port to two GPS receivers
- USB to Serial converters to increase number of available serial ports on PC and Laptop
- Tape measure
- Mounting points (eg. Star Pickets)

Images depicting the configuration of the small-scale testing equipment are shown in figure 3.1.

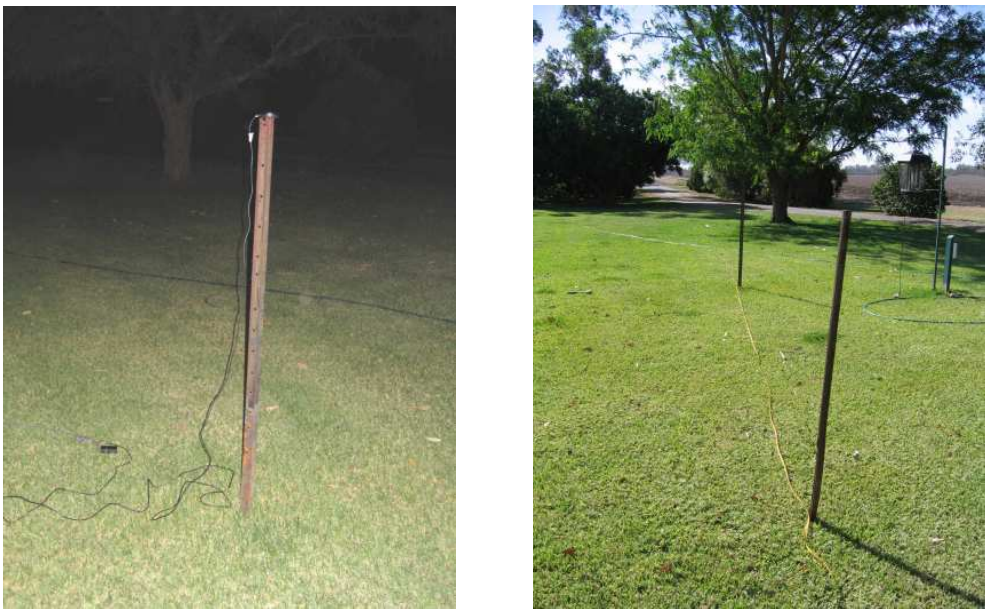

(a) Receiver mounted on star picket (b) Star pickets used to log receiver position over known distance

Figure 3.1: Data collection setup for small-scale testing

3.2.2 Testing and Data Collection Methods

The methodology to test the small-scale accuracy of the GPS system and collect the data is as follows:

- Ensure that both GPS receivers are turned on a transmitting position data
- Allow each transmitter approximately 15 minutes to fully download a new almanac from the satellites. This provides the greatest accuracy in the resulting tests
- Start the GPS software and hit the 'Start Logging' button to start logging the position of the mobile receiver to a CSV file

Once the system is set up as indicated above, testing of the accuracy of the positioning system commenced.

Initially, the receivers were left in the same spot so that the system recorded the modelled position of the mobile receiver. By recording position data over a period of five to 10 minutes, the drift of the system can be inferred. The drift can be specified as the distance the logged data points are away from the zero drift point as a value of $\pm\ x$ centimetres. This data can then be compared to the required differential and RTK GPS accuracy of $\pm$2cm.

Once the drift error has been calculated, the next accuracy to test is the ability to accurately log movement of the mobile receiver. This was tested using the method shown in figure 3.1 by measuring the distance between the two star pickets and then moving the mobile receiver between the two. A comparison of the known distance to the GPS distance provided an idea of the accuracy of the system.

The accuracy of the guidance part of the system was also tested using the previous apparatus. With the guidance heading set in the direction between the two star pickets, the accuracy of the guidance system can be judged by moving the receiver off line while walking between the two. If the guidance system tells you to move right when you are too far left, then the system operation is correct. If, however, the system tells you to turn right, when you are already to the right, then the system has an error. If you end up in the exact required position when the test is completed then the accuracy of the system can be judged as good.

3.3 On-Farm testing

On-farm testing consists of mounting the mobile GPS receiver on a piece of farm machinery and mounting the fixed GPS receiver on some form of stationary base station. A large amount of clear land is required for testing as this method of testing requires that the machine be driven up and down a ‘paddock’ to gain an understanding of the accuracy of the guidance, as well as its ability to offset the path over a swath width.

On-farm testing of the GPS 18-5 guidance system was hampered by problems with transmitting equipment. The Zigbee transmitters used with the GPS 35 system could transmit data with no errors at 9600 baud, however when the baud rate was increased to 38400 for the GPS 18-5’s, errors occurred in the transmission. Due to the importers of Zigbee transmitter equipment not having any pre-built RF modules in the country, a rather archaic method of transmission was used: a 100m long cable. This cable limited the range of testing, however it was enough to gain an understanding of the accuracy of the system.

3.3.1 Equipment

The equipment required for on-farm testing is as follows:

- GPS receivers
- PC or Laptop capable of communicating via RS232 port to two GPS receivers
- USB to Serial converters to increase number of available serial ports on PC and Laptop
- Large (50m) tape measure

- Base station mounting point
- Transmitters (between base station and machine)

  Zibgee transmitters (GPS 35 system)

  100m cable
- Farm machinery

  Tractor

  Implement (Cultivator, single tyne etc.)

As the on-farm testing configurations differed from the GPS 35 guidance system to the GPS 18-5 system, images depicting the configuration for testing of both systems will be shown in figure 3.2.

3.3.2 Testing and Data Collection Methods

The methodology to test the on-farm accuracy of both GPS system's and collect the data is as follows:

- Ensure that both GPS receivers are turned on a transmitting position data
- Allow each transmitter approximately 15 minutes to fully download a new almanac from the satellites. This provides the greatest accuracy in the resulting tests
- Start the GPS software and hit the 'Start Logging' button to record the drift of the system when stationary
- Drive to start point and store in software
- Drive to finish point and store in software
- Position machine at start point, activate guidance and drive machine along path while logging position

- Perform accuracy tests by driving up and back along the same path and measuring different in position
- Perform accuracy tests by driving up paddock, turning around over swath width and driving back parallel to original path and measuring distance between swaths
- Test Ability to offset over various swath widths

Even though the drift was obtained from the small-scale tests, it is always good to know whether there is any difference when the system is installed on a tractor. Interference from devices such as CB radio's and other items mounted on the tractor may affect the accuracy of the system, and such errors must be quantified.

To begin accuracy testing, the system must be told the heading which it is to follow. Once this heading has been established, guidance can occur.

The method of guidance implemented in testing the software was manually turning the steering wheel while obeying an arrow on the laptop screen.

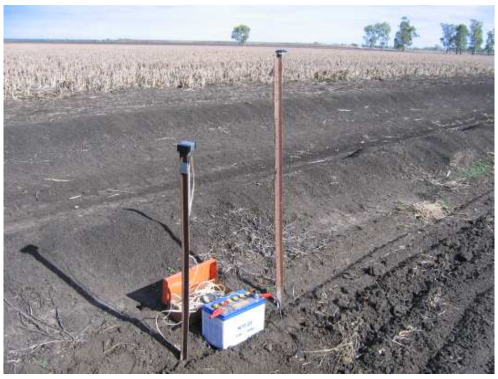
By driving the machine along the path, and then returning over the exact same path, the accuracy of the guidance system can be found. If the tyre tracks overlap perfectly, then the system is guiding accurately, however if there is an offset anywhere above two to three centimetres, the system accuracy is not good enough to be marketed as a ±2cm system.

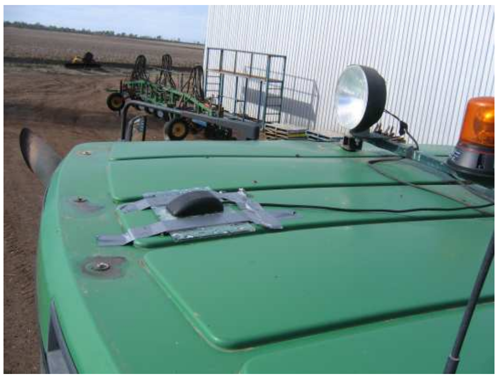
The same critique occurs for the testing of the swath width offset software. If the distance that the machine offsets is accurate to the required distance then the accuracy is good, if not then there is still work to be done.

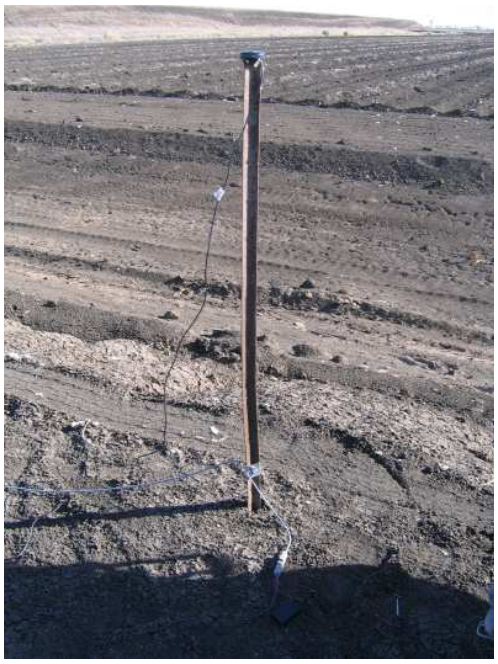
Throughout the agricultural industry, there are many different swath widths used. The main width used on the Darling Downs is 8.1m, however 12m wide machinery is not uncommon. Also, spraying is generally done over widths from 24m to 36m hence the software must be able to accurately accommodate these requirements.

3.4 Conclusion

Large amounts of data was recorded using the logging ability of the GPS software. This data contains drift information and the trajectories followed by the machinery as it was guided using both guidance systems. The data will be analysed in the coming chapters and the accuracy of the guidance systems will be quantified.


(a) Fixed GPS 35 receiver mounted on base station with Zigbee Transmitter (left)


(b) Mobile GPS 35 receiver mounted on tractor


(c) Fixed GPS 18-5 receiver mounted on base station

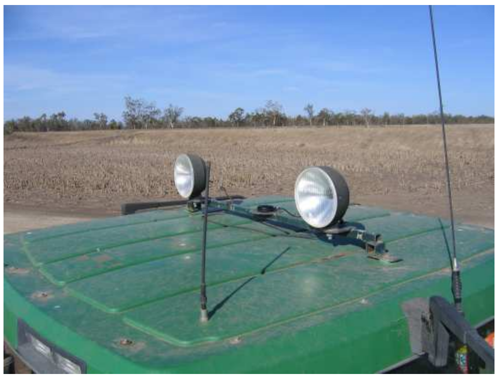

(d) Mobile GPS 18-5 receiver mounted on tractor

Figure 3.2: Data collection setup for on-farm testing of GPS system

Chapter 4

Review of GPS35 Receiver Software

4.1 Introduction

The initial guidance system consisted of two Garmin GPS35 receivers. The two were linked via extension telephone cable so that you could set the fixed receiver a fair distance from the mobile receiver, and still move the mobile receiver over a large range. While this system was good for testing - you could use it in your backyard and 'simulate' the operation - it lacked the ability to be split up for machine guidance testing. This chapter will review the operation and accuracy of the GPS 35 guidance system.

4.2 Data transmission

To set the system up for machine guidance, it required the implementation of a set of transceivers that could transmit and receive the data from the fixed receiver over considerable distance. Initially, the transceivers used for this were Proxim Proxlink's, however testing with these apparatus did not get off the ground. The Proxim transceivers did not want to operate and it was not until some time later (after a new communication method had been employed) that the issue was discovered. The Proxim's required between 6V and 9V DC for operation: it was discovered that if the input voltage was anything remotely greater than the maximum, then the transceivers would simply shut off. It was concluded that the 9V supplied by a regulator was higher than 9V. This problem was fixed by lowering the input voltage to 7.5V using a voltage regulator with a switchable output voltage.

When it was discovered that the Proxlink's were not working, two XBee Pro transceivers were sourced from the NCEA. They were programmed to run at 9600 baud and connected to a PCB. The PCB was then connected to the receiver (or computer) via an RS232 connection and to a 12V supply. The PCB channeled the data to the XBee Pro and also provided it with a regulated supply voltage of approximately 3.3V.

An image of the basic configuration of the XBee Pro system is showing in figure 4.1.

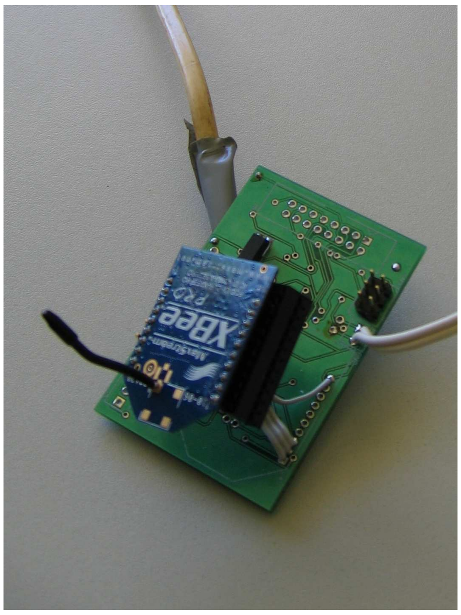


Figure 4.1: XBee Pro Zigbee transmitter with PCB

4.3 Software

As described in chapter 3, the position of the mobile receiver was to be logged throughout all the tests so that the accuracy of the system could be quantified. The initial GPS 35 guidance software that was supplied did not have any logging code and hence the code had to be added.

The code that was added to provide the ability to log the mobile receiver is based upon the same principal as the code described in section 6.3.1 except that the data being written to file consisted of position data rather than almanac data.

Once the logging ability was added, it was the required that the accuracy of the system be tested in both small-scale and on-farm situations, however it became clear that another piece of code was required to be added before any sort of accuracy testing could begin.

The software had the ability to guide the user along a straight line for however long was required however, it did not have the ability to turn the machine around over a swath width. The code to do this was added to the software (see section 6.7.8) and testing of the software commenced.

4.4 Testing of System

The GPS 35 guidance system was tested to the methodology set out in chapter 3 and the data analysed as follows.

4.4.1 Small-scale accuracy

The small-scale testing of the GPS 35 system involved placing the two receivers approximately 20m apart on a flat piece of ground and then logging the drift of the system.

For approximately the first eight minutes of operation, the receivers were unable to lock on to any satellites. This was because the almanac inside the receivers was out of date by at least a month, and it takes 15 minutes to download a new one (but only around eight minutes until enough data has been updated to lock on to some satellites).

Once the system had enough satellites to gain a position fix, the drift of the system was logged over a period of one minute. The results from this analysis are shown in figure 4.2.

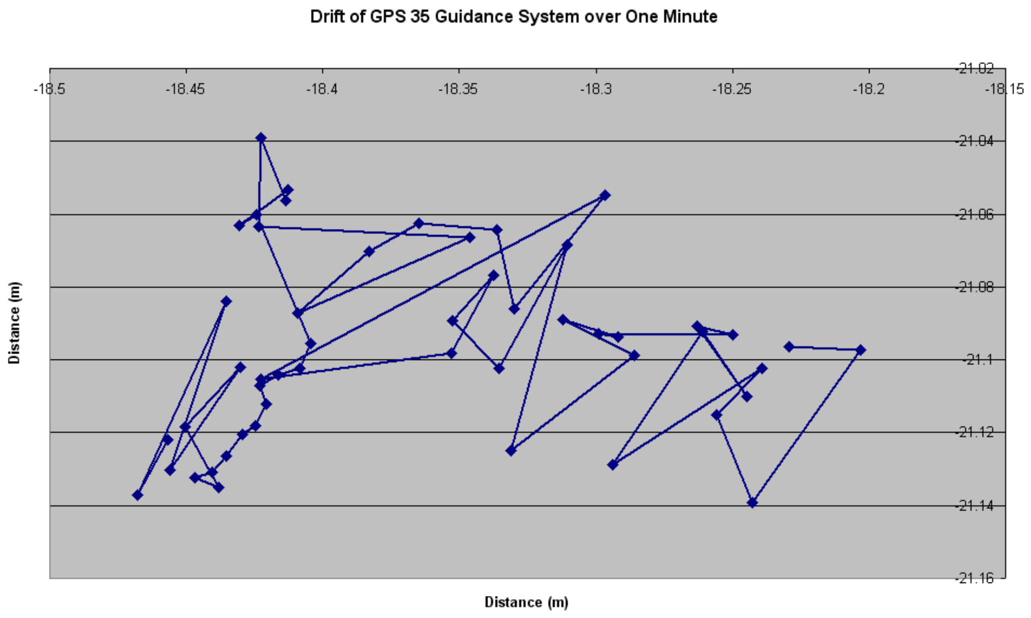


Figure 4.2: Drift of GPS 35 system

From this figure, we can infer that the accuracy of the GPS 35 system is far from what is required. The system has drifted approximately 20cm in the x-direction and 10cm in the y-direction; this is nowhere near the required accuracy value of ±2cm.

Extra graphs displaying drift information for the GPS 35 system can be viewed in appendix D.2.

4.4.2 On-farm accuracy

The on-farm accuracy of the GPS 35 system involved setting up the XBee Pro transmitters and mounting the mobile receiver on the roof of the cab of a John Deere 8300 tractor, and the fixed receiver on the base station as shown in figure 3.2.

The following image depicts the operation of the GPS 35 guidance system while testing was underway. Steering direction was provided by the laptop, which was sitting on the operators lap. This system was not optimal but for the purposes of testing the outgoing system, it was judged to be good enough.

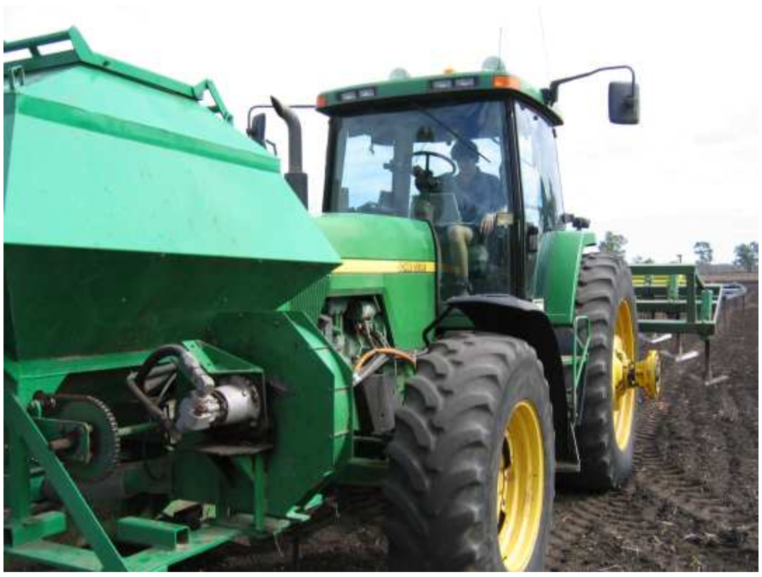


Figure 4.3: Operation of guidance system during testing

One the almanac in the GPS receivers had been fully updated, it was time to test the accuracy of the guidance system. An initial idea for testing whether the system could drive from one point to another, and then back again accurately is shown in figure 4.4.

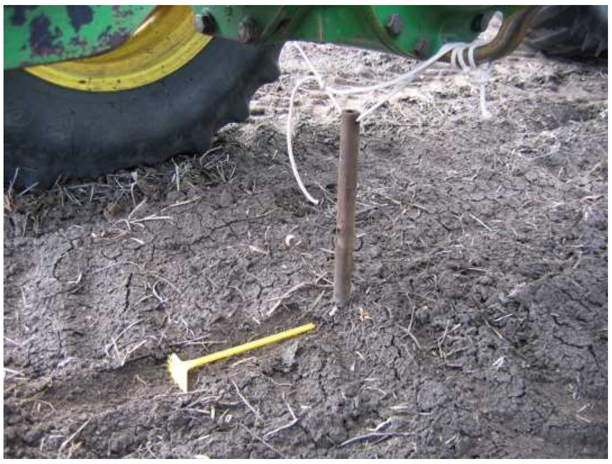


Figure 4.4: Locating point for guidance accuracy testing

By marking a point at the start of the run and the end, and dragging a line up the paddock using the steel rod shown, the accuracy of the system would be easily quantified. This idea turned out to be rather ineffective as it was extremely hard to realign the tractor with the markers after setting up the heading, and then the second marker could not be found when the end of the run was reached. For this reason, the accuracy of the system was judged from the data logged during on-farm testing.

The data logged during on-farm testing provided a good insight to the accuracy of the system. Most of the data logging was performed on the middle island of an (unfortunately) dry ring tank. This location provided a large flat area with uninterrupted views of the sky.

Testing that was conducted included the ability for the system to drive straight, and its ability to offset a swath width. The trajectory of the test rig while executing this test is shown in figure 4.5.

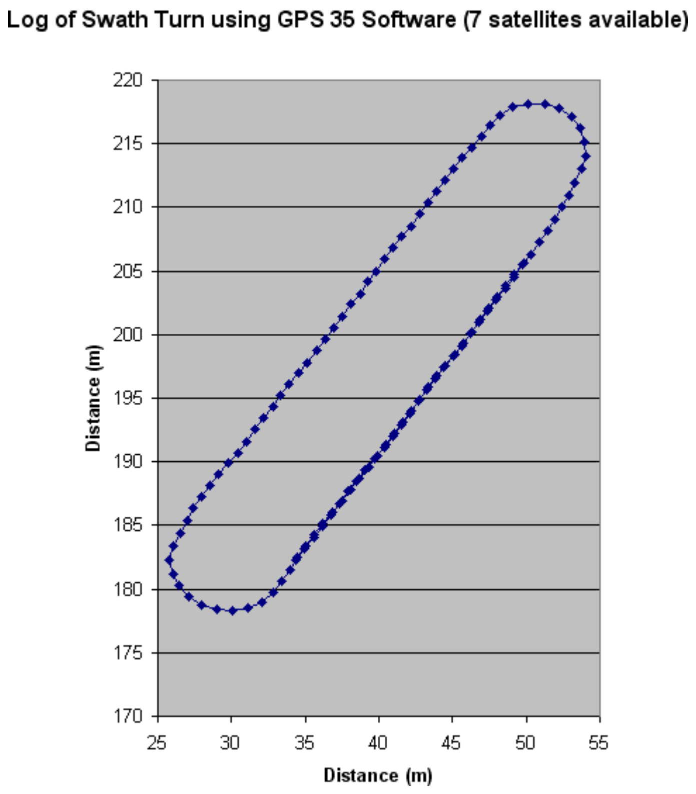


Figure 4.5: Log of trajectory of test rig

The distance between the two trajectories shown can be found by taking points on both the outward and return trajectories that are at approximately perpendicular to each other and calculating the distance between them; the distance between the trajectories should be 8m. The ability for the system to offset over multiple swath widths was also tested. A set of points and calculations of the swath width are shown in table 4.1.

Outward Trajectory		Return Trajectory		Swath Width
x	y	x	y	(m)
34.532	196.967	40.371	191.162	8.233
40.354	205.900	46.310	200.245	8.128

Table 4.1: Table of swath widths along trajectory

This data shows that the accuracy of the swath system in the GPS 35 software is fairly close to what is required. The software calculations of position should be accurate which leads to the conclusion that the errors in distance come from the rough ground and the operator.

Another problem discovered with the GPS 35 system while testing is its susceptibility to losing satellite lock while operating. To resume from a loss, the system 'guesses' the next guidance point based upon the direction being travelled at the last point of satellite lock. If lock is lost when the machine is travelling in a straight line, then this method works well, however if the machine is in the process of a turn, the software will assume the machine is travelling straight, and hence part of the turn will be lost.

A trajectory showing areas where satellite lock was lost is shown in figure 4.6.

Extra graphs and data indicating the accuracy of the GPS 35 system can be found in appendix D.3.

4.5 Usability of Guidance Software

Speaking from experience, the GPS 35 guidance system is lacking in its ability to make life easy for the operator.

Before the swath offset software was added, the method of picking your swaths down

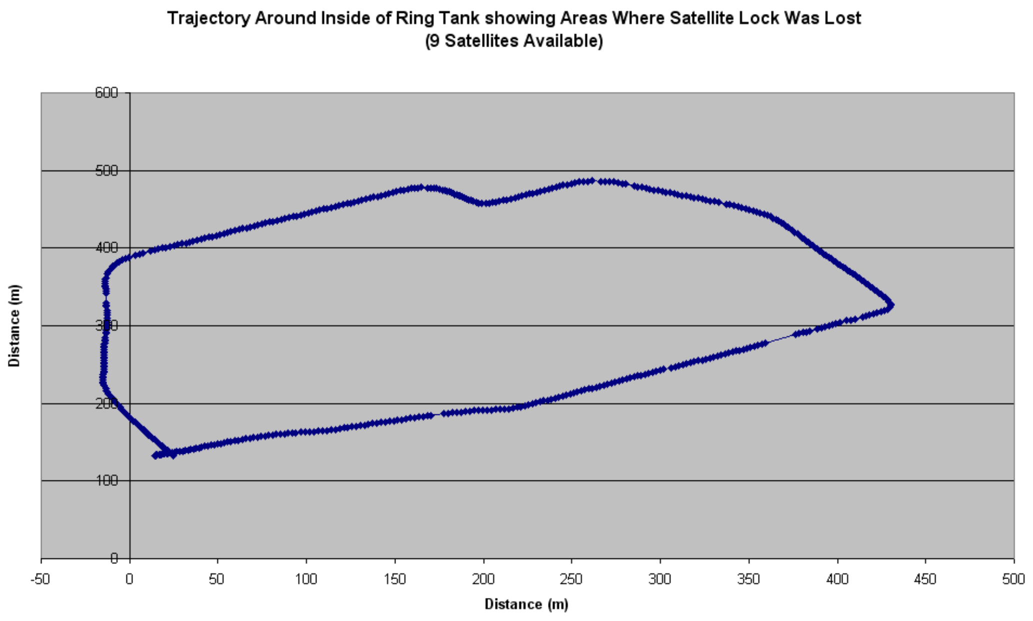


Figure 4.6: A trajectory where satellite lock was lost and then regained

the paddock required the operator to get out of the machine and mark each one with an object such as a Coke can. Although never undertaken, this method would have been extremely tedious and time consuming, and even then the accuracy of the trajectory could not be fully guaranteed due to the drift of the receiver and the driver's ability to place the marker in the centre of the machine.

Also, the lack of automatic guidance required the operator to guide the machine from a white line on the computer screen. This white line is easy enough to see, however it is not easy to follow. As the system only calculates its position once every second, the guide line would jump up to 50cm (from 'Turn Left' to 'Turn Right') every time the software calculated current position. This frequent switching made it almost impossible to gain a good understanding of the direction the system wanted to move in and caused rather wobbly lines up the paddock.

4.6 Conclusions

The GPS 35 guidance system is certainly the basis for better things to come. In its basic form it has the ability to guide a machine along a straight line with good accuracy, but it is not the best. Once the swath offset was added, the system's appeal certainly increased, but it is simply not accurate enough to have any sort of use in the agricultural field.

The problem certainly isn't in the software; it can only work with what it is given, and that is the data from the GPS 35 receivers. It is because of this that there are high hopes for the merits of the new system based upon GPS 18-5 receivers whose ability to transmit position data at 5Hz should greatly increase the accuracy of the system.

Chapter 5

Development of Autonomous Steering Unit

5.1 Introduction

An integral part of any GPS guidance system its ability to provide hands-free guidance. Hands-free guidance comes in many different incarnations being:

- Hydraulic Actuators; and
- Steering wheel mounted drive motors.

Both of these methods have a viable place in agricultural guidance, however they are costly and usually complicated to install. The hydraulic actuator guidance method cannot be easily moved from machine to machine and hence, the cost of the system increases rapidly with the number of machines that require guidance.

This section of the report will deal with the conceptual development of a new idea for the hands-free control of a piece of farm machinery.

5.2 Design Requirements

A steering unit needed a design that was not only cheap and effective, but also small, efficient and could also be easily moved from machine to machine. The unit also required the ability for guidance to be controlled by both the unit and the steering wheel, and a form of safety control where the operator would be able to override the unit and steer the machine around obstacles.

The unit is to be powered by an electric motor and that in turn will be operated by a H-Bridge interfaced with the guidance computer.

5.3 Conceptual Design

5.3.1 Differential Type Planetary Gearbox

The first idea for the design of the gearbox involved two planetary gear sets connected so that a differential style power transmission and large gear reduction occurred. The design was such that the gearbox would be inserted between the steering wheel and the steering column of the machine.

A planetary gearbox consists of three main components: the ring gear, the planet (or orbital) gears and the sun gear. The configuration of these components is shown in figure 5.1.

If you place an input on the sun gear and hold the ring gear stationary, you generate an output via the rotation of the planet gears about the sun. If you connect the planet gears together and use them as the output, the system operates as a gear reducer. The reduction can be modified by adjusting the size of the sun and planet gears.

The initial design concept is shown in figure 5.2.

If two planetary systems are placed on top of each other and the planet gears joined together and held stationary (as shown in figure 5.2), an input at the top (through the

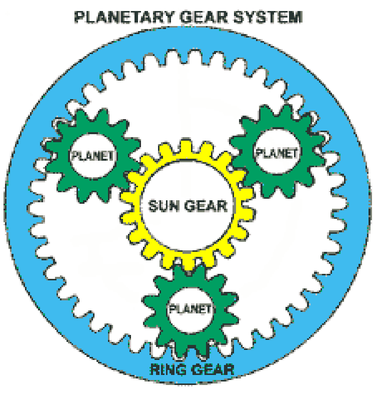


Figure 5.1: Configuration of a planetary gearbox

(Ofria 2000)

green shaft into the red gear) would constitute an output of the same magnitude and direction at the bottom (through the blue gear and out of the green shaft). If we now change the size of the planet gears so that the red gear depicts a 23 tooth gear and the blue gear depicts a 25 tooth gear, we can lower the ratio of input to output, but what does this mean? Well, nothing in the current configuration, except that when the steering wheel turns (attached to the top shaft), the steering output shaft will turn slightly slower.

If we were to now place a motor drive on to the grey 'planet lock' to spin the planet gears about the sun gears, we can start to develop a differential guidance system. If you hold the input shaft (steering wheel) stationary, and let the motor spin the planet gears about the sun gears, the output of the system will be only a few teeth worth of rotation on the bottom sun gear.

The reason for this reduction is best explained if we once again assume the sun and planet gears are the same size.

As the top planet gears rotate about their sun, they spin at a certain rate and hence the bottom planet gears also rotate at this rate. As all the gears are the same size, we can infer that for such a configuration, there will be no output from the bottom sun gear. Now, if we change the size of the gears, we can see that the top planet gears will be spinning slightly slower than in the previous configuration, as with the bottom planet gears. Due to this gear reduction, the bottom planet gears will 'drag' the bottom sun

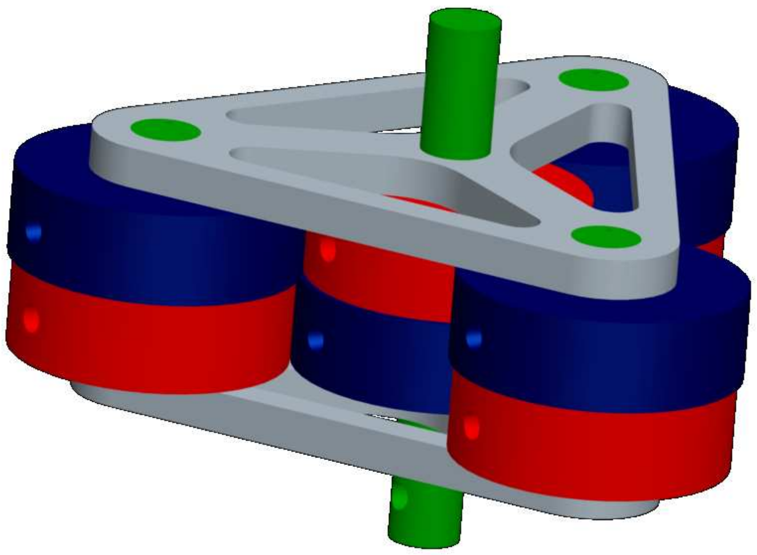


Figure 5.2: Initial design with two planetary gearboxes mounted together

gear around as they are no longer spinning fast enough to keep it stationary.

This system provides an extremely large gear reduction and corresponding increase in torque and as such, should not require a very large drive motor. It also accommodates the need for a user override with our without the motor running.

If the motor is not running, then the planet gears are held stationary by the motor and the planet lock and hence the steering input is transferred directly into the steering system. If the motor is running the planet gears clockwise and the steering wheel is turned clockwise at the same rate, it will cause the planet gears to stop moving and thus the lower sun gear will be dragged around. If the steering wheel is turned faster than the rotation of the planet gears, then the output shaft will be turned faster. If the steering wheel is turned in the opposite direction to the planet gears, the planet gears speed up and the output shaft also speeds up.

Once the basic concept had been developed, some gear sizes needed to be specified. The chosen gear sizes and a schematic are shown below, and were sourced from a Farnell Online Catalogue.

- Gear Type - Farnell Steel Spur Gear

- Gear Specs - Milled Teeth, Pressure angle 20^o
- Red Gear Specs

  No. of Teeth: 23

  Overall Diameter (d_a): 50mm

  Pitch Circle Diameter (d): 46mm

  Bore (B): 12mm

  Torque Rating: 2.6Nm

  Mass: 187g
- Blue Gear Specs

  No. of Teeth: 25

  Overall Diameter (d_a): 54mm

  Pitch Circle Diameter (d): 50mm

  Bore (B): 12mm

  Torque Rating: 3.1Nm

  Mass: 225g

(Farnell In One 2006)

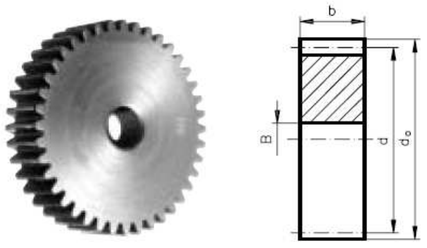


Figure 5.3: Schematic of Chosen Gear

(Farnell In One 2006)

Now that the basic idea has been developed, a method of connecting the gearbox to the steering column of a tractor must be devised. As the gearbox is meant to fit between the steering column and the steering wheel, the connection between the two must first be identified and is shown in figure 5.4.

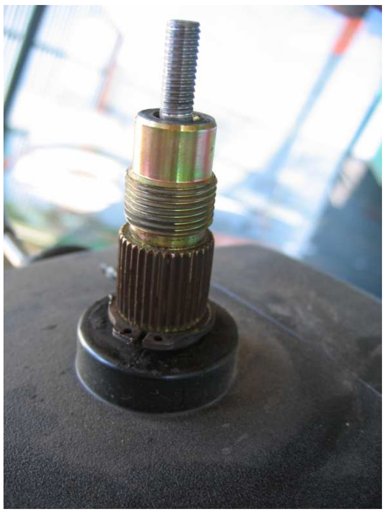


Figure 5.4: Steering column mounting point for gearbox

Personal Image

The mount consists of a spline at the base which the steering wheel slides on to, some threaded shaft in the middle, and a threaded rod at the top and is approx 69mm long. Mounting the gearbox to the system would be easy except for the threaded shaft.

The threaded shaft is the locking mechanism for the adjustment of the length of the steering column. For this reason, it must be easily accessible as the comfortable length of a steering column changes for each operator. For this reason, the following interface (shown in figure 5.5) was designed.

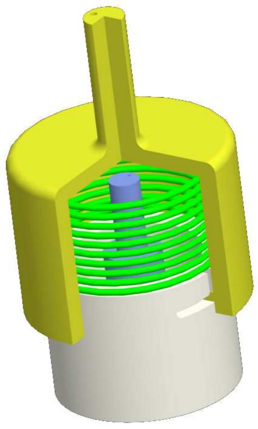


Figure 5.5: Interface between steering wheel mount and gearbox

The interface consists of a cup that slides over the spline on the mount, another cup that locks down into the bottom cup, and a spring to provide stability of the locking mechanism. The spring would have to be quite strong to minimise any play in the connection and also support the mass of the steering wheel, gearbox and operator. In this configuration, if the column length needed to be changed, the user could unlatch the two cups and remove the drive system. The conceptual design is shown in figure 5.6.

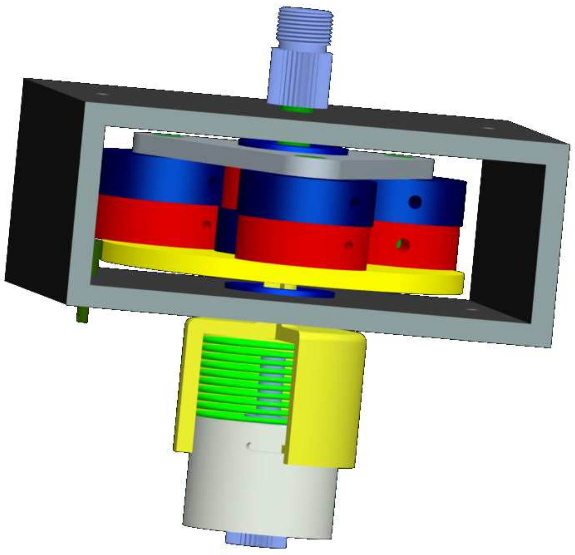


Figure 5.6: Conceptual design of gearbox

The usability of this design is somewhat limited however, as the total length of the gearbox and mounting points is approximately 200mm. This extra length between the steering column and steering wheel could make driving the machine uncomfortable as the steering column only has approximately 250mm of travel.

For this reason, a system to offset the gearbox from the steering column and provide room for the column height adjustment was developed and the design is shown in figure 5.7.

The offset design employs the use of two separate interface sections to connect the steering column and the steering wheel. When mounting the gearbox, the shaft on the steering column (light blue at bottom) slides up into the first interface boss mounted in the base of the gearbox casing. The shaft to adjust the steering column length slides into an extension shaft (shown in red) which then passes through the second interface.

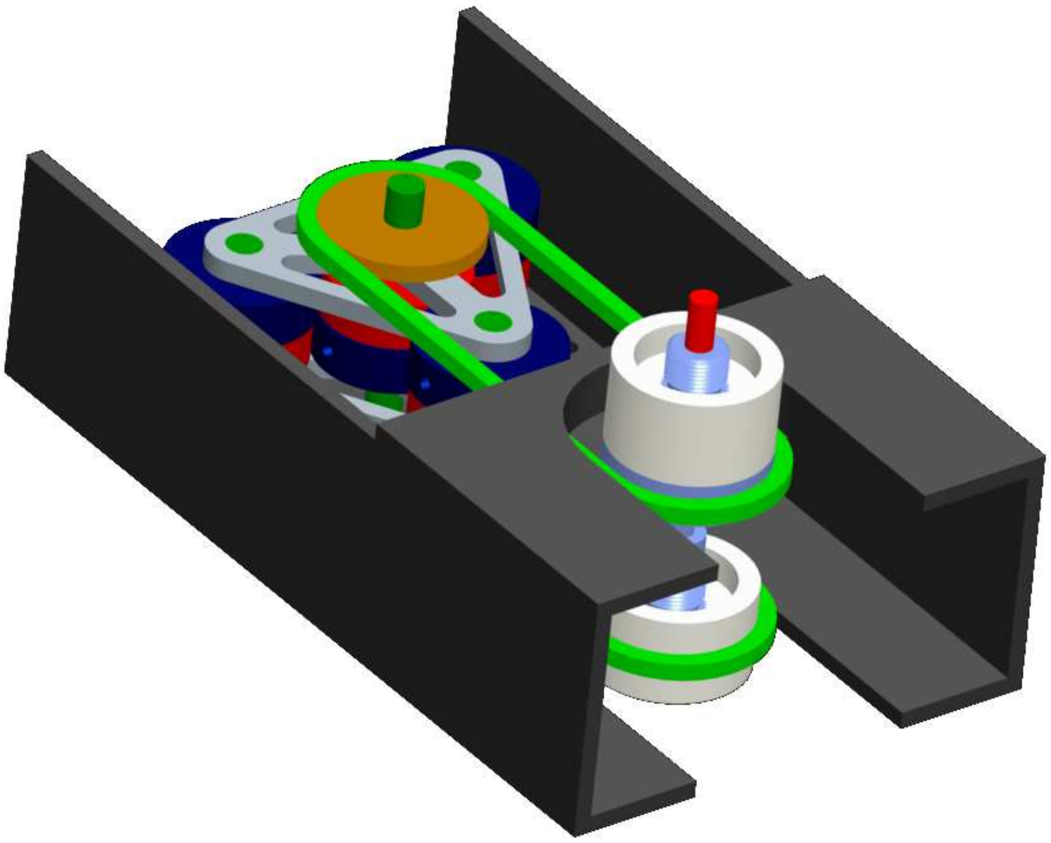


Figure 5.7: Conceptual design of offset system

Power is transferred from the output of the gearbox to the boss via the green chain.

The steering wheel's boss (in white at top) slides down over a splined shaft (light blue at top) that is connected to the input of the gearbox via the green chain.

By implementing this method of connection, the added length to the steering column is only 81mm and thanks to the extension shaft, no removal of the gearbox is required to change the length of the steering column.

Another view of the design in figure 5.7 can be found in appendix B.2.

As the offset gearbox design is probably more useful, conceptual detail drawings for specific items have been placed in appendix B.3.

5.4 Other Ideas

The conceptual design process is never really fully completed. New ideas are always developing as time goes by and situations change. The previously discussed ideas are what were considered to the greatest extent, however they were not the only designs thought of.

Another possible, less complicated design is shown in figure 5.8.

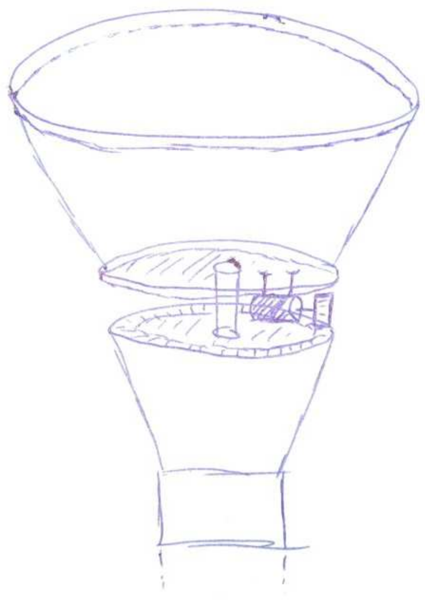


Figure 5.8: Less complicated steering design

In this design, a steering wheel is cut in half, the two halves are joined together via a central shaft and a ring gear is placed on the top of the bottom half. A motor is mounted to the top half of the steering wheel and its output is placed on the ring gear. The operation of this system is quite simple. When the steering wheel is held stationary, the motor is stationary and the output of the motor turns the bottom half of the steering wheel, thereby steering the machine. If the operator wants to steer the machine manually, the motor locks up and the entire assembly moves as one. If the guidance needs to be overridden in case of an emergency, the operator should be able to turn the steering wheel fast enough to counteract the steering of the motor.

5.5 Conclusions

Although a gearbox was never actually build, the conceptual designs are a step in the right direction. There are many different ways that a simple, cheap and user-friendly system could be developed and the presented ideas are only a few of them. Hopefully one of the designs does become a reality, but that may not occur for some time.

Chapter 6

Development of GPS18-5 Compatible Software

6.1 Introduction

The software supplied by Prof. John Billingsley for use with the GPS 18-5 receivers was already operational and was able plot the movement of the mobile receiver on a screen, and log its movements to a CSV file. Once two GPS 18-5 receivers had been acquired from the NCEA, testing of the system could begin, however there were a few issues within the software that needed fixing before any testing could begin.

6.2 Baud rate and parity errors

Before any sort of data collection could occur, the system had to talk to each of the receivers and change their baud rates from 9600 bits per second to 38400 bits per second. The code to do this was already in the software, however it was not operating correctly and the baud rate was not changing. It was discovered that the method used to change the baud rate required that the receivers be in Garmin mode and at a known baud rate. Garmin mode is a proprietary binary phase output mode which contains position and velocity information. The other receiver mode is NMEA mode and this mode transmits latitude, longitude and many other selectable sentences. The only NMEA data that is of consequence is the almanac sentence, and this will be discussed later.

6.2.1 Baud rate

The GPS 18-5's can be configured to start in Garmin mode at a baud rate of 9600 bits per second when first turned on. This baud rate is the main reason that trouble occurred when first trying to change the baud rate. If the serial port is not running at the same baud rate as the receiver, then the data being sent becomes corrupted and no control of the receiver can occur. This same problem occurs when trying to receive data from the receiver.

Once the baud rates of the GPS 18-5's and the computer had been matched, it was possible to send the receivers the commands necessary to change the baud rate.

The baud rate of the GPS 18-5's needed to be changed because they are capable of transmitting position data at a rate of 5Hz and the standard baud rate of 9600 is only able to transmit enough data for position messages at 1Hz. The baud rate required to transmit this information was 38400 bits per second and the code to change the baud rate is shown in appendix C.2.

6.2.2 Parity

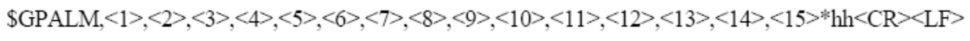
Once the receivers had been configured to talk to the software and data was coming through, it became clear there was another problem in the system. The software was not recognising the data that was coming in from the receivers and after some searching, the problem was found to be a parity error, however the location of the error was unknown. It was by fluke that the parity error was discovered to be that not all of the data being transmitted by the receiver was being 'taken' by the software. This caused unused data to be left sitting in the software until it was read in the next time the software looped causing the previous data to combine with the incoming data. The remedy was an easy fix, and simply required the addition of three new variables to 'dump' the excess information into.

The three new variables were added to the subfunction `getposrecord` (shown in appendix C.3) and are shown below:

- Height above sea level;
- UTC leap time; and
- Garmin days.

6.3 Receiving NMEA 0183 Almanac Sentence

The almanac data is transmitted from the GPS 18-5 receiver in NMEA 0183 sentences. To acquire this information from the receiver, it is a simple matter of stopping the receiver from sending position data, and then transmitting the sentence "$GPALM" to the receiver. The receiver then transmits the almanac data for each satellite in the system (normally 32). The NMEA 0183 sentence has the following structure:



$GPALM,<1>,<2>,<3>,<4>,<5>,<6>,<7>,<8>,<9>,<10>,<11>,<12>,<13>,<14>,<15>*hh<CR><LF>

Figure 6.1: NMEA Sentence configuration

(Garmin 2005)

Each of these sections (< ... >) corresponds to a different piece of information pertaining to the orbit of a satellite. This information is described below.

- < 1 >: The total number of almanac sentences to be transmitted by the GPS receiver.
- < 2 >: The number of the current almanac sentence being downloaded.
- < 3 >: The PRN (pseudorandom number) - different for each satellite; indicates which PRN, and hence, satellite the almanac data represents;

- $< 4 >$: The GPS week number - can be used to set the current date in a GPS receiver. This information is used by the receiver to work out which satellites may be visible at that point in time;
- $< 5 >$: SV Health - Zero if satellite is 'healthy';
- $< 6 >$: Eccentricity of Orbit - Measure of how much orbit shape deviates from a circle;
- $< 7 >$: Almanac Reference Time - also known as "Time of Applicability"; the number of seconds in the orbit when the almanac was generated;
- $< 8 >$: Inclination Angle - The angle at which the satellite orbit meets the equator;
- $< 9 >$: Rate of Right Ascension - rate of change in the measurement of the angle of right ascension;
- $< 10 >$: Root of semi-major axis - the measurement from the center of the orbit to either the point of apogee or perigee;
- $< 11 >$: Omega, Argument of Perigee - angular measurement along the orbit measured from the ascending node to the point of perigee in the direction of the satellite's motion;
- $< 12 >$: Longitude of Ascension Node - angular measurement from the vernal equinox;
- $< 13 >$: Mean Anomaly - angle travelled past the longitude of ascension node;
- $< 14 >$: Af0 clock parameter - satellite clock bias in seconds; and
- $< 15 >$: Af1 clock parameter - satellite clock bias in seconds per second.

(T.S. Kelso 2005)

The original GPS35 software has been modified by Professor John Billingsley to talk to the GPS 18-5 receivers at 5Hz transmission rate. However, this software is a basic skeleton of what is required for guidance. The software does not contain any of the guidance software, it simply displays the position of the current mobile receiver on the screen and allows you to log that data.

When the current software asks the receiver for the NMEA data, it is transmitted to the program, and displayed in the 'DEBUG' window of Visual Basic. It is required that this information be stored in a text file so that it can be interpreted to YUMA format by the software.

6.3.1 Downloading NMEA data to a text file

To download the data to a text file a file must first be opened within the software. To do this, a storage location and name for the file must first be created. This file must then be opened and this is shown in figure 6.2:

```
logfile = "c:\ " + "NMEAalm" + ".txt"
Open logfile For Output As 1
```

Figure 6.2: Code to generate file name and location

The variable `logfile` is a string the contains the filename and location of the file to be created. The program uses this data to generate a file called NMEAalm.txt in the C:\ directory.

Once the file has been opened, the almanac data coming in from the receiver must be written to it. This is done by capturing the data coming in through the COM port and storing each string of data as a new line in the file. The data comes in from the ComMob object on the Board form. The code to do this is shown in figure 6.3.

```
Write #1, Board.ComMob.Input;
```

Figure 6.3: Code to store almanac data to text file

The function "write" will write data to the location "#1" which in this case is a pointer towards the file previously opened "As 1".

Finally, the file must be closed and the code for this is shown in figure 6.4.

```
Close 1
```

Figure 6.4: Code to close data file

The function “Close” closes the file pointed to by “1”, which in this case is the almanac data file.

The full program listing to receiver the almanac data and save it in a file is shown in appendix C.4

Now that the text file containing all the NMEA data for the almanac data has been generated, it must be converted to YUMA format.

6.4 Converting NMEA 0183 to YUMA format

Initially, it was decided to build a program to convert the NMEA format to YUMA format outside of the GPS software as it would be easier to implement and once that software was working, it would then be transplanted into the GPS software for further implementation.

A subfunction called `getnmea` was developed to open the NMEA data file generated by the NMEA download software, and analyse it for use in the guidance program. It has four main parts: the variables, the opening of the data file, the analysis and computation of the data file, and the storage of the new data. Each of these parts will be explained in the next sections.

6.4.1 Variables of getnmea

There are a large amount of variables used in the `getnmea` subfunction. The variables that store the NMEA data after processing are stored in an array called ‘`nmeadata`’. They are defined as being of type `nmeadata` and as such are contained within the `nmeadata` variable. The variables and their descriptions are shown in table 6.1.

This set of data can only hold almanac data for one satellite, because even though the

Variable Name	Variable Description
total As Integer	Total number of alm sentences to be transmitted
prnno As Integer	PRN number
weekno As Integer	GPS week number
eccen As Double	Eccentricity of satellite orbit (Hex XXXX)
reftime As Double	Almanac reference time (Hex XX)
inc As Double	Satellite inclination angle (Hex XXXX)
rra As Double	Satellite rate of right ascension
roota As Double	Root of semi major axis of satellite (Hex XXXXXX)
omega As Double	Argument of perigee (Hex XXXXXX)
long As Double	Longitude of ascension node (Hex XXXXXX)
meana As Double	Satellite mean anomaly (Hex XXXXXX)
af01 As Single	Clock parameter (Hex XXX)
af11 As Single	Clock parameter (Hex XXX)

Table 6.1: Variables contained within nmeadata type

variables are an array of `nmeadata`, the variables themselves are not arrays. `nmeadata` must be converted to an array of size 32 to be able to accommodate the data retrieved from the NMEA 0183 data file. This is shown in figure 6.5.

```
Public nd(32) As nmeadata
```

Figure 6.5: Code required to generate array of nmeadata

Other `getnmea` variables are defined as being 'public' and their descriptions are shown in table 6.2.

6.4.2 Opening Data File

The data stored in the file NMEAalm.txt is opened as an input by the program using the code in figure 6.6.

The code opens the text file and provides it with the pointer reference of '1'. This pointer can then be referenced by the program at any point so that the required data

Variable Name	Variable Description
p As Integer	Number of almanac sentences to be analysed
a As String	String storing almanac sentence to be analysed
i As Integer	Current position in almanac sentence
dump as Double	Variable to store unneeded data

Table 6.2: Extra variables used by getnmea

```
Open "C:\NMEAalm.txt" For Input As 1
```

Figure 6.6: Code to open almanac data file

file can be accessed.

6.4.3 Retrieval and Computation of Data from File

Once the data file has been 'opened', the next required process is to move through the file, accessing and analysing the separate bits of almanac data. As there are 32 separate almanac sentences, the subfunction must loop 32 times and through each loop, it must 'look' through each line of data, searching for the comma's that separate the different parts of almanac data as previously shown in figure 6.1.

Each of the pieces of satellite orbit information stored in the NMEA sentence must be multiplied by a scale factor when they are calculated. The data in the NMEA sentence is simply a scale moving from zero to some maximum value. The data obtained by multiplying these scale values by their scale factor is the actual orbit information. The scale factors are shown in table 6.3

The code in figure 6.7 starts the loop to access all of the almanac sentences and using the 'Line Input' command, increments through to the next line of almanac data each time the loop occurs. The loop occurs 32 times which is the number of satellites whose position data is contained in the almanac. The 'Line Input' function stores the current line of data in the file (the almanac sentence) in a string variable called '`a`'. The 'InStr' command takes a string, and scans through it until a certain character, or set of characters, is found. In this case, 'InStr' looks through the string 'a' for a comma.

Orbit Information	Scale Factor
Eccentricity	Multiply by 4.768371582E^{-07} (Unsigned Result)
Almanac Reference Time	Multiply by 2^{12} (Unsigned Result)
Inclination Angle	Multiply by 5.992112458E^{-06} (Signed Result)
Rate of Right Ascension	Multiply by 1.142904749E^{-11} (Signed Result)
Root of Semi Major Axis	Multiply by 4.8828125E^{-04} (Unsigned Result
Argument of Perigee	
Longitude of Ascension Node	Multiply by 3.745070283E^{-07} (Signed Result)
Mean Anomaly	

Table 6.3: Scale Factors for YUMA calculations

Once a comma is found in the string, it returns the position of the comma in variable ‘`i`’.

```
Do Until p = 32
Line Input #1, a
i = InStr(a, ",")
```

Figure 6.7: Code to retrieve almanac sentence from file

An indicative NMEA 0183 sentence, found in an almanac file, is shown in figure 6.8. The first time that ‘InStr’ is executed, its pointer will detect the comma just after ‘$GPALM’. As such, it will return the value 7 which is the position just after the comma.

```
$GPALM,32,1,01,369,00,3430,4E,1E44,FD69,A10D64,B76C08,11C706,C24C30,052,001*05
```

Figure 6.8: Example NMEA 0183 Sentence

Once this comma is found, we know that the information directly after it will be the total number of almanac sentences transmitted. As we know this information already, it is useless and as such we can ‘dump’ it into a variable. The process of acquiring the data from the text file is shown in figure 6.9. The function ‘Val’ evaluates whatever is enclosed within its dominant brackets. In this case, ‘Val’ evaluates the expression ‘`Mid(a, i + 1)`. The function ‘Mid’ returns a specified number of characters from a string. In this case the ‘Mid’ function will return the characters from the position `i + 1` until the next break (in this case a comma) in the string '`a`'.

```
dump = Val(Mid(a, i + 1))
```

Figure 6.9: Code to retrieve data from file

After the program has completed the previous calculation, it must continue through the NMEA sentence until all useful data is retrieved. We previously defined `nd(32) As nmeadata` and we must now find a means of using all 32 'dimensions' of nd. This can be accomplished by using the next piece of information in the sentence: the 'number of current almanac sentence' information.

This information is acquired using a separate subfunction which was developed to make the program easier to read, and manipulate. The subfunction, called '`getalmdata`' is shown in figure 6.10.

```
Function getalmdata(a, i) As Double
i = InStr(i + 1, a, ",")
getalmdata = Val(Mid(a, i + 1))
End Function
```

Figure 6.10: Code for function 'getalmdata'

When '`getalmdata`' is called, two variables (`a and i`) are input to it. '`a`' is the string storing the NMEA data, and '`i`' is the position of the comma defining the previous piece of information. '`getalmdata`' uses 'InStr' to find the location of the next comma by starting looking from `i + 1` (a point past the previous comma). Once the new comma is found, the same combination of 'Val' and 'Mid' functions is used to acquire the data. The data is output by '`getalmdata`' to the main program, and its usage is shown in figure 6.11

```
p = getalmdata(a, i)
```

Figure 6.11: Code to use 'getalmdata'

In this case, the output data is stored in a variable `p`. `p` is used throughout the rest of the program to indicate which sentence is being analysed, and also to reference which 'dimension' of nd is currently being written to. An example of this is shown in figure

6.12. In this case, the program will store the pseudorandom number (PRN) of the satellite to the position `nd(p)`. If the PRN is for satellite 24, then `p` will be 24 and thus the data will write to `nd(24).prnno`. This method allows 32 different values to be stored in the `prnno` section of `nd`.

```
nd(p).prnno = getalmdata(a, i)
```

Figure 6.12: Method of storing to separate dimensions of nd

This same method is used to calculate the GPS week number, and the health of the satellite.

The next value that needs to be read from the sentence is the orbit eccentricity. This value is the first value that is in Hex and thus must be converted. As such, a new subfunction called '`getalmdatah`' was generated. It is very similar to '`getalmdata`' except for one difference in the 'Val' line and this is shown in figure 6.13. The addition of "&h" in front of the 'Mid' function causes the 'Val' command to evaluate the string as a Hex value and return an integer value to the main program.

```
getalmdatah = Val("&h" + Mid(a, i + 1))
```

Figure 6.13: Code to retrieve Hex data from sentence

There is however one small problem with the way that Visual Basic evaluates Hex values. If the number of characters is a multiple of four, it will evaluate the Hex value as though it is signed. If you have, 6 characters, it will evaluate the data as though it is not signed. Some of the NMEA data is signed, and some is not thus extra calculations were required to ensure accurate signing of all data.

The eccentricity data is made up of four Hex characters, however the data itself is not signed. As such, some extra calculations were made to the resulting integer value taken from '`getalmdatah`'. The calculation of the unsigned number, the multiplication of the scale factor, and the storage of the eccentricity is shown in figure 6.14.

The data from '`getalmdatah`' is stored in 'dump' and this variable is then tested to see if the value is less than zero. If it is, then 65536 is added to the value to make it positive, and the data is then multiplied by the scale factor. The data is then stored

```
dump = getalmdatah(a, i)
If dump < 0 Then
nd(p).eccen = (dump + 65536) * 4.768371582E-07
Else
nd(p).eccen = dump * 4.768371582E-07
End If
```

Figure 6.14: Calculations on signed Hex data

in `nd(p).eccen` with `p` identifying the PRN it corresponds to.

Once the program has calculated the eccentricity for a certain satellite, it then moves through and calculates the rest of the orbit data of the parameters and their corresponding scale factors as shown in table 6.3. Once the system has finished analysing the NMEA almanac data, the results are then read into the main guidance program so the information can be used to calculate satellite orbits.

6.5 Development of YUMA data file

The data calculated by `getnmea` would normally be read straight into the guidance program and not output to a data file, however for testing purposes YUMA data was output to a text file for comparison with other YUMA almanacs.

6.5.1 Definition of YUMA Almanac

A YUMA almanac is simply another, more easily viewed way of displaying the orbit data of a satellite (compared to Hex NMEA data) and it consists of the following data.

The NMEA almanac can provide enough data to calculate every part of a YUMA almanac, however for the purposes of the guidance software, we do not require the parameters Af0 and Af1, and hence they are not calculated by the system.

The method of writing the YUMA data to a text file uses the same process for opening

- ID
- Health
- Eccentricity
- Time of Applicability
- Orbital Inclination
- Rate of Right Ascension
- Root of Semi-Major Axis
- Right Ascension at Time of Almanac
- Argument of Perigee
- Mean Anomaly
- Af0
- Af1
- Week

Figure 6.15: Definition of YUMA almanac

(T.S. Kelso 2005)

the file as 6.2 except the data it stores comes out of variables, and not a COM port and the file location is different. The process is run from a subfunction called 'Displaysub' and an excerpt of the code to write the variable `nd` to a text file is shown in figure 6.16 and describes the loop required to write the separate sets of data, as well as skip the sets that correspond to unhealthy satellites.

The entire listing of the software required to convert NMEA data to YUMA data can be found in appendix C.5

```
For i = 1 To p
    If nd(i).weekno <> 0 Then
        print data to file
    End If
Next
```

Figure 6.16: Code required to write YUMA data to file

6.6 Useage of YUMA data

The YUMA almanac data is used by the program to calculate the azimuth and elevation of each of the satellites the software is using for guidance. This information is provided by the GPS 18-5 receiver, however the data is transmitted as integers and as guidance requires an accuracy of greater than six decimal places, the Garmin information is not useful. It is for this reason that the almanac is downloaded from the receiver and converted to YUMA format. By using the YUMA data, the software can calculate the parameters of the satellites' orbit's to greater than six decimal places.

Figure 6.17 shows a NMEA 0183 almanac string and its corresponding YUMA almanac format.

```
$GPALM,32,1,01,369,00,3430,4E,1E44,FD69,A10D64,B76C08,11C706,C24C30,052,001*05

**nmea2alm**   week 369 almanac for PRN-01   **nmea2alm**
ID:                          01
Health:                      000
Eccentricity:                6.370544434E-03
Time of Applicability(s):    3.194880000E+05
Orbital Inclination(rad):    9.889046834E-01
Rate of Right Ascen(r/s):   -7.577458489E-09
SQRT(A) (m^1/2):             5.153673828E+03
Right Ascen at TOA(rad):     4.363239074E-01
Argument of Perigee(rad):   -1.781332194E+00
Mean Anom(rad):             -1.514404557E+00
Af0(s):                      7.820129395E-05
Af1(s/s):                    3.637978807E-12
week:                        369
```

Figure 6.17: Comparison of NMEA 0183 String to YUMA Format

6.7 Adding Guidance Code to Software

The software supplied by Prof. John Billingsley for use with the GPS 18-5 receivers was already able plot the movement of the mobile receiver on a screen, and log those movements to a CSV file, however it did not have any of the guidance capabilities of the older software.

The modifications of the software to implement guidance are described in the following sections, but first it would be prudent to provide a screenshot of the guidance screen to provide insight to the different buttons their corresponding commands.

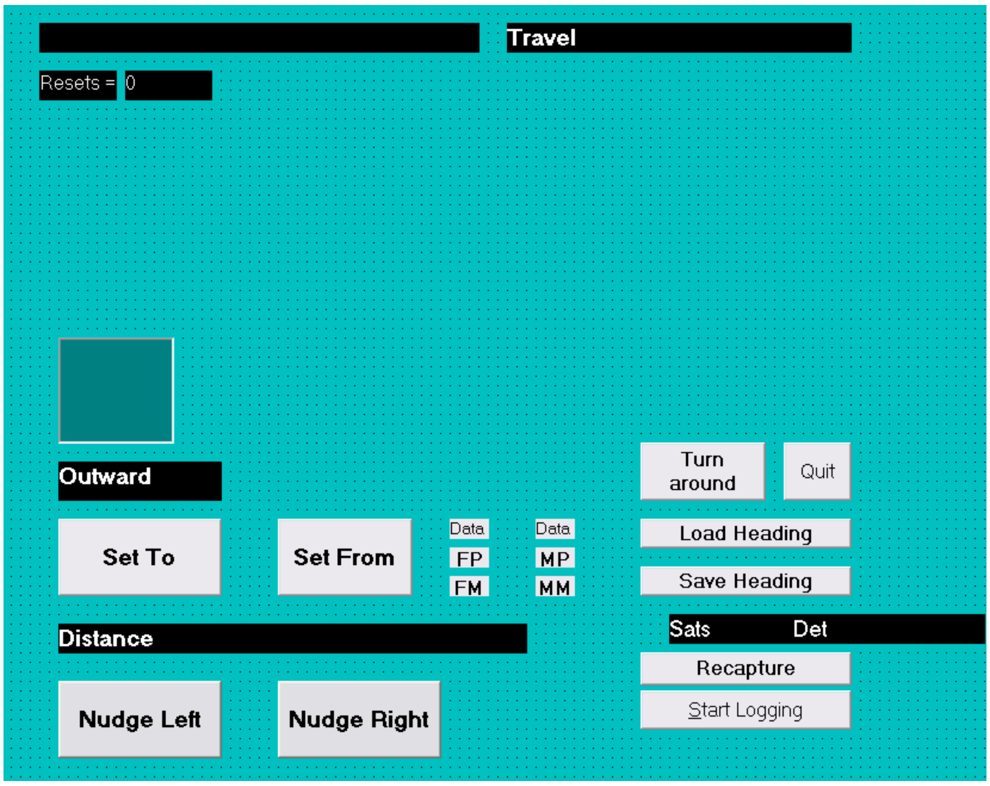


Figure 6.18: Screenshot of Guidance Screen

The code that corresponds to this form can be viewed in appendix C.6.

6.7.1 Position data for use in guidance

The variable that the position data was stored in each time the program looped had to be found before any sort of coding could be completed. Using the existing position logging code, and some scrutiny of the `model` subfunction, the variable to store the position of the mobile receiver was found to be `xm`.

This variable is an array of three pieces of information: Northing, Easting and Altitude. The altitude is not required for guidance and is thus ignored, but the northing and easting are the crux of the guidance data. `xm(1)` describes the northing data and `xm(2)` describes the easting data.

`xm` stores the offset of the mobile receiver from the original starting point. When there are enough satellites for the system to get a differential fix, the subfunctions `calcvel2` and `solvev` are used to calculate the variable `dxc`. `dxc` contains the northing, easting and altitude differences (in metres) of the current point compared to the origin point. As the software operates, `xm` is incremented by `dxc` and thus the position of the mobile receiver in reference to the origin point is calculated with great accuracy.

`xm` is used in many different parts of the guidance software and these separate parts are described in the following sections. The main function that calculates the guidance will be left until last, so that an understanding of all that is required to guide a machine can be developed.

The code for the subfunctions `calcvel2` and `solvev` can be found in appendices C.7 and C.8 respectively.

6.7.2 The 'dokeys' subfunction

The `dokeys` subfunction provides the user with the ability to control the program using the computer keyboard. The forms that Visual Basic generates have an embedded property called 'KeyPress' which takes the ASCII value of the pressed key and stores it in a variable. This information is then used as shown in figure 6.19.

```
Private Sub Form_KeyPress(Keyascii As Integer )
ainkey$ = Chr$(KeyAscii)
If ChrB$(KeyAscii) = esc Then End
End Sub
```

Figure 6.19: Code to gain ASCII value of pressed key

The variable `ainkey$` is a variable defined to be a string and the command `Chr$` converts the ascii value stored in `KeyAscii` to a string. The `ainkey$` variable is then analysed by dokeys, and if there is a command corresponding to that key, it executes the command. The final test tests whether the program should end by comparing the value of `KeyAscii` to the value of `esc`. In this case, `esc` contains a string corresponding to the escape key, hence the program will end if escape is pressed.

The full code listing of `dokeys` can be viewed in appendix C.9.

6.7.3 The 'Set From' and 'Set To' commands

A guidance system is nothing without some initial path to follow, and the `Set From` and `Set To` commands are what set this path. The commands store the start and end points of the line (sometimes called an 'A-B line'), to be followed by the guidance software.

The `dokeys` subfunction controls the activation of the code to calculate the 'from' and 'to' points and thus requires the buttons 'f' and 't' to be pressed to activate their respective commands. It is also possible to activate the commands using the 'Set From' and 'Set To' buttons on the guidance screen (shown in figure 6.18). These buttons contain the code shown in figure 6.20 and this code imitates a key press action.

```
ainkey$ = "f"
```

Figure 6.20: Code to imitate a key press

When `Set From` is called, it simply stores all the current position data in `xm` into a new variable called `xf` meaning position 'from'. It then sets the variable `fromset` to true to signify that a starting point has been calculated.

When `Set To` is called, it checks if a heading has already been calculated, and if not, sets a variable `xt` to the position in `xm`. If a heading is already set, it tells the operator to set their 'From' point first.

The software is written so that if a heading is already set, you can begin guidance from any point; all that is required is execution of the `Set From` command. If a heading is not already set, and the 'from' and 'to' points have been found, the program then calls the subfunction `makenormal`.

The code executed by the 'Set From' and 'Set To' commands can be found in appendix C.9

6.7.4 The 'makenormal' subfunction

The subfunction `makenormal` is used by the software to generate a normalised vector of northing and easting data for the heading in which the machine will be guided. It calculates the distance from the 'from' point to the 'to' point in terms of northing and easting and then calculates the magnitude of this vector. It then normalises the vector by dividing the components by the magnitude.

This normalised heading can be used from any start point, and can be stored to disk by the program for later use.

Once the calculations are complete, the subfunction calls the `show heading` subfunction.

The `makenormal` subfunction code can be found in appendix C.10.

6.7.5 The 'showheading' subfunction

The `showheading` subfunction display a graphic representation of the guidance heading on the guidance screen. It uses the normalised heading vector along with 'outward' data to draw a line on the 'Compass' frame (see figure 6.21).

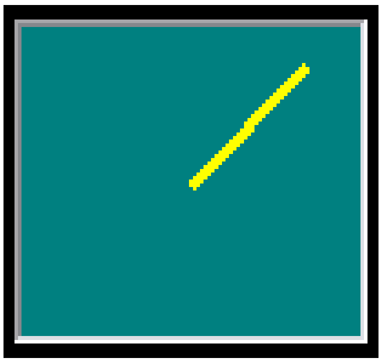


Figure 6.21: Guidance heading

The ‘compass’ depicts the angle of the heading with reference to North. In the above case, the compass depicts that if you followed the guidance program, you would be travelling either North East or South West.

The code to plot the compass can be viewed in appendix C.11.

6.7.6 The ‘outward’ variable

`outward` is a variable which is either positive one or negative one depending on the direction of travel. It is what reverses the guidance commands and causes the distance travelled to be subtracted when you turn around. If you have just begun your run, it displays ‘Outward’ in the label containing ‘Outward’ in figure 6.18 and if you have turned around, it displays ‘Return’. An example would best describe its use.

Imaging you are travelling up the paddock and you drive to the left. The software tells you to turn right. Now, if you do not change outward and simply turn around and then drive right while heading back down the paddock, the software believes you have turned left, and tells you to turn right. This is in error as you would in fact be turning further away from the desired line.

6.7.7 Saving and Loading a Heading

Saving and loading a heading can be done by clicking either the ‘Save Heading’ or ‘Load Heading’ buttons on the guidance screen (figure 6.18).

‘Save Heading’ opens a text file in the same manner as stated previously in this document and then writes the two normalised vectors comprising the heading vector. It then displays the text ‘Heading saved to disk’ on the steer screen.

‘Load Heading’ opens the data file on disk and reads the stored normalised vector data into the program.

The code to save and load a heading can be found in appendix C.6.

6.7.8 The ‘model’ subfunction

The guidance portion of the software occurs in the subfunction `model`. This function contains the code that calls `calcvel2` and through it `solvev`. It then takes `xm` as it is updated by `dxc` and uses that data to display the position of the mobile receiver on the screen. It is to this portion of the software that we will add the main part of the guidance software.

The guidance software uses the position of the mobile receiver with reference to the origin point (the start of the first run) and the normalised heading coordinates to guide the machine. The calculation of the variable `side` - the variable that stores the distance the machine is away from the required line - and the variable `Travel` - the variable that stores the distance travelled - is shown in figure 6.22.

```
If headset And fromset then
side = (xm(1) - xf(1)) * normal(1) + (xm(2) - xf(2)) * normal(2)
Travel = (xm(1) - xf(1)) * normal(2) - (xm(2) - xf(2)) * normal(1)
...
End If
```

Figure 6.22: Code to calculate side and Travel

Initially, the system checks to see if there is an actual heading to follow by testing if `headset` and `fromset` are both true. If so, then the guidance calculations can occur. If not, then the system notifies the operator that they have yet to fully define a heading.

To calculate the distance the machine is away from the required line, a vector calculation is required. By subtracting the coordinates of the origin location from the current location you obtain the vector from the origin to the current point. If you now multiply this by a normalised vector in the direction of the heading, and then add the two results, you obtain a vector perpendicular to the heading, which is the distance from the line to the current position.

A similar calculation occurs when calculating `Travel` except you reverse the components of the normalised vector, giving a vector in the direction of the heading. When you subtract the two results, you obtain the distance from the origin to the point along the heading that is perpendicular to the current position of the machine.

As has been previously mentioned, the ability for the guidance system to turn around over an accurate swath width is the main reason for GPS guidance. There is only one line of code within the software that provides this function and it is shown in figure 6.23.

```
side = side - 8 * Int((side + 4) / 8)
```

Figure 6.23: Code to generate accurate swath widths

Using this code, it is possible to skip swath widths of practically any distance. The three numbers, 8, 4 and 8 are all a product of what swath width is required (in this case 8m). If the values were to change to, say, 8.1, 4.05 and 8.1, they would define a swath width of 8.1m.

The next portion of the code in `model` is concerned with using outward to ensure the values of `side` and `Travel` are in the right direction, printing the values of `side` and `Travel` to the screen and telling the operator that they have yet to set a heading.

The next part of the code deals with displaying an indicating arrow on the guidance screen. The software overrides the default scale of the guidance screen so that the top left corner has coordinates (-1, 1.5) and the bottom right corner has coordinates (1, -0.5). This change in scale sets the (0, 0) point in the middle of the screen a small distance up from the bottom, and also sets the width of the screen equal to three, or in the case of the guidance system, three metres.

When the software operates, a line moves from side to side indicating whether you need to turn right or left, and also how far off the correct line you are driving. While the arrow is easy to follow, it is not easy to judge the location of an approaching swath from the arrow, and hence another line was added to the plot. This line is plotted vertically and represents the location of the required line. If the bottom of it and the bottom of the arrow do not line up, then the machine is off line. The code to do this is shown in figure 6.24.

```
Steer1.Line (0, 0)-(-side, steerscale)
Steer1.Line (-side, 0)-(-side, steerscale), vbRed
Steer1.PSet (0, 1.1 * steerscale), vbGreen
```

Figure 6.24: Code to display guidance arrow on screen

The base of the arrow stays at (0, 0) and the top moves back and forwards (visibly) between (-1.5, 1) and (1.5, 1). The large screen scale provides the user with an increased magnification of the distance they are from the required line thereby increasing the useability of the guidance system.

The full code listing of the `model` subfunction can be found in appendix C.12.

6.7.9 ‘Nudge’ Ability

‘Nudge’ refers to the ability for the guidance program to move the guidance either left or right by a small amount, and still continue guiding along the path. It is useful if the path that is being followed (say a row of cotton in a paddock) suddenly offsets to the left or right. This problem could occur because of drift error while planting, or in the case of this software, someone driving offline and accidentally pressing the ‘Set From’ button. In any case, it is always useful to be able to finely adjust the path that the tractor is travelling.

The ‘nudge’ ability was implemented by modifying the `side` variable every time that `model` was executed. Modifying the ‘nudge’ distance required implementation of a variable called `sideoffset` as an integer, and the addition of commands to `dokeys` where if ‘o’ (or the corresponding ‘Nudge Left’ button) was pressed, 25 was added to `sideoffset` and if ‘p’ (or the corresponding ‘Nudge Right’ button) was pressed, 25 was subtracted from `sideoffset`.

The code to change the value of `side` using `sideoffset` was implemented into `model` before the code that governed the swath width. The code to control the ‘nudge’ is shown in figure 6.25.

```
side = side + (sideoffset / 1000)
```

Figure 6.25: Code to implement ‘nudge’ ability

In the above code, `sideoffset` is divided by 1000 because the unit of `sideoffset` is millimetres, and `side` is in metres.

By implementing the `sideoffset` adjustment before the swath control, the program has to continually obey the offset set by `sideoffset`, hence the variable is also modified by `outward` so that if the offset is left nonzero when a turn is made, the offset will be in the right direction.

6.8 Simulation Programs

During the process of developing the software for the GPS 18-5 receivers, two simulation programs were written to gain a better understanding of the operation of some of the algorithms. The first simulation program was designed to provide easy simulation (i.e. no requirement for operational GPS data) and insight into the operation of the swath width algorithm and the second designed to simulate a form of steering motor control.

The first simulation was developed by setting up a blank form and using the `KeyPress` property, it was possible to generate software that would increase the value of `side` at a rate of 0.1m per each key press. It would then print `side` to screen along with the modified value of side (`side1`) that the algorithm had analysed.

The code for this simulation is shown below in figure 6.26.

```
Form1.Cls
side = side + 0.1
side1 = side - 8 * Int((side + 4) / 8)
Form1.Print "Side "; "Side1"
Form1.Print side, side1
```

Figure 6.26: Code simulating swath width control

By running this code, and changing the values that define the swath width, it was possible to gain a better understanding of the operation of the algorithm, and also work out if it would work for non-integer widths such as 8.1m (which it does).

The second simulation was designed to simulate the simple control of a steering motor where if the machine was too far right, the system would steer left three times and then stop and vice versa. The algorithm for the control system is certainly not good enough for commercial use, however its main use was to see if the tractor could drive from a point to another point in a straight line, ignoring any sort of wiggles along the way.

The simulation also required the use of the `KeyPress` property, but also required six separate labels on the form. When the user pressed the 'd' key, the system would interpret that as wanting to turn left, and it would turn left until 'd' had been pressed three times. After that, any further key presses would be ignored. If the system then needed to turn right (the user pressing the 'f' key) it would do so until the control detected that the system was now steering three increments in to the right. This would in fact take six presses as the system was previous at three right, and needed to come back to centre first.

The simulation's ability was further increased to the point where it could actually control a motor attached to a H-bridge. Connection of a motor, however, was not so successful.

The code for this simulation program can be found in appendix C.13.

Chapter 7

Testing of GPS 18-5 Based System

7.1 Introduction

The only way to fully quantify the accuracy and usability of the newly developed GPS 18-5 based guidance system is to test it as it would be used commercially.

The testing process seeks to quantify the accuracy of the system to provide guidance along a heading, the accuracy of its ability to offset over a specified swath width as well as measure the usability of the guidance software from an operators perspective.

7.2 Data transmission

As was with the GPS 35 guidance system, transmission of data between the base station and the machine is required for fully operational on-farm guidance. The Proxim Proxlink transceivers were going to be used for this purpose until it was discovered that they had a maximum transmission baud rate of 19200 bits per second; half of what is required. The next avenue explored was the use of the XBee Pro zigbee transmitters from the GPS 35 system. First tests of the zigbee's were not promising as the data being received by the laptop was corrupted. It was discovered that the zigbee's were programmed to transmit at 9600 baud thus they needed to be reprogrammed to 38400 baud. Once this was done however, the corrupted data issues were still a problem and the zigbee transmitters could not be used in the guidance system.

As a last resort to get the testing process operating, a 100m two-wire shielded cable was used to connect the base station to the machine. It was not a pretty method of transmission, and would certainly not be used in a farming operation, but it did the job. The transmission system is shown in figure 7.1. The configuration of the transmission system as mounted on the tractor is shown in appendix D.4.

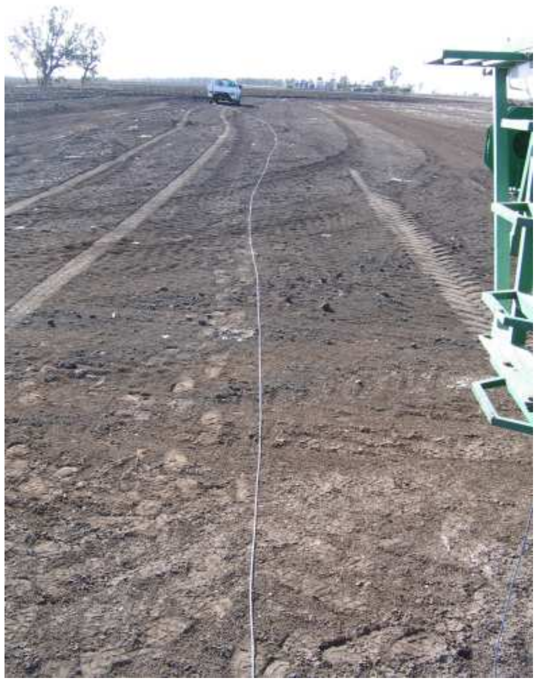


Figure 7.1: Transmission cable used to connect base station to machine

If more time was available, a set of zibgee transmitters would have been sourced directly from MaxStream. The XBee Pro RF modules come prepackaged as RF modems that you can simply plug an RS232 port in to. I believe that if these had been used, the corruption errors that were experienced would have been corrected, and true system testing could occur. An image of the RF modem is shown in figure 7.2

Figure 7.2: MaxStream XBee Pro RF Modem

7.3 Testing of system

The GPS 18-5 guidance system was testing to the methodology set out in chapter 3 and that data analysed as follows.

7.3.1 Accuracy of Orbit Data Calculations

To quantify the error between the Garmin orbit data and the YUMA-calculated orbit data (see section 6.5), the orbit data for four satellites calculated by the Garmin and software was logged over a period of 1hr. That information was then plotted in Excel, and the orbit data compared. It was expected that the Garmin information would 'step' up or down, depending on the orbit data, while the YUMA data would move smoothly from point to point. A graph showing a comparison of orbit data for one of the logged satellites is shown in figure 7.3.

Figure 7.3 clearly shows that the orbit data calculated from the YUMA data is accurate as it follows the Garmin information, and is also of a much higher quality.

Graphs showing the accuracy of the orbit data calculations for this satellite and three others along with a software representation of the accuracy of the orbit data calculations

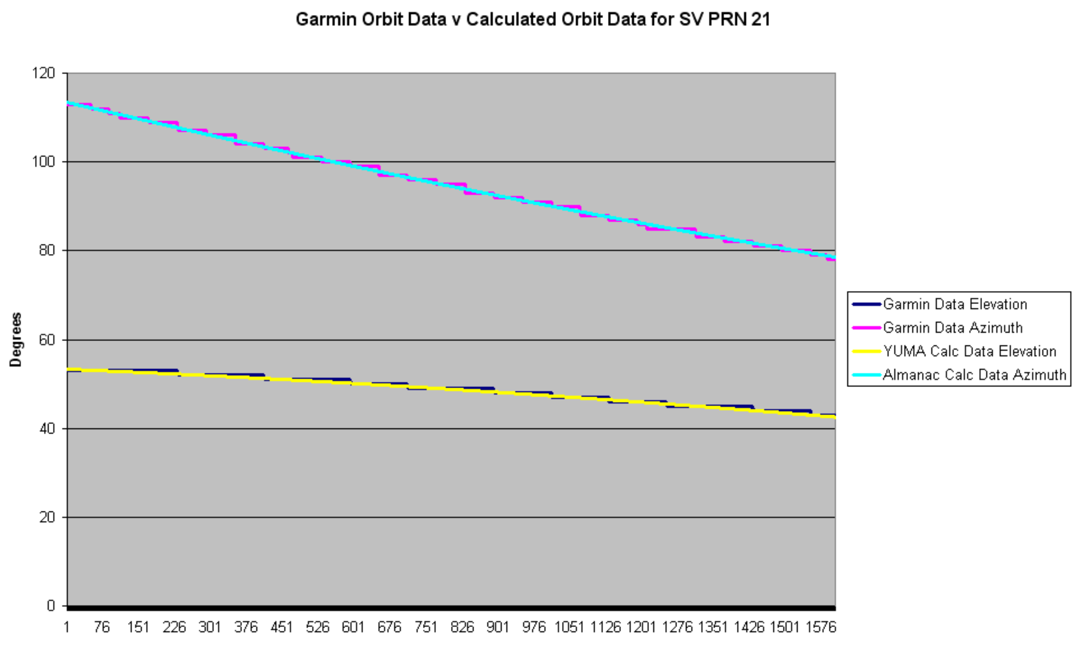


Figure 7.3: Comparison of Garmin and Calculated Orbit Data

are shown in Appendix D.1

7.3.2 Small-scale accuracy

The small-scale testing of the GPS 18-5 system involved placing one of the receivers on the roof of a house, and the other approximately 15m away on a star picket. This configuration was used for logging the drift of the guidance system while under controlled conditions.

As the GPS 18-5 software bases its guidance on the YUMA almanac described in section 6.5, the almanac of the receivers had to be fully updated before any testing could occur. After approximately 15 minutes, the NMEA 0183 almanac was downloaded from one of the receivers and then converted to YUMA format using the method described in section 6.4. Once this process was completed, testing of the system was commenced.

Once the system was operational, the drift was logged over a period of five minutes and the results of this analysis are shown in figure 7.4.

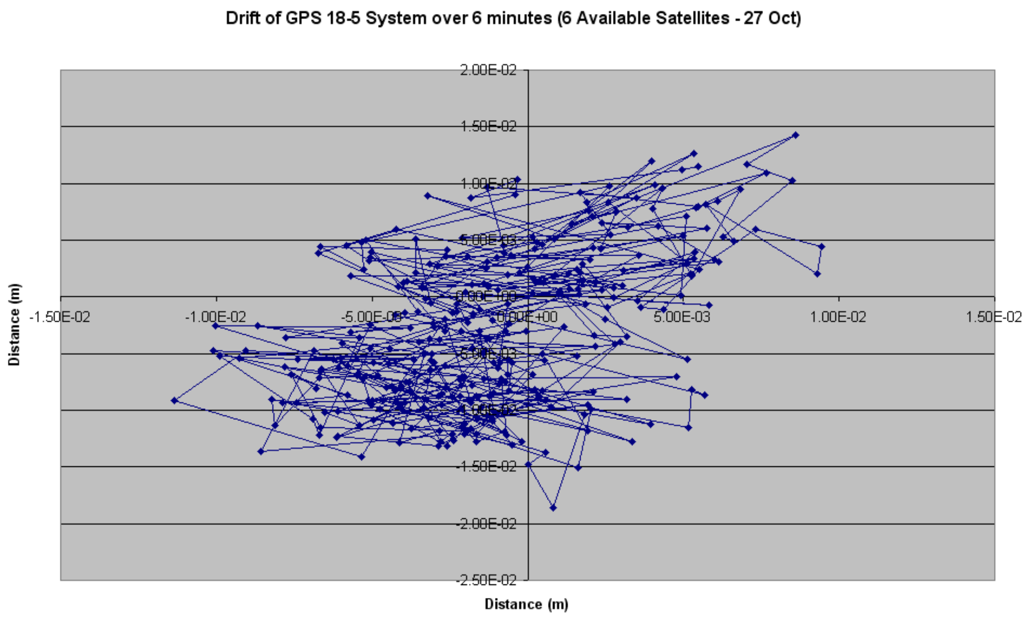


Figure 7.4: Drift of GPS 18-5 system

From the above figure, we an infer that the accuracy of the GPS 18-5 system is what is required. The system has drifted a total of two centimetres in the x-direction and 3.5cm in the y-direction. This accuracy is easily within the required accuracy of ±2cm.

Extra graphs displaying drift information for the GPS 18-5 system can be viewed in appendix D.5

7.3.3 On-farm accuracy

Testing of the on-farm accuracy of the GPS 18-5 system required laying the transmission cable out, mounting the fixed receiver on the base station and the mobile receiver on the cab of a John Deere 8300 tractor.

During testing, steering direction was once again provided by the laptop, however this time it was mounted on a platform just to the right of the steering wheel. This was done to make it easier for the operator to watch the guidance screen and the paddock ahead at the same time. It also freed up a hand as the operator did not have to hold the laptop any more.

The accuracy of the system was supposed to be tested with and without hands-free guidance, however the hands-free guidance system did not come to fruition. It was discovered that the serial ports on the laptop would not drive the H-bridge controlling the motors as the port does not have the ability to go and stay high. Instead it switches on and off rapidly and as such, does not operate a H-bridge.

The method that was going to be used to interface the motor with the steering wheel for testing purposes *only* was tested and the results were promising. The motor was not mounted; a second person held on to it while the H-bridge was activated. The motor (out of a cordless drill) had no trouble at all turning the steering wheel thus confirming that a motor such as that could be used.

Testing of the system is shown in figure 7.5.

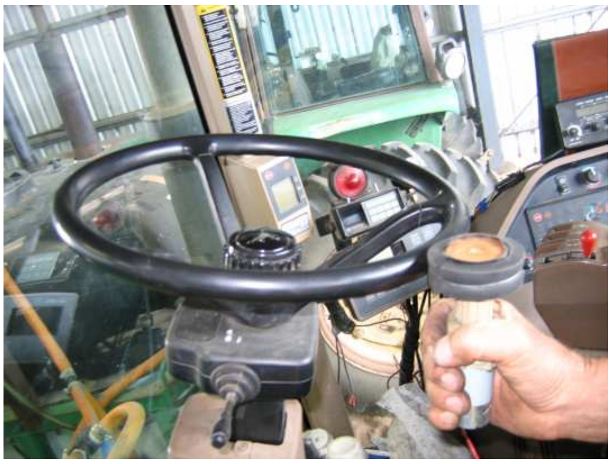


Figure 7.5: Motor configuration for testing hands-free guidance

While testing the motor, it became clear that any form of autonomous guidance mounted on the steering wheel would either require some very smart mathematics or a steering position sensor: there was at least 30 degrees of free play in the steering system. If a steering position sensor was mounted, then the software would know when the wheels started turning, and would be able to calculate the required steering angle after that. If no steering angle sensor was used, the controller would have to turn the steering wheel until the system detected that a change in trajectory was occurring. It could then use this information along with previous data points and motor input values to calculate the angle of curvature of the trajectory to develop an idea of the steering angle.

Testing of the repeatability and accuracy of the system was completed in two separate manners: the first being traversing a trajectory in both directions and the second being testing the swath offset.

Testing the ability for the system to traverse a trajectory in both directions also tests the systems ability to guide the machine in a straight line. Figure 7.6 shows the path of the test rig while driving up and back along a trajectory.

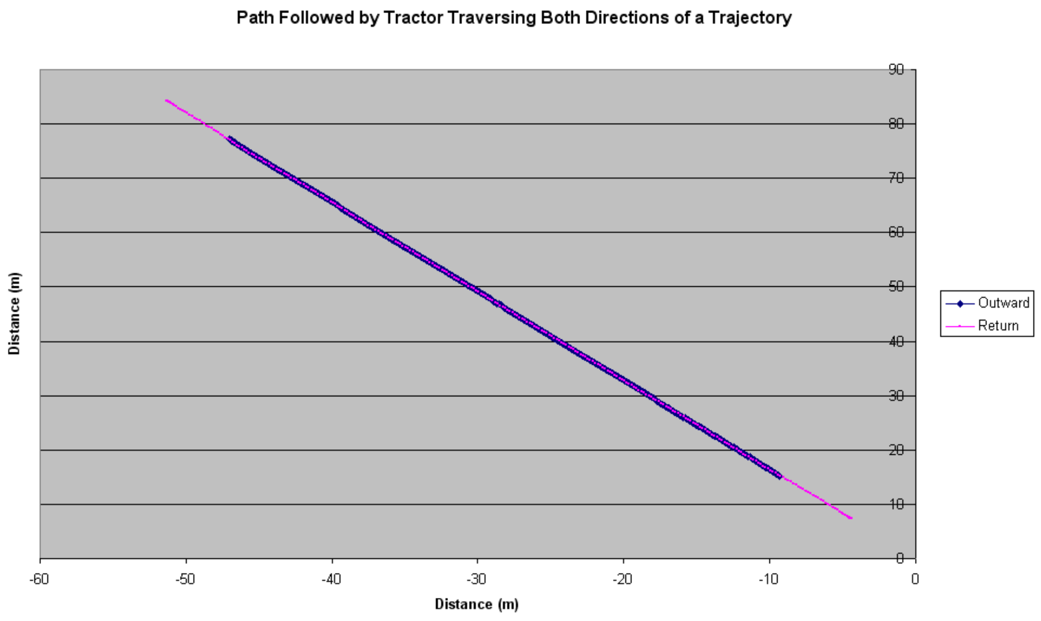


Figure 7.6: Record of path travelled by tractor moving along trajectory in both directions

Table 7.1 shows the distances between the logged trajectories.

Outward Trajectory		Return Trajectory		Swath Width
x	y	x	y	(cm)
-28.969	47.410	-28.924	47.388	5
-12.106	19.802	-12.055	19.743	7.7

Table 7.1: Table of distances between logged trajectories

The figure shows that the path of the test rig on the outward run is almost exactly the same as the path on the return run. From this, we can infer that the accuracy of the

GPS 18-5 guidance software is much greater than the GPS 35 software, and also that it may possess the accuracy required to become commercial.

The single trajectory testing is only half of the story though; we now have to test the systems ability to offset over swath widths. The trajectory of the test rig while executing this test is shown in figure 7.7

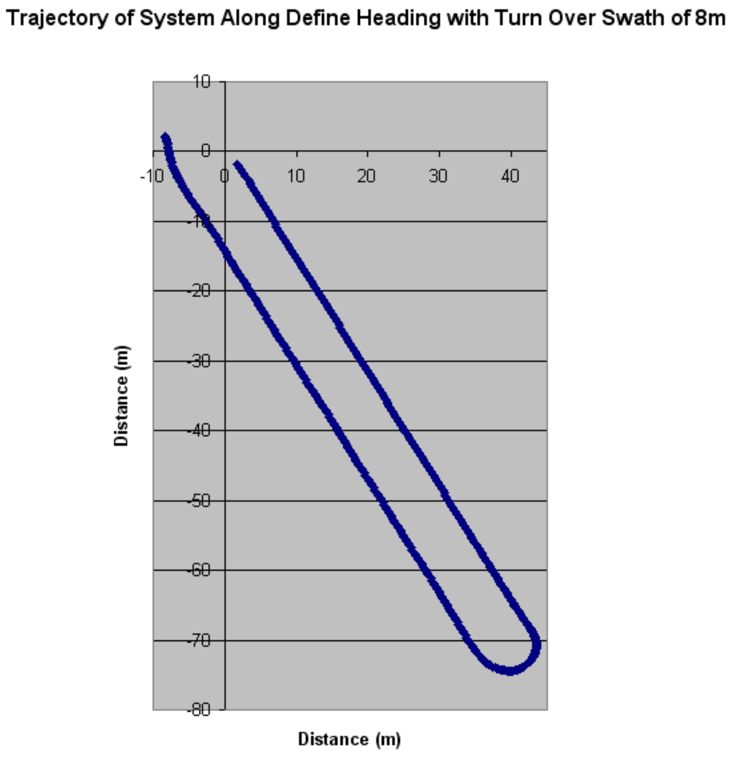


Figure 7.7: Guidance system swath-width offset testing

Extra images showing the trajectory followed by the test rig can be found in appendix D.6

The distance between the trajectory's can be found using the same method as section 4.4.2 and should be equal to 8m. Data on the swath width are shown in table 7.2.

The distance between the two trajectory's was also measured with a tape measure, and the distance was calculated to be approximately 8.05m thus the data shows that the accuracy of the GPS 18-5 system is extremely good, especially considering the guidance was accomplished manually.

Outward Trajectory		Return Trajectory		Swath Width
x	y	x	y	(m)
9.186	-13.954	2.798	-18.844	8.04
20.48	-32.308	13.93	-39.833	8.01
32.172	-51.632	25.604	-56.195	7.997

Table 7.2: Table of swath widths along trajectory

This accuracy must be maintained for many different swath widths, and as such a turn over a swath width of 24m (signifying a sprayer width) was also recorded. The trajectory is shown in figure 7.8.

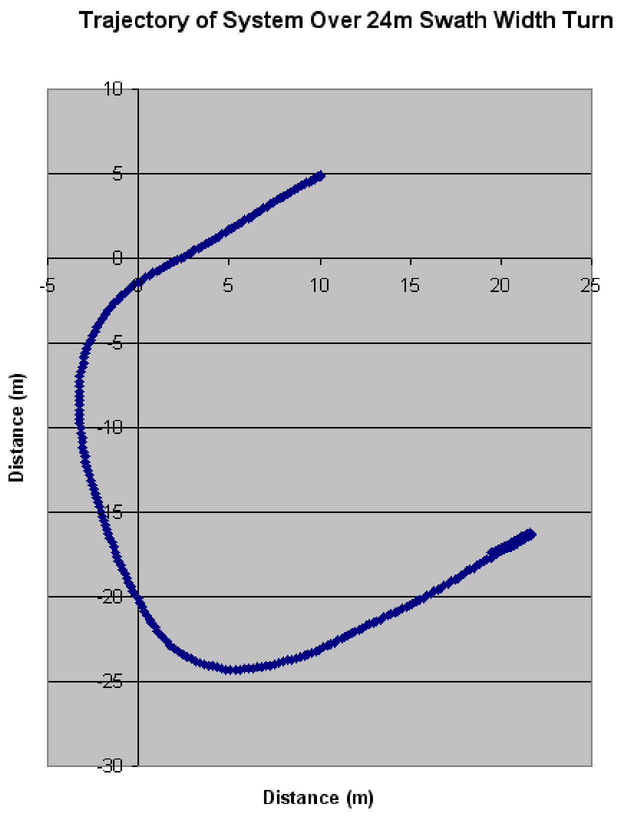


Figure 7.8: Path followed over 24m swath offset

The distance between the two trajectory's is shown in table 7.3

The distance between the trajectory's was measured with a tape measure and the distance found to be 24.1m. This data shows that the system is accurate when turning around over 24m. The reason for the error is most likely due to the lack of room needed

Outward Trajectory		Return Trajectory		Swath Width
x	y	x	y	(m)
3.677	0.814	15.749	-20.046	24.10
6.388	2.594	18.303	-18.391	24.13

Table 7.3: Table of swath widths along trajectory

to properly line the tractor up with the trajectory.

7.4 Usability of Guidance Software

The GPS 18-5 guidance system is much more user-friendly than the GPS 35 system.

Due to the fact that the position and hence the guidance signals are coming at a rate of 5Hz instead of 1Hz, the guidance arrow does not jump around nearly as much as the GPS 35 system; it is quite easy to follow. Another advantage lies in the addition of a vertical line marking the position of the required trajectory compared to current position. The GPS35 system only displayed a white arrow, and while it did provide guidance information, it was very hard to judge how far from the required trajectory you were. The vertical line remedies that problem.

A screenshot of the guidance screen can be found in appendix D.12.

7.5 Conclusion

The GPS 18-5 guidance system is a large improvement over the GPS 35 system in accuracy and usability. Driving a tractor in a straight line is made quite easy with the new receivers (thanks mostly to the 5Hz transmission of position signals) and the accuracy of the swath offset system means turning around is also an easy task.

Chapter 8

Conclusions and Further Work

8.1 Introduction

In this chapter, the objectives of the project will be addressed and uses evidence from the previous four chapters to support any statements made.

8.2 Hardware Upgrade

This section addresses the following project objective:

Upgrade Hardware and Quantify Gains in Accuracy

Chapters 4 and 7 presented information regarding the accuracy of the GPS 35 and GPS 18-5 based guidance systems. Using the data gained from receiver drift logging, it would seem that the GPS 35 receivers would be adequate for guidance applications where ±2cm accuracy was not required (such as spraying) while the GPS 18-5 receivers would be more than adequate for high precision applications such as furrowing and planting.

Table 8.1 shows a comparison between the accuracies of the GPS 35 system and the GPS 18-5 system.

Receiver	Drift in x-direction	Drift in y-direction	Accuracy
GPS 35	20cm	10cm	$\approx \pm$15cm
GPS 18-5	2cm	3.5cm	$\approx \pm$2cm

Table 8.1: Drift Comparison of GPS 35 and GPS 18-5 Receivers

8.3 Software Upgrade

This section addresses the following project objective:

Upgrade Software to Accommodate New Hardware, Add New Features and Quantify Gains in Accuracy, Productivity and Usability

Chapter 6 presented the code that was added to the guidance software to accommodate the new hardware, convert a NMEA almanac to a YUMA almanac, calculate satellite locations, provide guidance ability and provide swath offset ability. It also presented simulation code to simulate the specific algorithm used in the swath offset ability and a conceptual algorithm for control of a hands-free controller.

Chapter 7 presented the results from testing of the additions to the GPS 18-5 software.

Section 7.3.1 provided an analysis of the accuracy of the calculation of the satellite positions. The accuracy was calculated to be excellent and figure 7.3 is repeated here for convenience.

As the accuracy of the calculated orbit data is very accurate, we can infer that the calculation of the YUMA almanac from the NMEA almanac is also correct.

Section 7.3.3 provided an analysis of the accuracy of the system's guidance ability in a straight line and across swath widths.

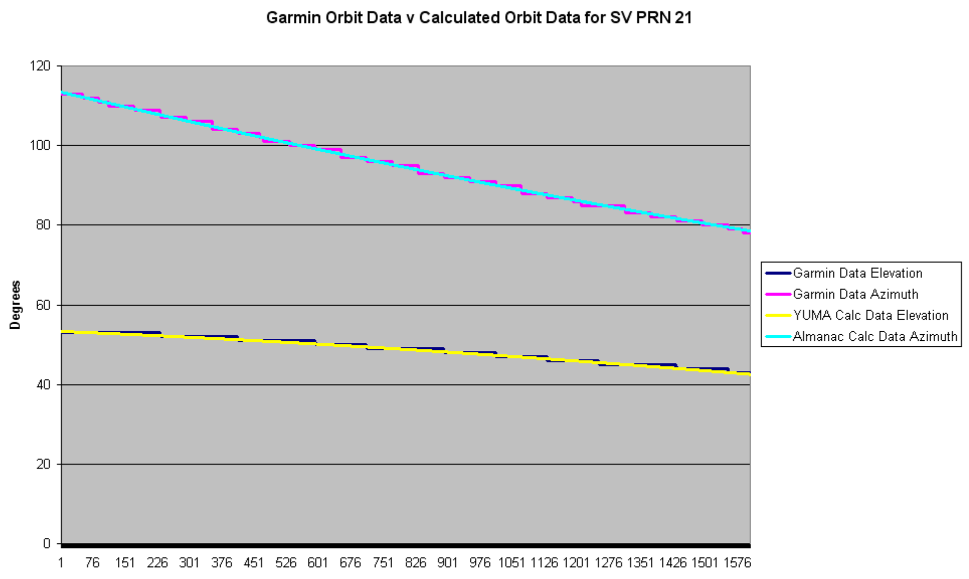


Figure 8.1: Comparison of Garmin and Calculated Orbit Data

Analysis of the system's ability to drive in a straight line, and repeatably cover the same trajectory was very promising. The trajectory plot (figure 7.6) and the table of distances between the two trajectories (table 7.1) are included here for convenience.

Outward Trajectory		Return Trajectory		Swath Width
x	y	x	y	(cm)
-28.969	47.410	-28.924	47.388	5
-12.106	19.802	-12.055	19.743	7.7

Table 8.2: Table of distances between logged trajectories

Although the distance error between the two trajectories is greater than ±2cm, it can be said that had hands-free guidance been used the results would have been more accurate as manual guidance still has many shortfalls when guiding off a computer screen.

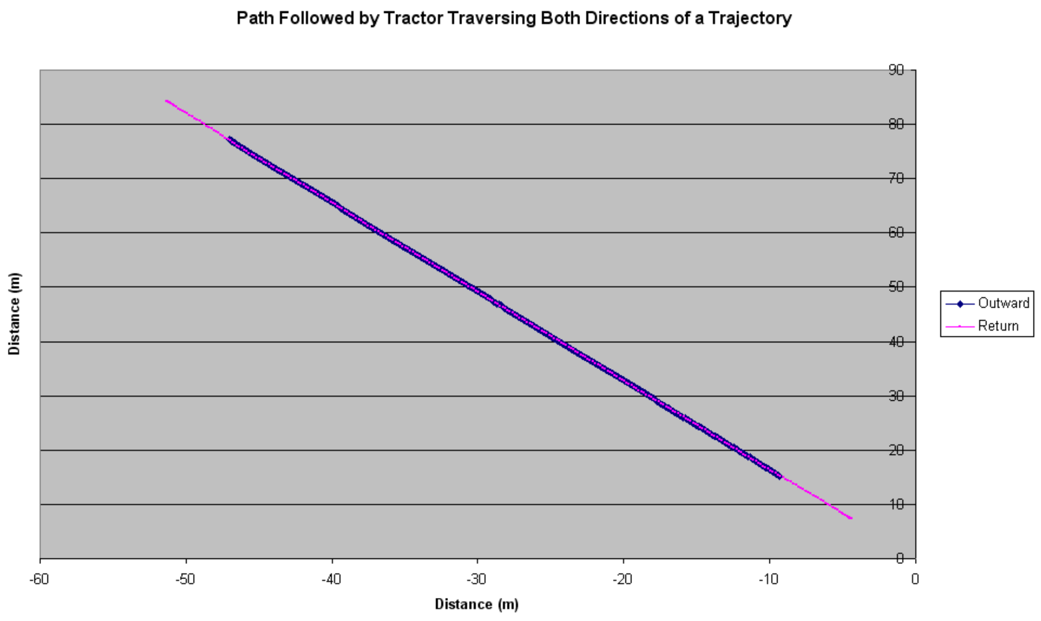


Figure 8.2: Record of path travelled by tractor moving along trajectory in both directions

The final requirement of accurate swath width offsetting was also analysed using the data logged during the tests.

The trajectory plot (figure 7.7) and the table of distances between the two trajectories (table 7.2) are included here for convenience.

Outward Trajectory		Return Trajectory		Swath Width
x	y	x	y	(m)
9.186	-13.954	2.798	-18.844	8.04
20.48	-32.308	13.93	-39.833	8.01
32.172	-51.632	25.604	-56.195	7.997

Table 8.3: Table of swath widths along trajectory

With an average swath width along the trajectory of 1.67cm and a maximum displacement of 4cm, the results of the test were conclusive; the swath width offset ability was working to high accuracy. As with the system's ability to cover a trajectory more than once, the error in this guidance is mainly due to operator error and some unlevel ground.

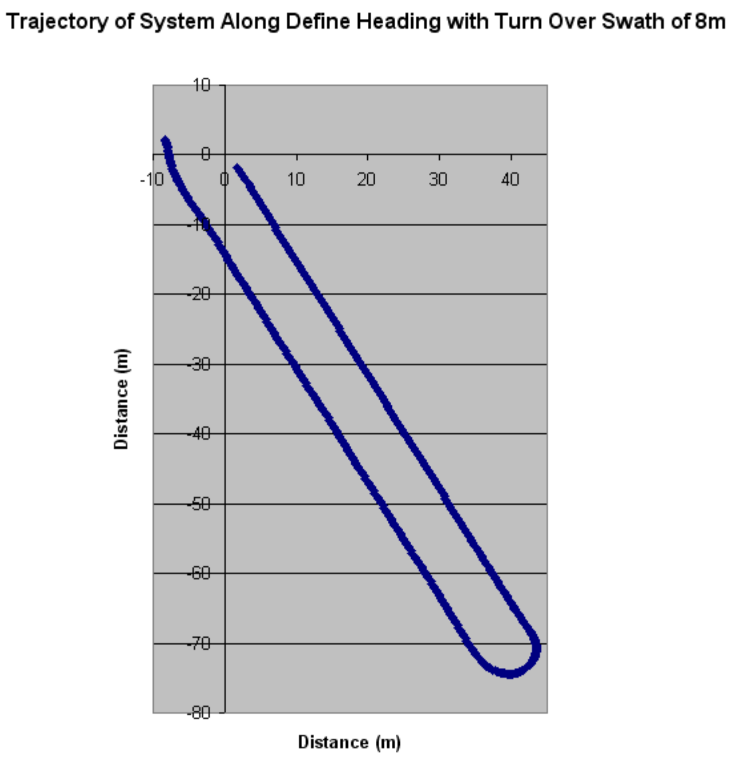


Figure 8.3: Guidance system swath width offset testing

The accuracy of the GPS 18-5 software's offset swath width ability provides a marked increase in productivity compared to the operation of the GPS 35 software described in section 4.5; Coke cans are no longer required to mark out the separate swaths!

As mentioned in section 7.4, the GPS 18-5 software is much more useable than the previous GPS 35 software. Where the previous guidance arrow would jump up to 50cm at a time, the new software jumps about the greatly reduced amount of 10cm. This and the fact that guidance data is produced at five times per second makes the system very easy to use.

8.4 Develop Autonomous Steering

This section addresses the following project objective:

Develop System for Autonomous Steering Operation

Chapter 5 presented three separate conceptual designs for an autonomous steering gearbox. The gearbox designs were not implemented, however there should not be a large amount of work required on the current designs before they were able to be built and tested.

As the gearbox designs were not implemented, testing of the hands-free ability of the guidance system was unable to be completed. However, the data collected during testing with manual guidance showed excellent accuracy thus the conclusion can be drawn that that a hands-free system would only further increase the accuracy.

8.5 Discussion of Possibilities for Further Development

While the GPS 18-5 guidance system is now operational and can guide agricultural machinery, many more features could still be added if research continued.

A microcontroller that accepts serial data could be used to control the H-Bridge and hence the hands-free steering system. It could be interfaced with the H-Bridge and programmed to control the motor speed using pulse-width modulation. Direction and speed control could be achieved by assigning serial input data from zero to 127 as varying grades of motor speed in one direction, 128 as stop and 129 to 256 as varying grades of motor speed in the opposite direction. This system would be compatible with the serial to USB converters currently used with the system.

Other abilities that could be added to the software are:

- Ability to work in circles - This is very useful for farmers who irrigate under centre pivot irrigators.
- Ability to follow a path that is not a perfectly straight line - This is useful to farmers who have paddocks that bend around corners.

To enable the guidance system to guide around corners, a system could be developed that requires the user to follow the first A-B line (hands-free), manually turn the corner, then travel the second A-B line (hands-free) while recording the path followed. The system could then guide back along the path as usual, except that it could be analysing the coming 10 data points to see if a corner is approaching and then calculate the steering angle required to follow the corner.

References

Billingsley, J. (2006), *A Sideways Look at GPS*, Billinglsey, John.
`http://www.skyrule.com/usq/`
current 2006.

Broida, R. (2003), *How to Do Everything with Your GPS (How to Do Everything)*, McGraw-Hill/Osborne Media, New York, U.S.

Bruce Wiebusch (2001), *Design News*, Reed Business Information.
`http://www.designnews.com/article/CA181389.html`
current 2001.

Dana, P. H. (2006), *Global Positioning System Overview*, University of Texas.
`http://www.colorado.edu/geography/gcraft/notes/gps/gif/orbits.gif`
current 2006.

Deere & Company (2006*a*), *John Deere Introduces Swath Control Pro on 4720 and 4920 Sprayers*, Deere & Company.
`http://www.deere.com/en_US/newsroom/2006/releases/farmersandranchers/`
`060303_swathcontrolpro.html`
current 2006.

Deere & Company (2006*b*), *Stellar Support*, Deere & Company.
`http://stellarsupport.deere.com/en_US/`
current 2006.

Deere & Company (2006*c*), *Turnable MFWD Fenders*, Deere & Company.
`http://salesmanual.deere.com/sales/salesmanual/en_NA/tractors/2006/`

feature/wheels_and_tires/6000_7020sf/6015_6020_fenders_turnable_mfwd.html
current 2006.

Dye, S. & Baylin, D. F. (1997), *The GPS Manual. Principles and Applications*, Baylin Publications, Boulder, CO, USA.

El-Rabbany, A. (2002), *Introduction to GPS: The Global Positioning System*, Artech House, Inc, Norwood, MA, USA.

Farnell In One (2006), *Spur Gears*, Australia.

Garmin (2005), *GPS 18 Technical Specifications*, revision d edn, Garmin International, Inc., 1200 E. 151st Street, Olathe, KS 66062 USA.

GRDC (2005), *GRDC - Research Updates - Soil Compaction Trials*, Grains Research & Development Corporation.
http://www.grdc.com.au/growers/res_upd/hirain/h05/whitlock.htm
current 2005.

Hofmann-Wellenhof, B., Lichtenegger, H. & Collins, J. (2001), *GPS. Theory and Practice*, 5th edn, SpringerWienNewYork, New York, USA.

Hollingsworth, D. (2006), *Project KidSafe*, State Capitol.
http://republican.sen.ca.gov/web/36/projectkidsafe/ab1442_sb881.asp
current 2006.

IE Aust (2000), *Code of Ethics*, The Institution of Engineers, Australia.
http://www.ieaust.org.au/about_us/res/downloads/Code_of_Ethics_2000.pdf
current 2000.

Kaplan, E. D. (1996), *Understanding GPS. Principles and Applications*, 5th edn, Artech House, Norwood, MA, USA.

Kowoma (2006), *How GPS works: The control segment*, Kowoma.
http://www.kowoma.de/en/gps/control_segment.htm
current 2006.

Langley, R. (1990), ‘Why is the gps signal so complex?’, *GPS World* **1**(3), 56–59.

MaxStream (n.d.), *XBee OEM RF Modules*, Maxstream, Inc, Lindon, UT 84042.

McManis, C. (2003), *H-Bridge Theory & Practice - - Chuck's Robotics Notebook*, Chuck McManis.
`http://www.mcmanis.com/chuck/robotics/tutorial/h-bridge/`
current 2003.

Ofria, C. (2000), *A Short Course on Automatic Transmissions*, Dolphin Transmissions.
`http://www.dolphintransmissions.com/transmission.htm`
current 2000.

Schrimpf, P. (2005), 'Precision ag's killer app', *Croplife* **6**.

Shaw, M., Sandhoo, K. & Turner, D. (2000), 'Modernization of the global positioning system', *GPS World* **11**(9), 36–44.

Trimble (2002), *AgGPS Autopilot System*, Trimble Navigation Limited, Sunnyvale, CA 94086.

Trimble (2006), *Trimble - All About GPS*, Trimble Navigation Ltd.
`http://www.trimble.com/gps/`
current 2006.

T.S. Kelso (2005), *Definition of a YUMA Almanac*, Centre for Space Standards and Innovation.
`http://www.celestrak.com/GPS/almanac/Yuma/definition.asp`
current 2005.

Wells, D. e. a. (1987), *Guide to GPS Positioning*, New Brunswick: Canadian GPS Associates, Fredericton, Canada.

Xu, G. (2003), *GPS. Theory, Algorithms and Applications*, Springer-Verlag Berlin Heidelberg New York, Germany.

Appendix A

Project Specification

University of Southern Queensland

Faculty of Engineering and Surveying

ENG4111/4112 Research Project
PROJECT SPECIFICATION

For:	Nicholas Ian Hayllor
Topic:	Development of a Cheap GPS Guidance System for use in Agriculture.
Supervisors:	Prof. John Billingsley
Project Aim:	This project aims to further develop a cheap, carrier phase-based, GPS guidance system for agriculture through revision of current software and hardware and on-farm and laboratory testing of factors such as accuracy and ergonomics. It also aims to develop a mechanical device for 'hands-free' guidance.
Sponsorship:	National Centre for Engineering in Agriculture
Programme:	Issue B, 20 May 2006

1. Research background information into the operation of the Global Positioning System (GPS) and its transmission and communication protocols.

2. Acquire current hardware and guidance software revisions for analysis and further development.

3. Perform on-farm and in-house evaluation of equipment testing accuracy, repeatability and operational ease-of-use.

4. Revise software and hardware based upon results from testing.

5. Adjust software to provision for turning around to an exact distance away from a previous pass.

6. Develop mechanical device and interface with guidance computer.

As Time Permits:

7. Continue development of a current interface design with the NCEA through research into ergonomics and current industry requirements.

Agreed:

_______________ (Student) __________________ (Supervisor)

___/___/___ ___/___/___

Appendix B

Autonomous Steering Unit Conceptual Design Information

B.1 Initial Conceptual Design

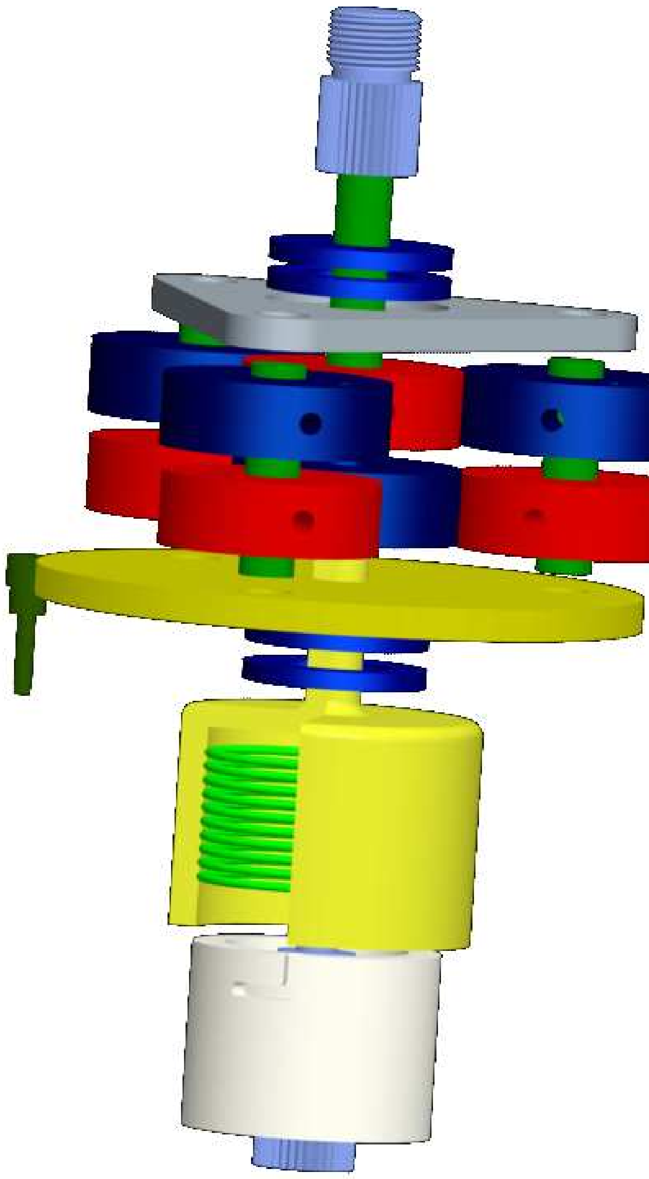


Figure B.1: Exploded view of gearbox design

B.2 Adjusted Design with Offset

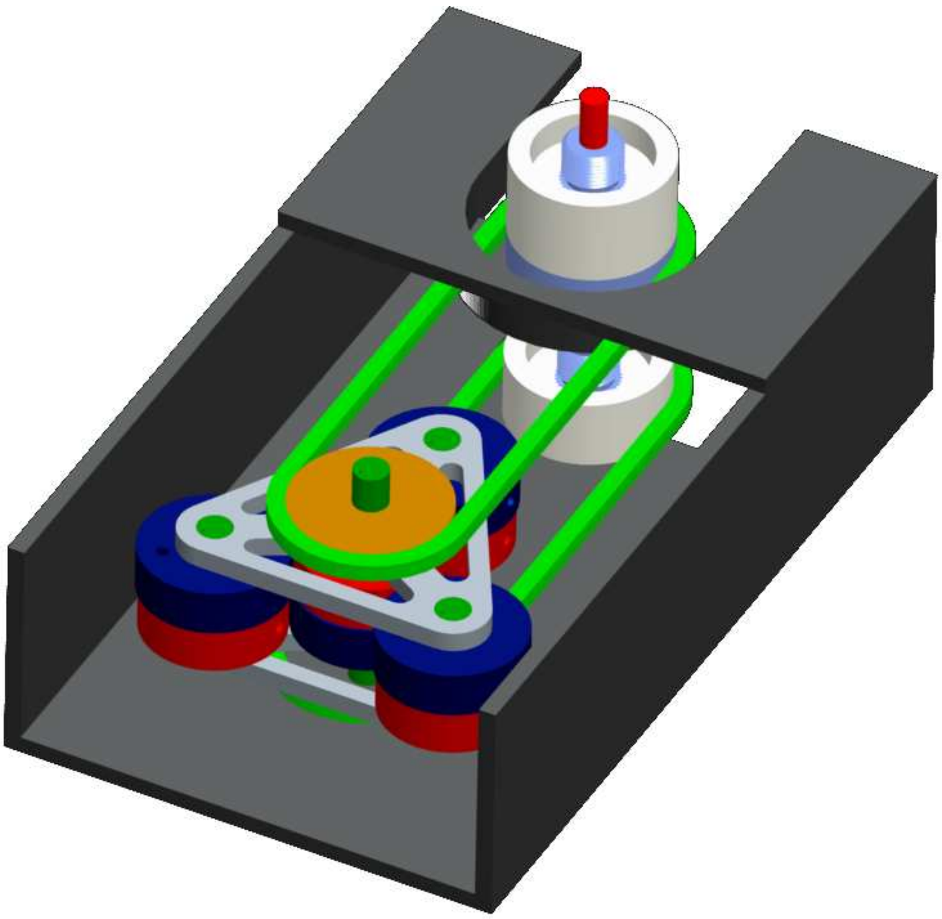


Figure B.2: Alternative view of gearbox offset design

B.3 Detail Drawings

B.3.1 Planet Lock Detail Drawing

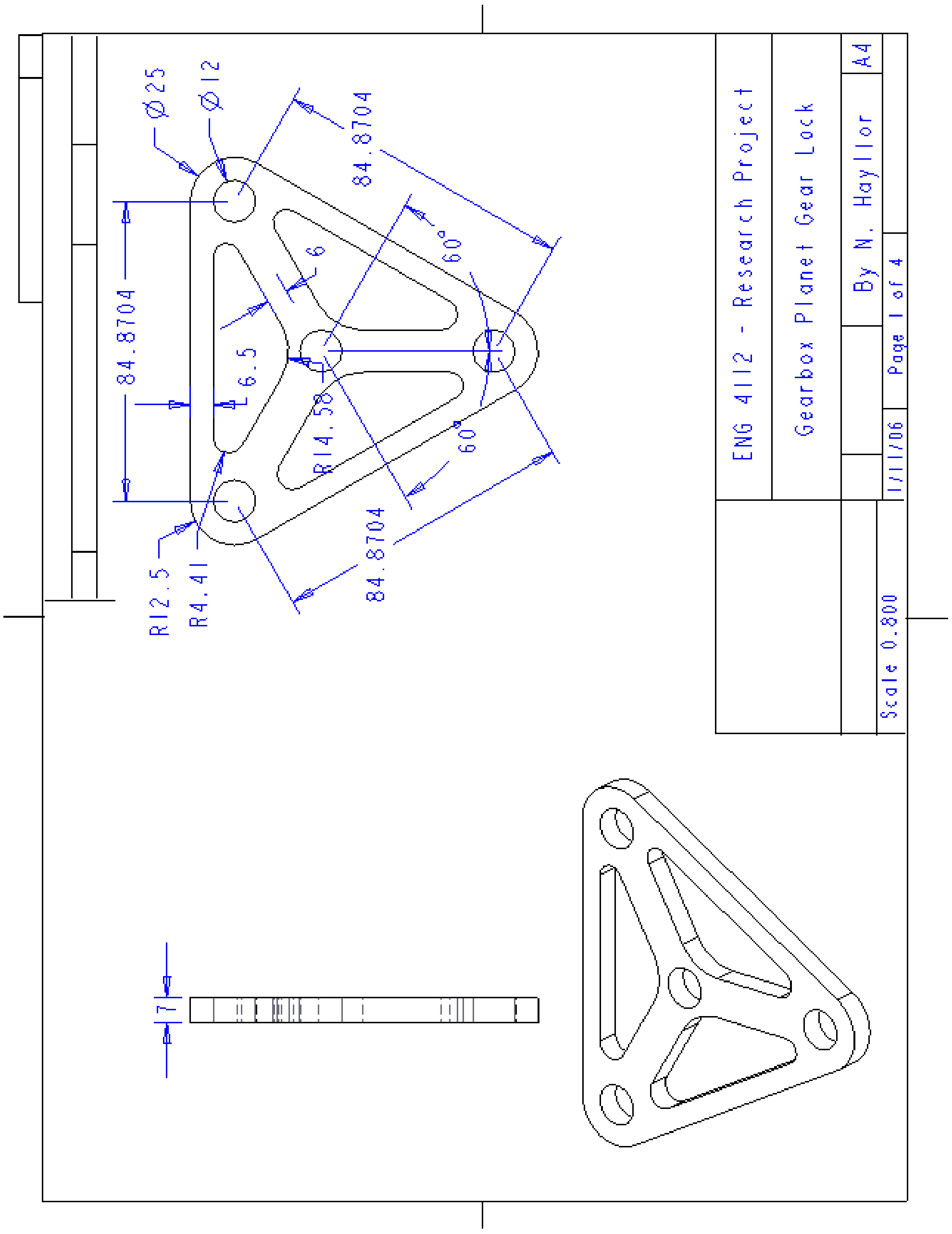


Figure B.3: Detail design for part Planet Lock

B.3.2 Steering Wheel Splined Shaft Mount

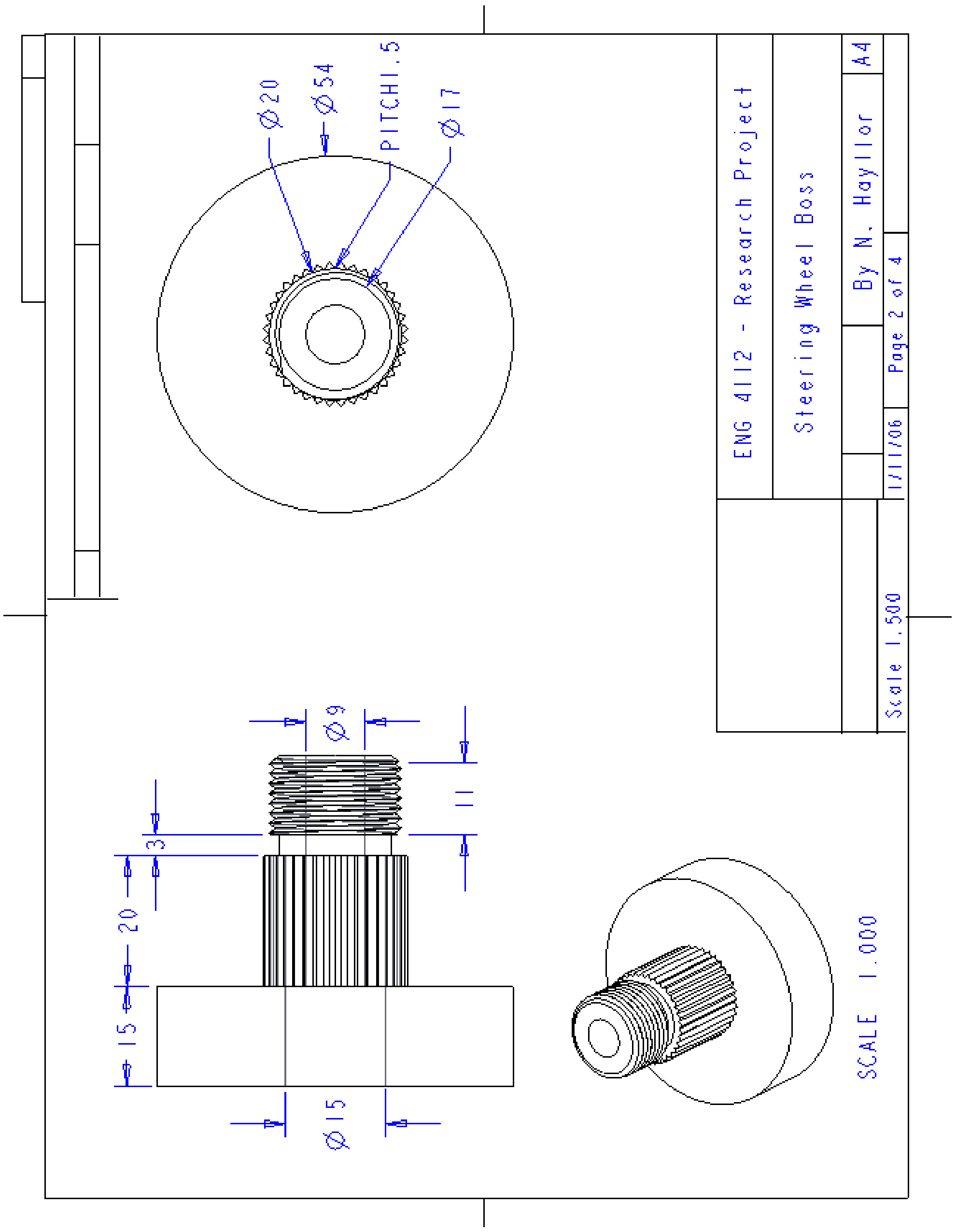


Figure B.4: Boss connecting steering wheel to gearbox via chain

B.3.3 Extension Shaft

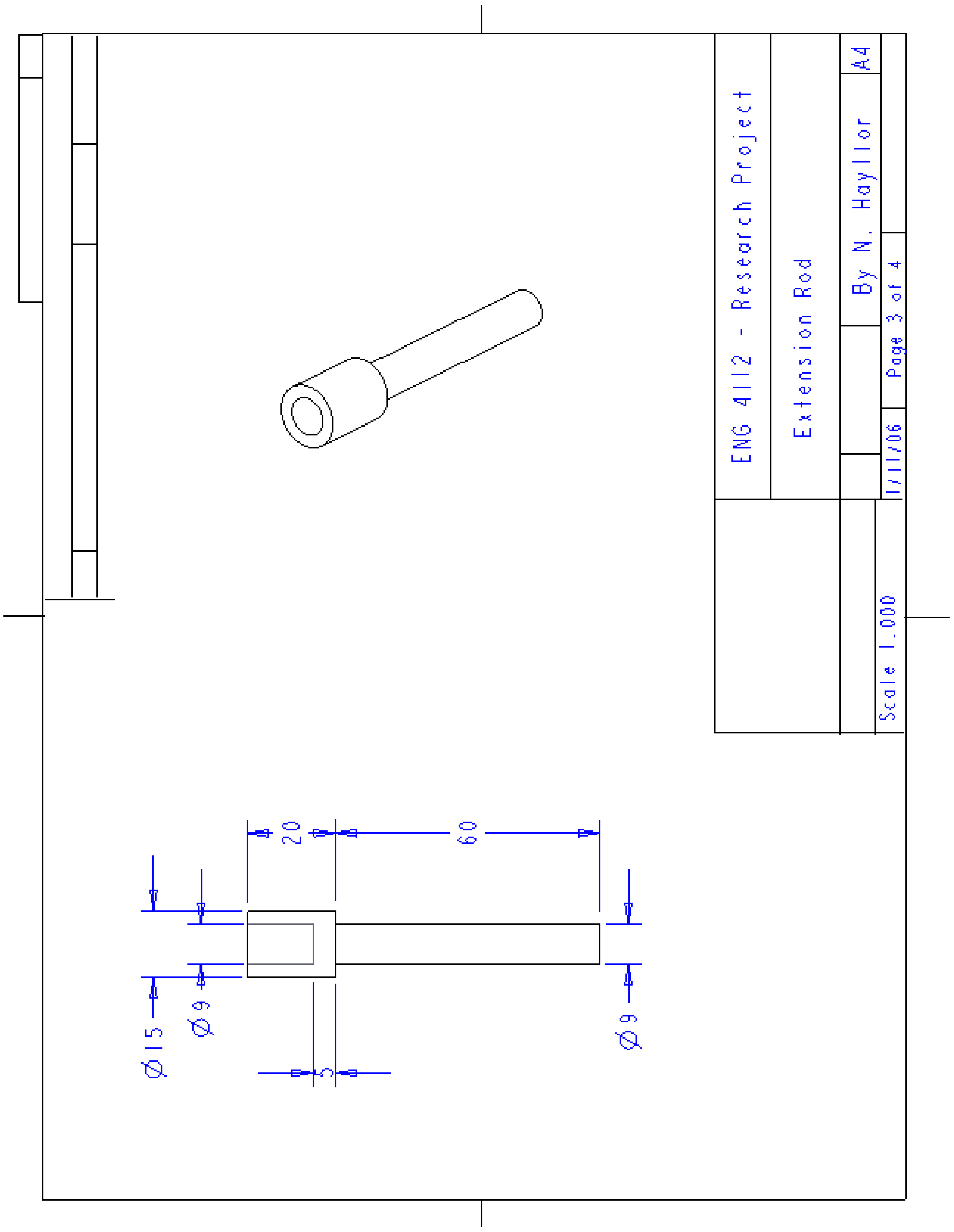


Figure B.5: Rod that extends column length adjuster

B.3.4 Steering Column Mount Boss

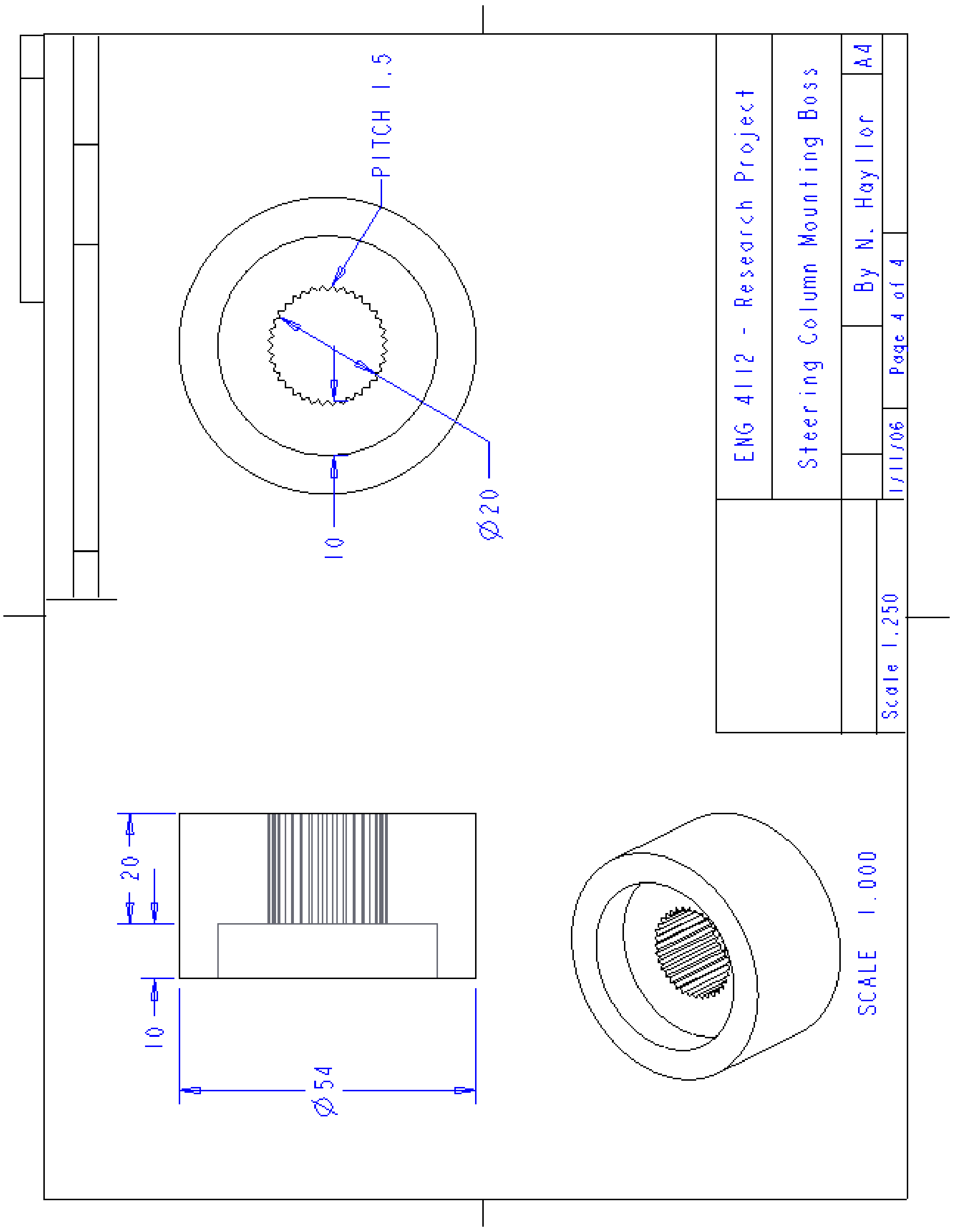


Figure B.6: Boss to mount gearbox on steering column

Appendix C

Software Source Code

C.1 Introduction

This appendix is concerned with presenting the Visual Basic source code for all modified functions and subfunctions within the guidance program. The order in which the function and subfunction listings appear is concurrent with chapter 6 of the document.

C.2 The change38400 VB Subfunction

The function change38400 changes the baud rate of the GPS 18-5 from 9600 baud to 38400 baud.

Listing C.1: Function to change the baud rate of receiver.

```
Attribute VB_Name = "Module1"
Sub change38400()
Board.Print "Change requested", Port

Dim data As Variant, i As Integer
data = Array(&H1C, 0, 0) 'stop messages
message (data)  '&H1C, ChrB$(0) + ChrB$(0)

Board.Print "Messages stopping port"; Port

timeout = False
Do
  timeup = Timer + 2
  getmess
Loop Until timeout
Board.Print
Board.Print "now change baud"
timeup = Timer + 5
timeout = False
data = Array(&H30, 0, &H96, 0, 0) 'Set baud rate 38400 - &
   H9600 = 3840
message (data)

pause 0.2
Do
    getheader
    Board.Print Hex$(ident); " : ";
    If ident = &H31 Then
       Board.Print "Baud "; getLONG
    Else
       For i = 1 To mesbytes
           bget
           DoEvents
       Next
    End If
    tail
    Board.Print
Loop Until (ident = &H31) Or timeout
If timeout Then Stop

data = Array(&H6, &H31, 0) 'OK, change it
message (data)

pause 0.2

If Port = 2 Then
Board.ComMob.PortOpen = False

Board.ComMob.Settings = "38400,n,8,1"
Board.Print "New baud rate"
Board.ComMob.PortOpen = True
timeup = Timer + 1
ping

ping
```

```
data = Array(&HA, 49, 0) 'enable position data (50 for disable
    )
message (data)

data = Array(&HA, 110, 0) 'enable measurement data (111 for
    disable)
message (data)

Else

Board.ComFix.PortOpen = False

Board.ComFix.Settings = "38400,n,8,1"
Board.Print "New␣baud␣rate"
Board.ComFix.PortOpen = True
timeup = Timer + 1
ping

ping

data = Array(&HA, 49, 0) 'enable position data (50 for disable
    )
message (data)

data = Array(&HA, 110, 0) 'enable measurement data (111 for
    disable)
message (data)

End If

End Sub
```

C.3 The getposrecord VB Subfunction

The function getposrecord stores the incoming Garmin binary phase data into variables corresponding to the type of data being received.

Listing C.2: Function that stores position record from receiver.

```
Attribute VB_Name = "Module2"
Sub getposrecord()

posrec(port).Alt = getSINGLE          'ellipsoid altitude
    metres
posrec(port).epe = getSINGLE          'est pos error metres
posrec(port).eph = getSINGLE          'horizontal error metres
posrec(port).epv = getSINGLE          'vertical error metres
posrec(port).fix = getINTEGER         '0,1=no fix, 2=3D, 3=3D,
    4=2D diff, 5=3D diff
posrec(port).gps_tow = getDOUBLE      'gps time of week sec
posrec(port).Lat = getDOUBLE          'latitude radians
posrec(port).Lon = getDOUBLE          'longitude radians
posrec(port).lonvel = getSINGLE       'longitude velocity
    metres/sec
posrec(port).latvel = getSINGLE       'metres/sec
posrec(port).altvel = getSINGLE       'metres/sec
' ********* Added Variables
posrec(port).mslhght = getSINGLE      'Height above sea level
posrec(port).leapsec = getINTEGER     'UTC Leap time
posrec(port).grmndays = getLONG       'garmin days
End Sub
```

C.4 The askalm VB Subfunction

The function `askalm` asks the GPS 18-5 receiver to transmit its almanac and then stores this data in a text file

Listing C.3: Function to download almanac from rceiver.

```
Attribute VB_Name = "Module2"
Sub askalm()
port = 2
Dim t As Single, i As Integer
setNMEA

'    Board.CommMob.PortOpen = False
  '  Board.CommMob.Settings = "38400,n,8,1"
    ' Board.CommMob.PortOpen = True
'    change38400

Board.CommMob.InputLen = 0
Board.CommMob.InputMode = 0  'text

'send "$PGRMO,,2"

pause 0.2

send "$PGRMO,GPALM,1"
'For i = 1 To 5
   t = Timer
   Do
   Loop Until Timer > t + 5
'_____________________________________________
   logfile = "c:\GPSInfo\" + "NMEAalm" + ".txt"
   Open logfile For Output As 1
   Write #1, Board.CommMob.Input;
   Close 1
'_____________________________________________

  '  Debug.Print Board.CommMob.Input
'Next
Board.CommMob.InputLen = 1
Board.CommMob.InputMode = 1  'binary
Stop
End Sub
```

C.5 The NMEA2YUMA VB Subfunction

The function NMEA2YUMA converts an NMEA 0183 almanac data file to the YUMA format, and stores it in a text document.

Listing C.4: Function to change NMEA 0183 format to YUMA.

```
Attribute VB_Name = "Module1"
Option Explicit
Declare Sub CopyMemory Lib "KERNEL32" Alias "RtlMoveMemory" (
   hpvDest As Any, hpvSource As Any, ByVal cbCopy As Long)

Type nmeadata
    total As Integer          'total number of alm sentences to
       be transmitted
    prnno As Integer          'prn number
    health As Integer         'gps health
    weekno As Integer         'gps week number
    eccen As Double           'eccentricity of satellite (Hex
       XXXX)
    reftime As Double         'almanac reference time (Hex XX)
    inc As Double             'inclination angle (Hex XXXX)
    rra As Double             'rate of right ascension (Hex XXXX
       )
    roota As Double           'root of semi major axis (Hex
       XXXXXX)
    omega As Double           'argument of perigee (Hex XXXXXX)
    long As Double            'longitude of ascension node (Hex
       XXXXXX)
    meana As Double           'mean anomaly (Hex XXXXXX)
    af01 As Single            'clock parameter (Hex XXX)
    af11 As Single            'clock parameter (Hex XXX)
End Type

Public nd(32) As nmeadata
Public p, p1 As Integer
Public a As String, i As Integer, i2 As Integer

Sub getnmea()
Dim dump As Double
Open "C:\NMEAalm.txt" For Input As 1

Do Until p = 32

        Line Input #1, a
        i = InStr(a, ",")

    dump = Val(Mid(a, i + 1)) 'number of sentences in data
       file

    p = getalmdata(a, i) 'store number of current alm sentence

    nd(p).prnno = getalmdata(a, i) 'store prn number for
       satellite

    nd(p).weekno = getalmdata(a, i) 'store GPS week number
    dump = getalmdata(a, i) ' dump satellite health (00 if OK)

    'Eccentricity is a non-negative value, however the val
       function assumes it is signed
    'Need to do a small if statement

    dump = getalmdatah(a, i)    'store satellite eccentricity
    If dump < 0 Then            'value is being evaluated as
       signed, but eccentricity is not signed
        nd(p).eccen = (dump + 65536) * 4.768371582E-07
    Else
        nd(p).eccen = dump * 4.768371582E-07
    End If

    dump = getalmdatah(a, i) ' store almanac reference time
    nd(p).reftime = dump * 2 ^ 12

    dump = getalmdatah(a, i) 'store inclination angle (has 4
       characters, thus negative automatically interpreted
    nd(p).inc = 0.9424777961 + dump * 0.000005992112458

    dump = getalmdatah(a, i) 'store rate of right ascension (
       has 4 characters)
    nd(p).rra = dump * 1.142904749E-11

    'The following 4 data will not return negative numbers as
       the system seems to only do signed
    'calculations if the amount hex characters is a multiple
       of 4 and they all have 6 characters

    dump = getalmdatah(a, i) 'store root of semi major axis
    nd(p).roota = dump * 0.00048828125

    dump = getalmdatah(a, i) 'store omega, argument of perigee
    If dump <= Val("&h" + "7FFFFF") Then
        nd(p).omega = dump * 3.745070283E-07
    Else
        nd(p).omega = (dump - 16777216) * 3.745070283E-07
    End If

    dump = getalmdatah(a, i) 'store longitude of ascension
       node
    If dump <= Val("&h" + "7FFFFF") Then
        nd(p).long = dump * 3.745070283E-07
    Else
        nd(p).long = (dump - 16777216) * 3.745070283E-07
    End If

    dump = getalmdatah(a, i) 'store mean anomaly
    If dump <= Val("&h" + "7FFFFF") Then
        nd(p).meana = dump * 3.745070283E-07
    Else
        nd(p).meana = (dump - 16777216) * 3.745070283E-07
    End If

    nd(p).af01 = getalmdatah(a, i) ' store af0 clock parameter

    nd(p).af11 = getalmdatah(a, i) ' store af1 clock parameter

'********************************

    'Note need to remember that when doing negative
       calcuations, the value to take from
    'the hex value needs to be one bigger than say FFFF
       (=65535) to get -1 for ans for
    'FFFF (i.e. need to do val("&h" + FFFF) - 65536

'*********************************
Loop

Close 1

Displaysub

End

End Sub

Function getalmdata(a, i) As Double

i = InStr(i + 1, a, ",") 'jumps past current comma to find
   next comma

getalmdata = Val(Mid(a, i + 1))

End Function

Function getalmdatah(a, i) As Double

i = InStr(i + 1, a, ",") 'find next comma

getalmdatah = Val("&h" + Mid(a, i + 1))

End Function

Sub Displaysub()
Dim i, j As Integer, logfile As String, info As String

logfile = "C:\yumatest.txt"
j = FreeFile
Open logfile For Output As #j
    For i = 1 To p
    If nd(i).weekno <> 0 Then
    Print #j, "****** Week"; nd(i).weekno; " almanac for PRN-"
       ; nd(i).prnno; "******"
    Print #j, "ID:                        "; nd(i).prnno
    Print #j, "Health:                    "; nd(i).health
    Print #j, "Eccentricity:              "; nd(i).eccen
    Print #j, "Time of Applicability(s):  "; nd(i).reftime
    Print #j, "Orbital Inclination(rad):  "; nd(i).inc
    Print #j, "Rate or Right Ascen(r/s):  "; nd(i).rra
    Print #j, "SQRT(A) (m^1/2):           "; nd(i).roota
    Print #j, "Right Ascen at TOA(rad):   "; nd(i).long
    Print #j, "Argument of Perigee(rad):  "; nd(i).omega
    Print #j, "Mean Anom(rad):            "; nd(i).meana
    Print #j, "Week:                      "; nd(i).weekno
    Print #j, " "
    End If

Next

Close #j

End Sub
```

C.6 The steer1 VB Code

The code in steer1 is executed whenever a button on the Steer1 form is pressed.

Listing C.5: Code executed by buttons on form Steer1.

```
Attribute VB_Name = "Module2"
Private Sub Form_KeyPress(KeyAscii As Integer)
ainkey$ = Chr$(KeyAscii)
If ChrB$(KeyAscii) = esc Then End
End Sub

Private Sub Hload_Click()
Dim modulus As Single
Open "c:\heading.txt" For Input As 1
Input #1, normal(1), normal(2)
Close 1
modulus = Sqr(normal(1) ^ 2 + normal(2) ^ 2)
normal(1) = normal(1) / modulus
normal(2) = normal(2) / modulus
headset = 1
Cap.Caption = "Heading read from disk"
showheading
End Sub

Private Sub Hsave_Click()

If headset Then
   Open "c:\heading.txt" For Output As 1
   Write #1, normal(1), normal(2)
   Close 1
   Cap.Caption = "Heading saved to disk"
End If

End Sub

Private Sub NudgeLeft_Click()
ainkey$ = "o"
End Sub

Private Sub NudgeRight_Click()
ainkey$ = "p"
End Sub

Private Sub Quit_Click()
End
End Sub

Private Sub Recap_Click()
ainkey$ = "k"
End Sub

Private Sub Setfrom_Click()
ainkey$ = "f"
End Sub

Private Sub Setto_Click()
ainkey$ = "t"
End Sub

Private Sub Startlog_Click()

If logging = False Then
```

```
    'logfile = "c:\GPSInfo\Log-" + Mid$(Str$(Timer * 100), 2) +
        ".csv"
    logfile = "C:\Documents and Settings\Jim\My Documents\Nick'
       s Project\Project 1\GPS18-5 Stuff\GPS18-5 Software\
       TestMon\Log-" + Mid$(Str$(Timer * 100), 2) + ".csv"
    Open logfile For Output As 3
    logging = True
    Startlog.Caption = "Stop logging"
Else
    logging = False
    Close 3
    Startlog.Caption = "Start logging"
End If

End Sub

Private Sub Turn_Click()

outward = outward * -1

showheading
End Sub
```

C.7 The calcvel2 VB Subfunction

The code in `calcvel2` is executed when readings are being taken from both the fixed and mobile receivers.

Listing C.6: Code executed by 'model' to calculate position difference.

```
Attribute VB_Name = "Module2"
Sub calcvel2()
Dim j As Integer, k As Integer
Dim chan  As Integer, prn As Integer
Dim savediffc0(31) As Boolean, ev As Single

For prn = 0 To 31
    savediffc0(prn) = False
Next

getephem = 0
j = 0
For chan = 0 To 11
    prn = ph(mobile, chan).prn
    If (prn >= 0) And (prn <= 31) Then
        If valid(fixed, prn) And valid(mobile, prn) Then
            If makeephem(prn, chan) <> 0 Then
                If ephem(prn, 3) < -0.25 Then          'high in the
                    sky
                    savediffc0(prn) = True
                    If prc0saved(0, prn) And (ph(fixed, chan).slip
                        = 0) And (ph(mobile, chan).slip = 0) Then
                        vuse(j) = prn
                        j = j + 1
                    End If
                End If
            Else
                getephem = 1
            End If
        End If
    End If
Next

nvuse = j - 1

If (nvuse >= h) Then
    venough(difference) = True
    solvev difference
Else
    venough(0) = False
    For j = 0 To 3
        dxc(difference, j) = 0
    Next

End If

For prn = 0 To 31
    If savediffc0(prn) Then
        prc0(0, prn) = (cycle(mobile, prn) - cycle(fixed, prn))
            * wave
        prc0saved(0, prn) = True
    Else
        prc0saved(0, prn) = False
    End If
Next

If getephem Then requeph

End Sub
```

C.8 The `solvev` VB Subfunction

The code in `solvev` is executed whenever `calcvel2` is executed.

Listing C.7: Code executed by 'calcvel2' to find difference in position between two receivers.

```
Attribute VB_Name = "Module2"
Sub solvev(p As Integer)

Dim k As Integer, L As Integer
Dim i As Integer, j As Integer
Dim vmag As Single, vdiff(3) As Double

'if p = 0, solve for difference between two receivers
'else solve for fixed or mobile receiver

For i = 0 To 3
    vdiff(i) = 0
    For j = 0 To nvuse
        k = vuse(j)
        If p = 0 Then
            vdiff(i) = vdiff(i) + ephem(k, i) * ((cycle(mobile, k
                ) - cycle(fixed, k)) * wave - prc0(p, k)) '- Tr(k)
        Else
```

```
            vdiff(i) = vdiff(i) + ephem(k, i) * (cycle(p, k) *
                wave - prc0(p, k) - prcalc(k))  ' - Tr(k)
        End If
    Next
Next

For i = 0 To 3
    For j = 0 To 3
        square(i, j) = 0
        For k = 0 To nvuse
            L = vuse(k)
            square(i, j) = square(i, j) + ephem(L, i) * ephem(L,
                j)
        Next
    Next
Next

invert square()

vmag = 0
For i = 1 To 3
    dxc(p, i) = 0
    For j = 0 To 3
        dxc(p, i) = dxc(p, i) + square(i, j) * vdiff(j)
    Next
    vmag = vmag + Abs(dxc(p, i))
Next

If vmag > 10 Then
    venough(p) = False
    For i = 0 To 3
        dxc(p, i) = 0
    Next
End If      'velocity limit 36kph

End Sub
```

C.9 The dokeys VB Subfunction

The subfunction dokeys is executed everytime the program loops and executes commands based upon key press inputs.

Listing C.8: Code to analyse KeyPress events in software.

```
Attribute VB_Name = "Module2"
Sub dokeys()
Dim i As Integer, j As Integer
Dim k  As Integer, prn As Integer

   e = DoEvents
   a$ = inkey$
   If a$ <> "" Then Pic.Caption = a$
   Select Case a$
      Case "a"
         stopset = True
      Case "c"
         Pic.Cls
         For j = 0 To 3
           xv(0, j) = 0
           xv(1, j) = 0
           xv(2, j) = 0
           xgm0(j) = xgmob(j)
           xx0(j) = 0
           xm(j) = 0
         Next
         grid
      Case "␣"
         Board.Cls
         Pic.Cls
         grid
      Case "k"                      'kill - was z
         For prn = 0 To 31
            locked(prn) = 0
         Next
         notset = 7
         For i = 1 To 3
           ivg(i) = 0
         Next
         warmup = 0
      Case "h"
         h = 2
      Case "3"
         h = 3
      Case "e"
         requeph  'Board.ComMob.Output = askephem
         j = 0
      Case "0"
         h = 0
      Case "s"
         Pic.Cls
      Case "m"
         missed = 1
      Case "f"
         For i = 1 To 3
            xf(i) = xm(i)
         Next
         fromset = 1
         Steer1.Cap.Caption = "New␣From␣point␣set"
         If (toset = 1) And (headset = 0) Then makenormal
      Case "t"
         If headset = 0 Then
            For i = 1 To 3
               xt(i) = xm(i)
            Next
            toset = 1
            Steer1.Cap.Caption = "New␣target␣set"
            If fromset Then makenormal
         Else
            headset = 0
            Steer1.Setto.Caption = "Set␣To␣"
            Steer1.Cap.Caption = "Set␣your␣From␣point␣first"
         End If
      Case "o"  '
           sideoffset = sideoffset + 25
           Steer1.Resets.Caption = CStr(sideoffset / 100)
      Case "p"
           sideoffset = sideoffset - 25
           Steer1.Resets.Caption = CStr(sideoffset / 100)
     End Select
End Sub
```

C.10 The `makenormal` VB Subfunction

The subfunction `makenormal` generates a normalised vector of the heading vector generated after the ‘From’ and ‘To’ points have been defined.

Listing C.9: Code to generate normalised heading vector.

```
Attribute VB_Name = "Module2"
Sub makenormal()
Dim mag As Single

normal(2) = xt(1) - xf(1)
normal(1) = xt(2) - xf(2)
mag = Sqr(normal(1) ^ 2 + normal(2) ^ 2)
normal(1) = -normal(1) / mag
normal(2) = normal(2) / mag
Pic.Line (xm(2) - xx0(2), xm(1) - xx0(1))-Step(normal(2),
   normal(1)), 9
Pic.Line (xf(2) - xx0(2), xf(1) - xx0(1))-(xt(2) - xx0(2), xt
   (1) - xx0(1)), 13
Steer1.Setto.Caption = "Change heading"
Steer1.Cap.Caption = ""
Steer1.Hsave.Enabled = True
headset = 1
showheading
End Sub
```

C.11 The `showheading` VB Subfunction

The subfunction `showheading` displays the heading in terms of a compass on the guidance screen.

Listing C.10: Code to generate heading on guidance screen.

```
Attribute VB_Name = "Module2"
Sub showheading()
Steer1.compass.Cls
If headset Then
   Steer1.compass.Line (0, 0)-(-normal(1) * outward, normal(2)
       * outward), vbYellow
End If
If outward > 0 Then
   Steer1.Outback.Caption = "Outward"
```

```
Else
    Steer1.Outback.Caption = "Return"
End If
End Sub
```

C.12 The `model` VB Subfunction

The subfunction `model` calculates the variables side and travel and is what produces the guidance line on the computer screen.

Listing C.11: Code that calculates guidance data.

```
Attribute VB_Name = "Module2"
Sub model()  'only called when notset=0 and mobile posmessage
Dim i As Integer, j As Integer, side As Single, Travel As
    Single
Dim xe(3) As Single, emag As Single
Dim xp(3) As Single, dett(2) As Single
Dim nsat As Integer

'If you've got corresponding readings for mobile and fixed
' calulate dxc(difference,j)
If (Abs(tow(fixed) - tow(mobile)) < 0.5) Then
    calcvel2 'calculate move increment differentialy
    Board.Print "diff␣sats␣", nvuse + 1; venough(difference),
    For j = 0 To nvuse
        Board.Print vuse(j);
    Next
    Board.Print , "␣Det␣"; determinant
    dett(difference) = determinant
Else
    venough(difference) = False
End If

calcvel mobile
' calulate dxc(mobile,j) from mobile data alone
If venough(mobile) Then

Board.Print "mobile␣sats␣", nvuse + 1; venough(mobile),
For j = 0 To nvuse
    Board.Print vuse(j);
Next
Board.Print , "␣Det␣"; determinant
dett(mobile) = determinant

calcvel fixed 'checks newmeas
'calulate dxc(fixed) from fixed data alone
Board.Print "fixed␣sats␣␣", nvuse + 1; venough(fixed),
For j = 0 To nvuse
    Board.Print vuse(j);
Next
Board.Print , "␣Det␣"; determinant
dett(fixed) = determinant
```

```
Board.Print "Lat␣&␣Long␣␣", xgmob(1) - xgm0(1); -xgmob(2) +
    xgm0(2)

For i = 0 To 2
    If venough(i) Then
        For j = 1 To 3
            xv(i, j) = xv(i, j) + dxc(i, j)
        Next
    End If
Next

If venough(difference) Then
    For i = 1 To 3
        xm(i) = xm(i) + dxc(difference, i)
    Next
    Steer1.BackColor = vbBlack
ElseIf venough(mobile) Then
    For i = 1 To 3
        xm(i) = xm(i) + dxc(mobile, i)
    Next
    Steer1.BackColor = vbBlue
    Else
    Steer1.BackColor = vbMagenta
End If

For i = 1 To 3
    Board.bxx(i) = Format$(xm(i), "##0.000")
Next

If logging Then
    Write #3, venough(difference);
    For j = 1 To 3
        Write #3, xm(j);
    Next
    Write #3, nvuse + 1, nuse
End If
Board.Print "Sats␣used␣", "elevation", "azimuth", "strength"
nsat = 0
For j = 0 To 11
   Board.ForeColor = vbRed
   If (satdat(j).status And 4) Then
       Board.ForeColor = vbBlack
       nsat = nsat + 1
   End If
   Board.Print satdat(j).prn, satdat(j).eldeg, satdat(j).azdeg,
       satdat(j).snr
   'Board.Print almsatdat(j).prn, almsatdat(j).elevation,
       almsatdat(j).azimuth
Next
Board.ForeColor = vbBlack
Board.Print

End If

' ****** Adding steering code to model

If nvuse <> 0 Then
Steer1.sats.Caption = "Sats␣" + Str$(nvuse + 1)
Steer1.det.Caption = "Det␣=␣" + Format$(determinant, "##0.000"
    )
End If

If headset And fromset Then
```

```
    side = (xm(1) - xf(1)) * normal(1) + (xm(2) - xf(2)) *
       normal(2)
    Travel = (xm(1) - xf(1)) * normal(2) - (xm(2) - xf(2)) *
       normal(1)
    side = side * outward
    sideoffset = sideoffset * outward
    side = side + (sideoffset / 1000)

    side = side - 24 * Int((side + 12) / 24)

    Steer1.SideData.Caption = CStr(side) + "␣␣␣␣" + CStr(
       outward)
    Travel = Travel * outward
    Steer1.Cap.Caption = "Track␣error␣(mm)␣␣" + Format$(side *
        1000, "######0")
    Steer1.Travel.Caption = "Travel␣" + Format$(Travel, "
       #####0.00") + "␣m"
ElseIf fromset Then
    Steer1.Cap.Caption = "Set␣target␣direction␣with␣'To'"
ElseIf toset Or headset Then
    Steer1.Cap.Caption = "Set␣starting␣point␣with␣'From'"
Else
    Steer1.Cap.Caption = "Set␣starting␣point,␣target␣direction
       "
End If

If outward > 0 Then
   Steer1.Outback.Caption = "Outward"
Else
   Steer1.Outback.Caption = "Return"
End If

Steer1.Cls

If fromset And headset Then
   Steer1.Line (0, 0)-(-side, steerscale)
   Steer1.Line (-side, 0)-(-side, steerscale), vbRed
   Steer1.PSet (0, 1.1 * steerscale), vbGreen
End If
If fromset Then
   Steer1.Dist.Caption = "Distance␣" + Format$(Sqr((xm(1) - xf
      (1)) ^ 2 + (xm(2) - xf(2)) ^ 2), "#####0.0") + "m"
End If

tlast = Timer

' ****** End steering code in model

newmeas(fixed) = False
newmeas(mobile) = False
newpos(fixed) = False
newpos(mobile) = False

For i = 0 To 2
   venough(i) = False
Next

Pic.PSet (xv(difference, 2) - xx0(2), -xv(difference, 1) + xx0
   (1)), vbYellow
Pic.PSet (xv(mobile, 2) - xx0(2), -xv(mobile, 1) + xx0(1)),
   vbRed
```

```
Pic.PSet (xv(fixed, 2) - xx0(2), -xv(fixed, 1) + xx0(1)),
   vbGreen
Pic.PSet (xm(2) - xx0(2), -xm(1) + xx0(1)), vbWhite
Pic.PSet (xv(fixed, 2) - xx0(2), -xv(fixed, 1) + xx0(1)),
   vbGreen
Pic.PSet (xgmob(2) - xgm0(2), -xgmob(1) + xgm0(1)), vbBlue

tlast = Timer
End Sub
```

C.13 The Motorsim VB Function

The function Motorsim was used to simulate the operation of a simple motor control algorithm for hands-free guidance.

Listing C.12: Function to simulate guidance motor control.

```
Attribute VB_Name = "Module2"
Option Explicit
Public pos As Integer
Dim outdat As String
Dim MIC As Integer

Private Sub Form_KeyPress(KeyAscii As Integer)

Dim Inkey$
Dim ainkey$

ainkey$ = Chr$(KeyAscii)
Inkey$ = ainkey$
ainkey$ = ""

If Inkey$ <> "" Then
     Inkey1.Caption = Inkey$
     Select Case Inkey$
          Case "d"
               If pos > -3 Then
               pos = pos - 1
               Position.Caption = CStr(pos)
               send (0)
               End If

          Case "f"
               If pos < 3 Then
               pos = pos + 1
               Position.Caption = CStr(pos)
               send (1)
               End If

     End Select
End If

End Sub

Sub send(port As Integer)

Select Case port
```

```
    Case 0
        Do
            DoEvents
        Loop Until MC1(port).OutBufferCount = 0
        MC1(port).Output = outdat
        MIC = MIC - 1
        MIC1.Caption = CStr(MIC)

    Case 1
        Do
            DoEvents
        Loop Until MC1(port).OutBufferCount = 0
        MC1(port).Output = outdat
        MIC = MIC + 1
        MIC1.Caption = CStr(MIC)

End Select

End Sub

Private Sub Form_Load()

MC1(0).CommPort = 7
MC1(0).Settings = "300,n,8,1"
MC1(0).PortOpen = True

MC1(1).CommPort = 2
MC1(1).Settings = "300,n,8,1"
MC1(1).PortOpen = True

'outdat = Val("&h" + "FFFFFFFFFFFF")
outdat = "FFFF"

End Sub
```

Appendix D

Satellite Orbit Data, Drift Testing and Guidance Trajectories

D.1 Comparison of calculation of satellite orbit parameters

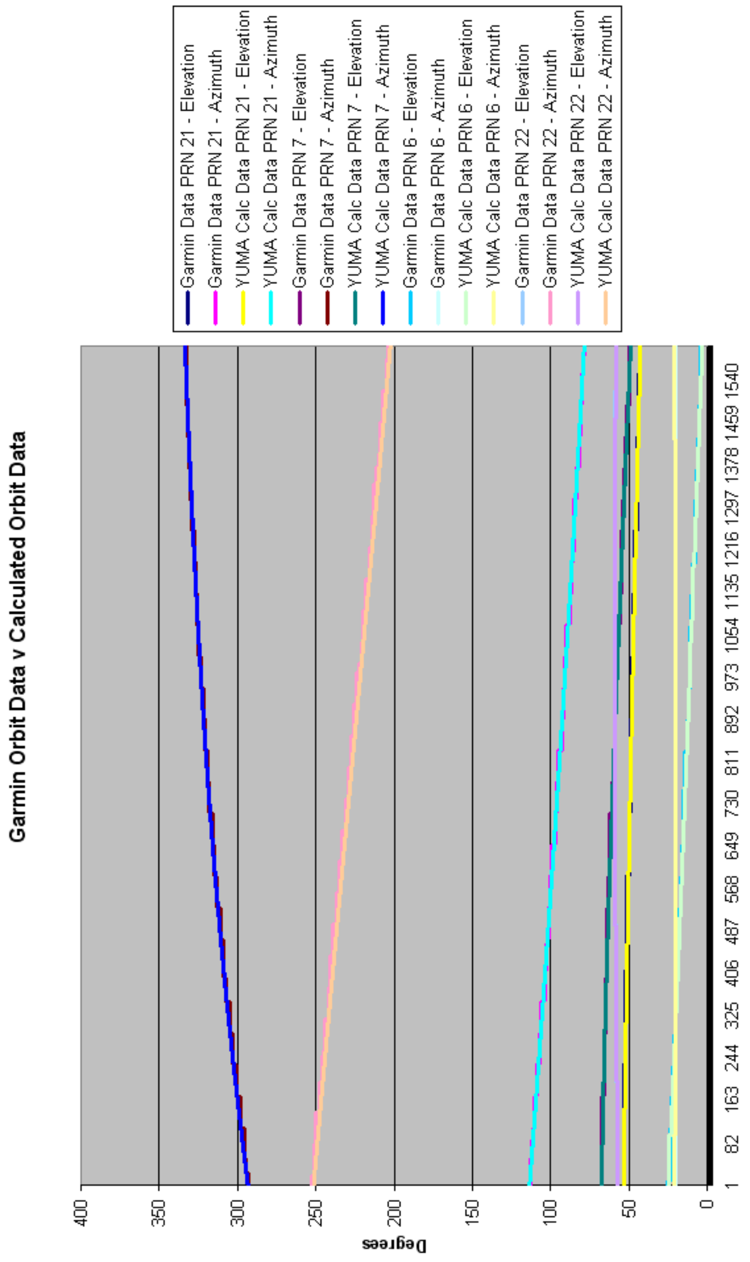
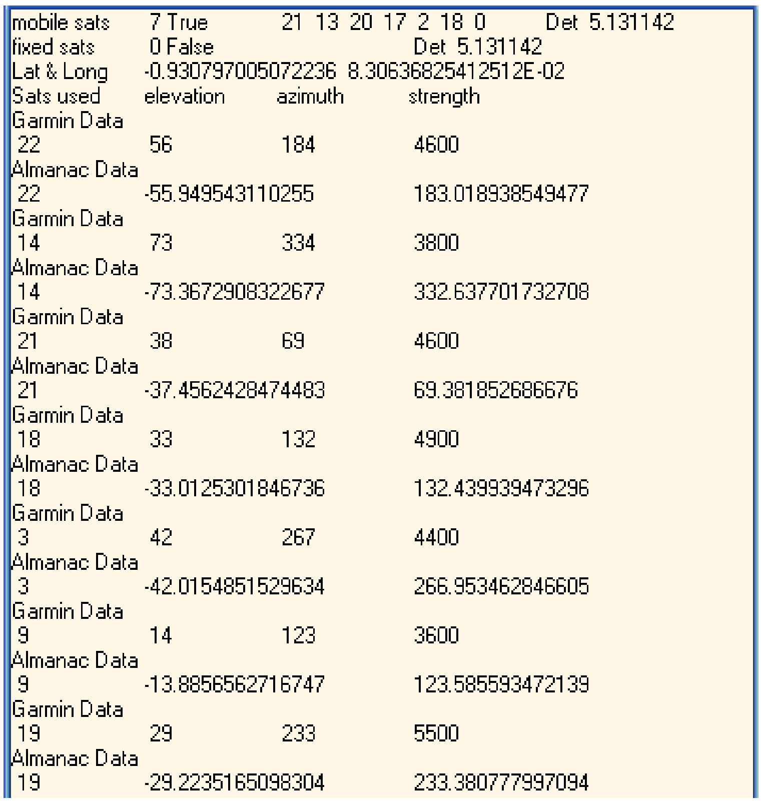


Figure D.1: Orbit Parameters of Satellites calculated by Garmin and from almanac data

Note: The elevation is negative because the angle is being taken from satellite to receiver, whereas the Garmin elevation is taken from receiver to satellite.



```
mobile sats     7 True          21  13  20  17  2  18  0        Det  5.131142
fixed sats      0 False                        Det  5.131142
Lat & Long     -0.930797005072236  8.30636825412512E-02
Sats used      elevation        azimuth         strength
Garmin Data
 22             56              184             4600
Almanac Data
 22            -55.949543110255                 183.018938549477
Garmin Data
 14             73              334             3800
Almanac Data
 14            -73.3672908322677                332.637701732708
Garmin Data
 21             38              69              4600
Almanac Data
 21            -37.4562428474483                69.381852686676
Garmin Data
 18             33              132             4900
Almanac Data
 18            -33.0125301846736                132.439939473296
Garmin Data
 3              42              267             4400
Almanac Data
 3             -42.0154851529634                266.953462846605
Garmin Data
 9              14              123             3600
Almanac Data
 9             -13.8856562716747                123.585593472139
Garmin Data
 19             29              233             5500
Almanac Data
 19            -29.2235165098304                233.380777997094
```

Figure D.2: Software representation of comparison of satellite orbit calculations

D.2 Small-scale testing data for GPS 35 Guidance System

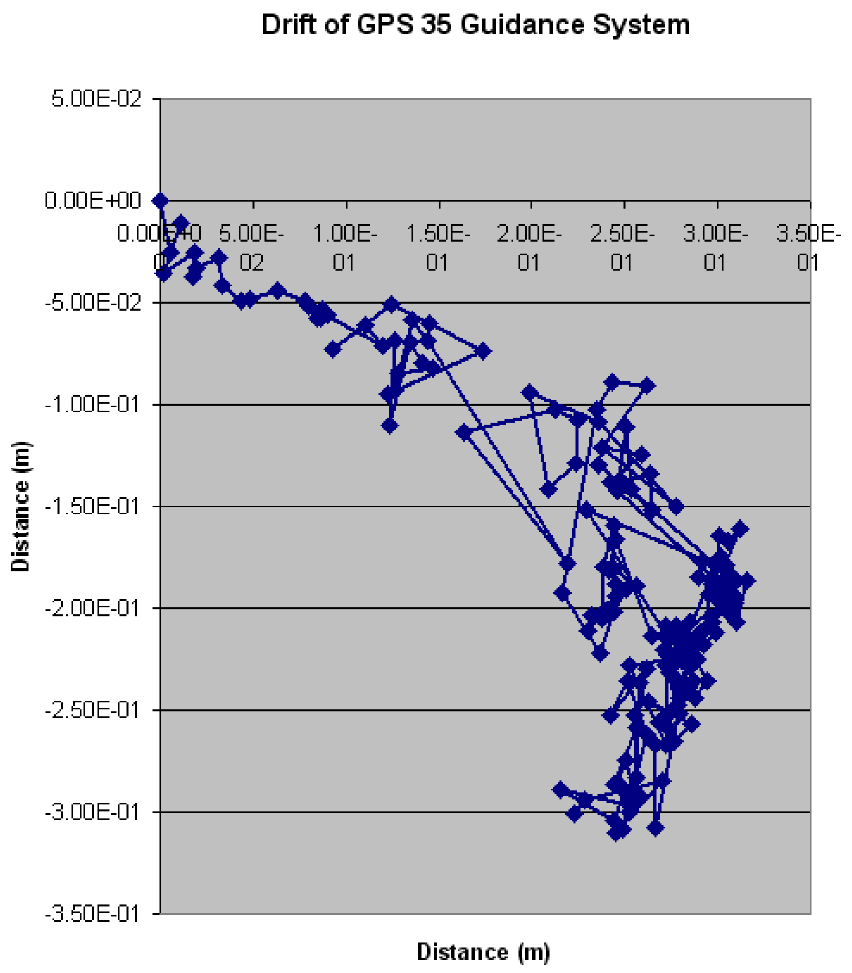


Figure D.3: Drift of mobile receiver in GPS 35 system

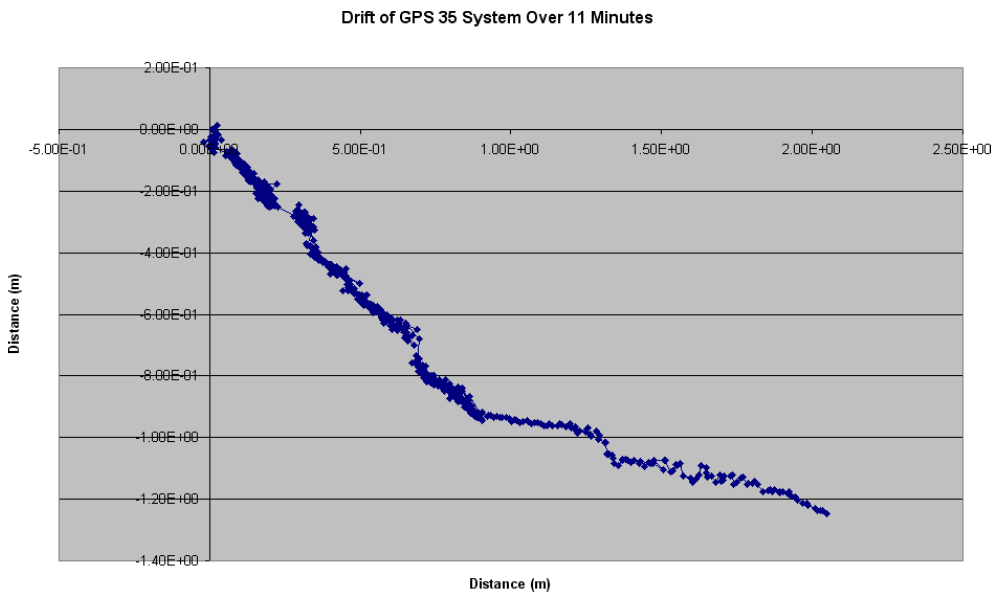


Figure D.4: Drift of mobile receiver in GPS 35 system over 11 minutes

D.3 On-farm testing data for GPS 35 Guidance System

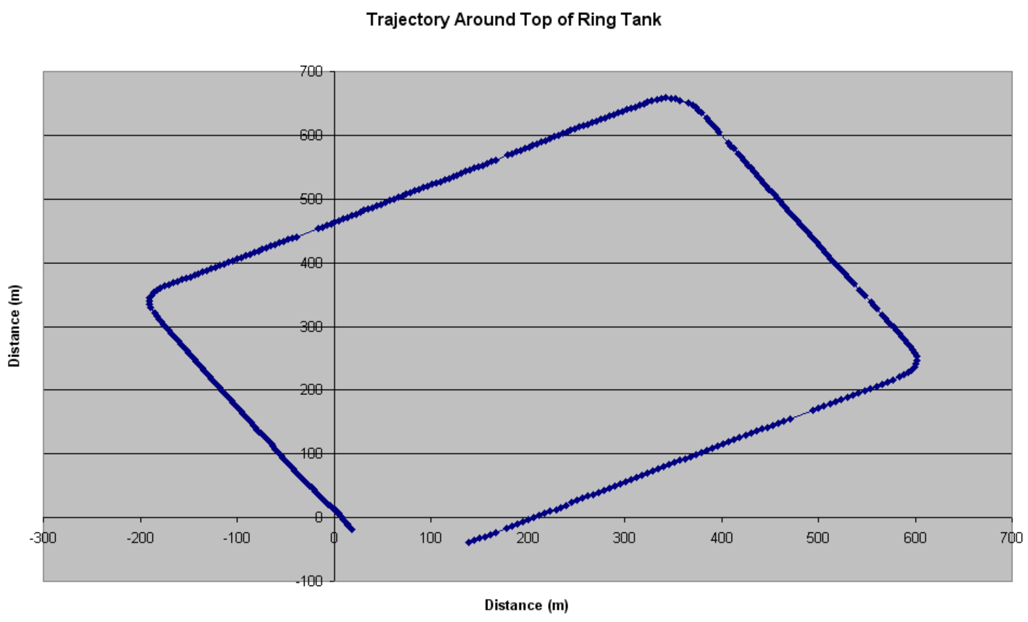


Figure D.5: Trajectory of path around top of farm ring tank

The following table shows the GPS-calculated distance from one side of the ring tank to the other. The ring tank is approximately 650m in length.

Outward Trajectory		Return Trajectory		Swath Width
x	y	x	y	(m)
-104	183	443	525	645
-178	306	397	605	648

Table D.1: Table of ring tank length

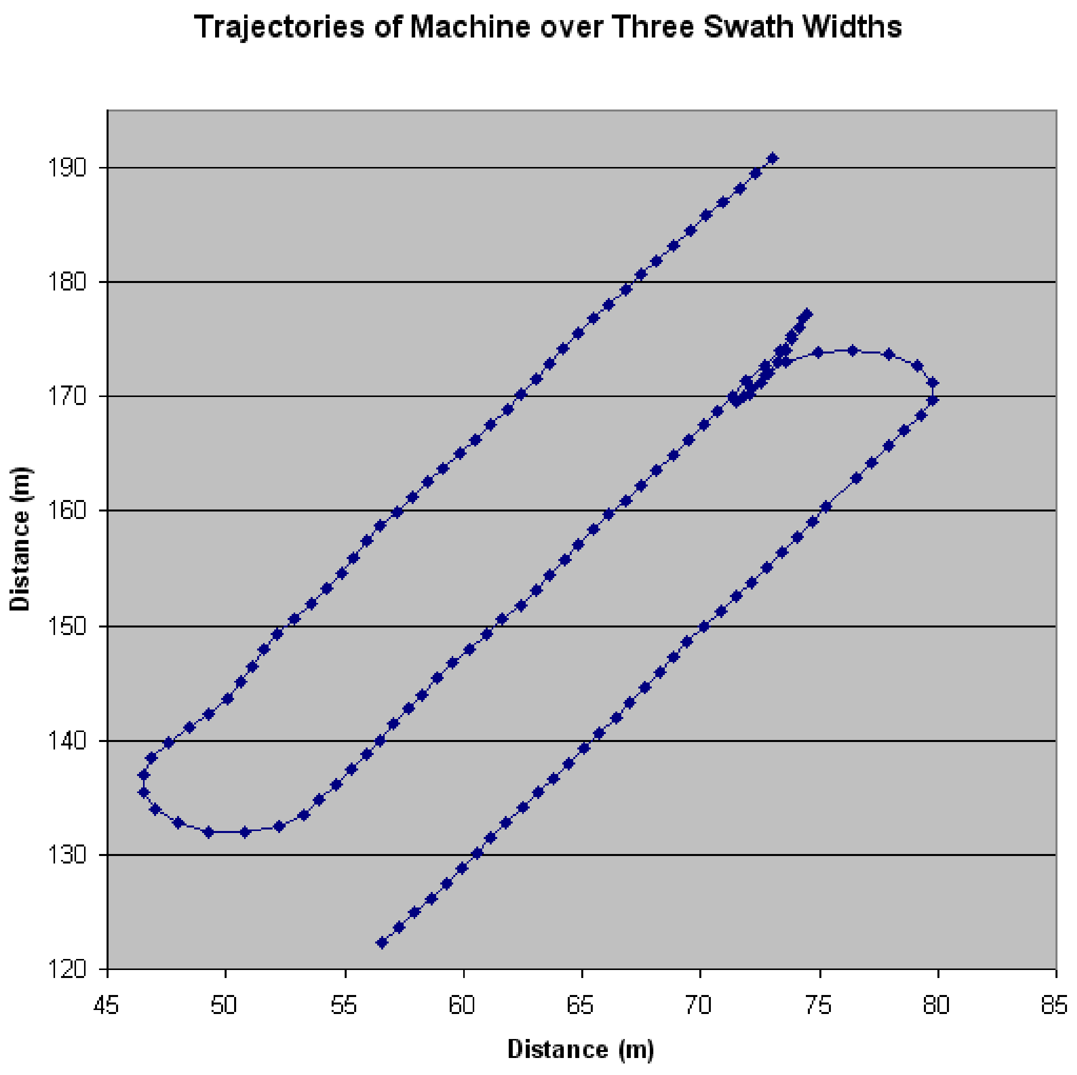


Figure D.6: Trajectory of machine across swath widths

D.4 Data transmission setup for GPS 18-5 System

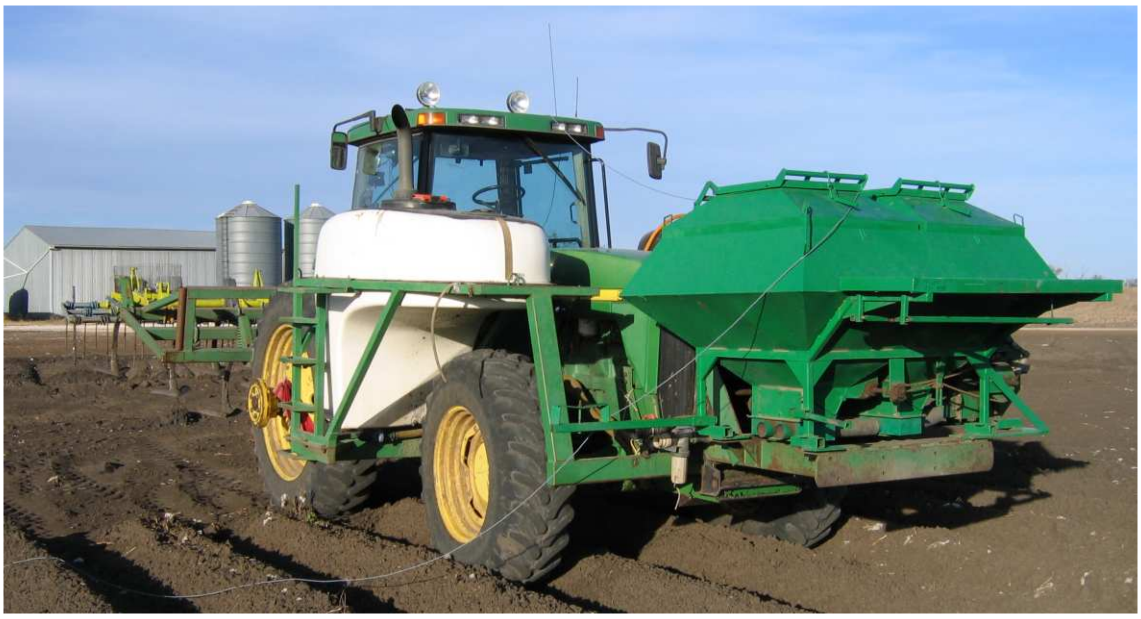


Figure D.7: Set up of transmission cable on tractor

D.5 Small-scale testing data for GPS 18-5 Guidance System

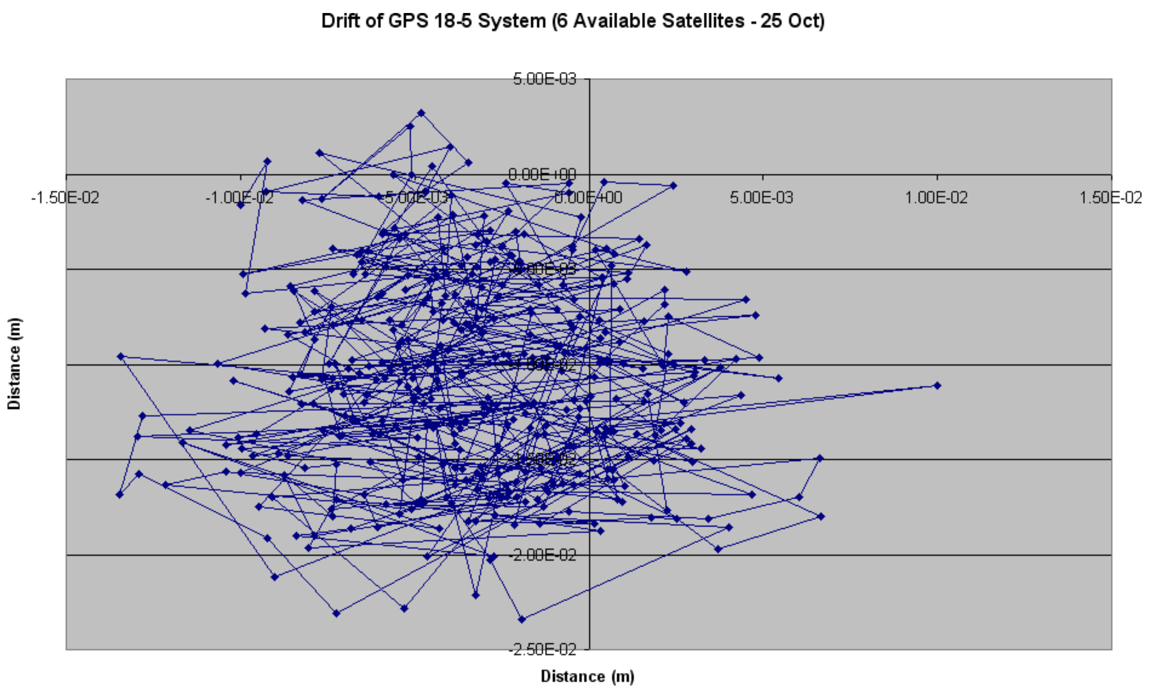


Figure D.8: Trajectory of machine across swath widths

In the above figure, the system drifts a total of 2.5cm in the x-direction and three centimetres in the y-direction confirm the accuracy of the system being within the ±2cm requirement.

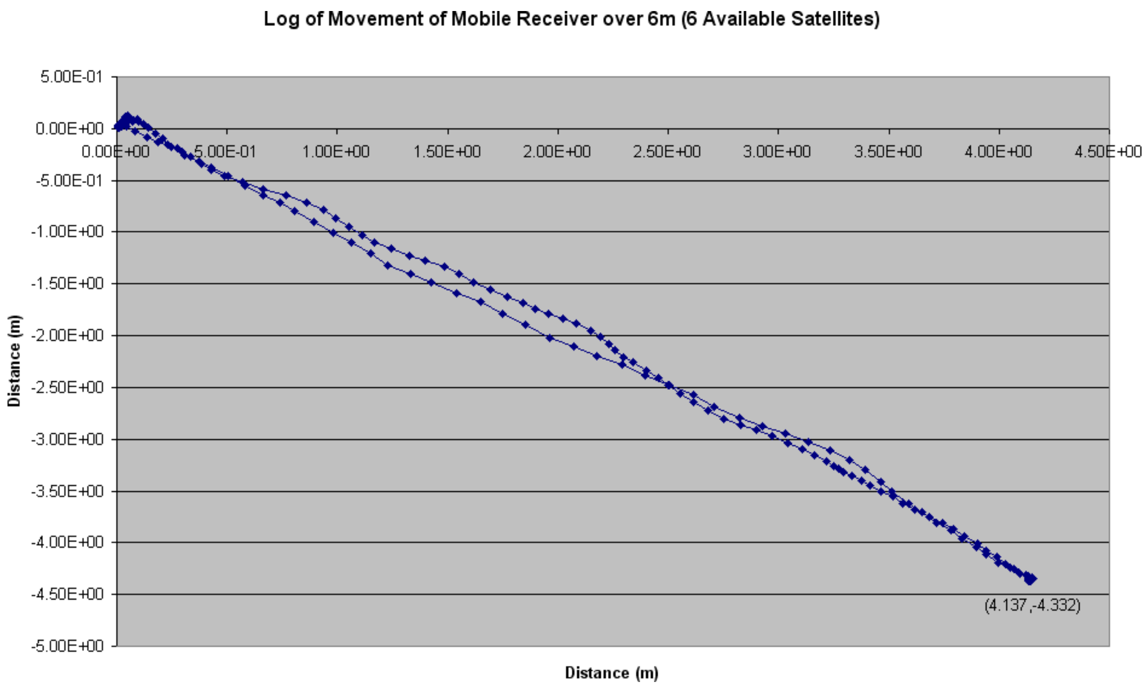


Figure D.9: Trajectory of machine across swath widths

Figure D.9 portrays the system logging the mobile receiver position as it is moved over a distance of six metres, and back again.

If coordinates of the origin point are assumed to be very close to zero and the coordinates of the end point are assumed to be in the middle of the block of data points, the length of the line can be calculated. The calculation is shown below.

$$x = \sqrt{4.137^2 + (-4.332)^2} \tag{D.1}$$

$$x = 5.99m$$

D.6 On-farm testing data for GPS 35 Guidance System

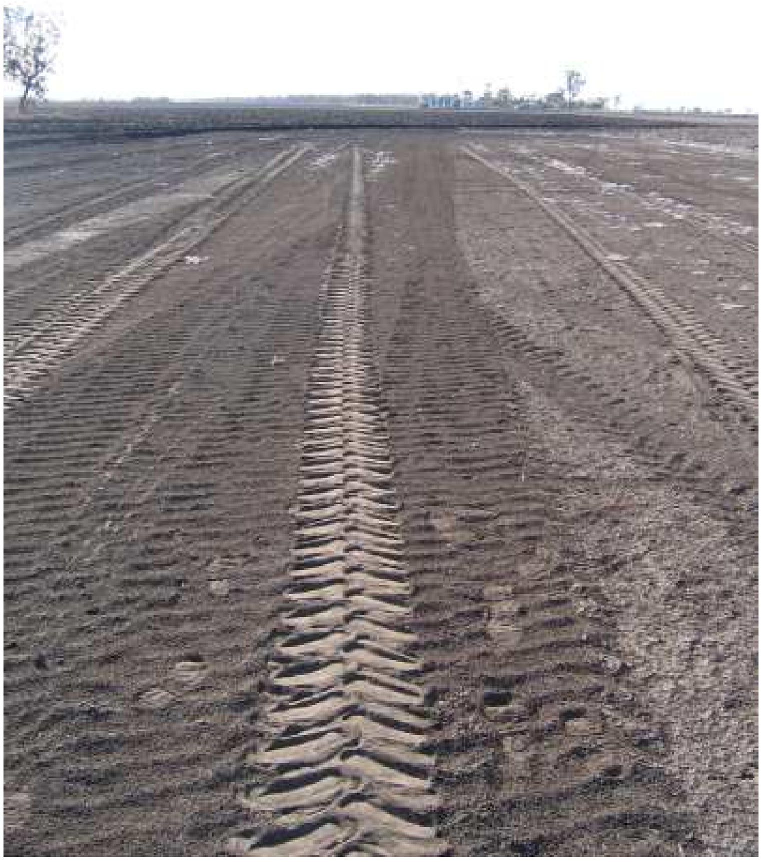


Figure D.10: Tyre path of tractor under guidance

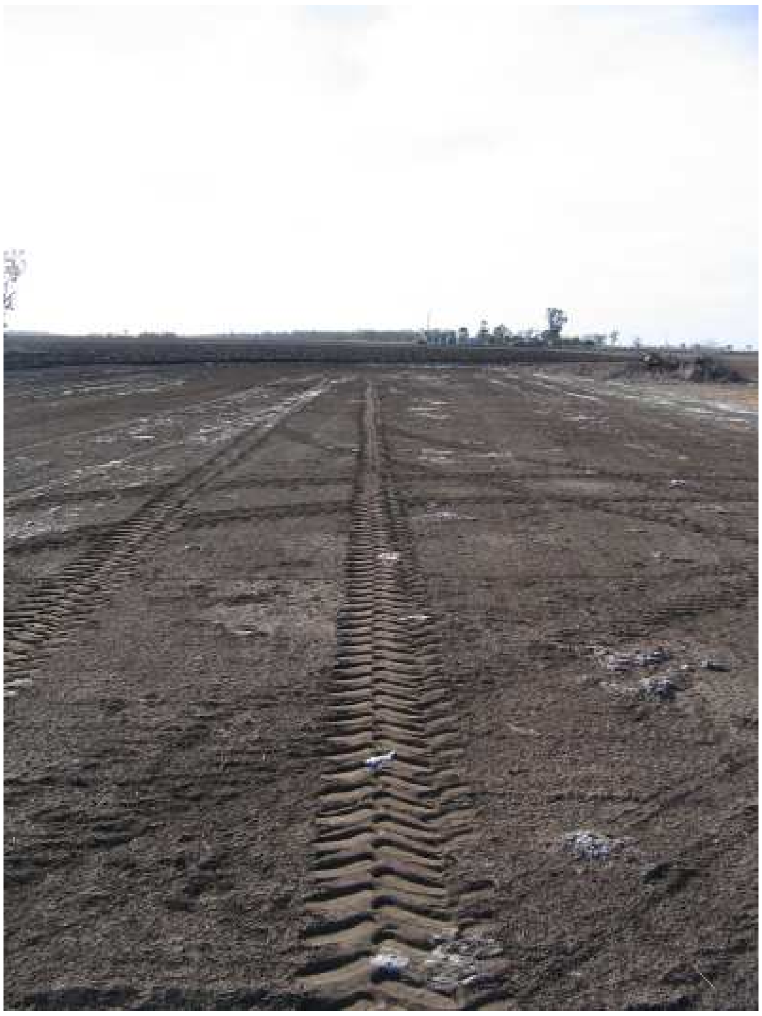
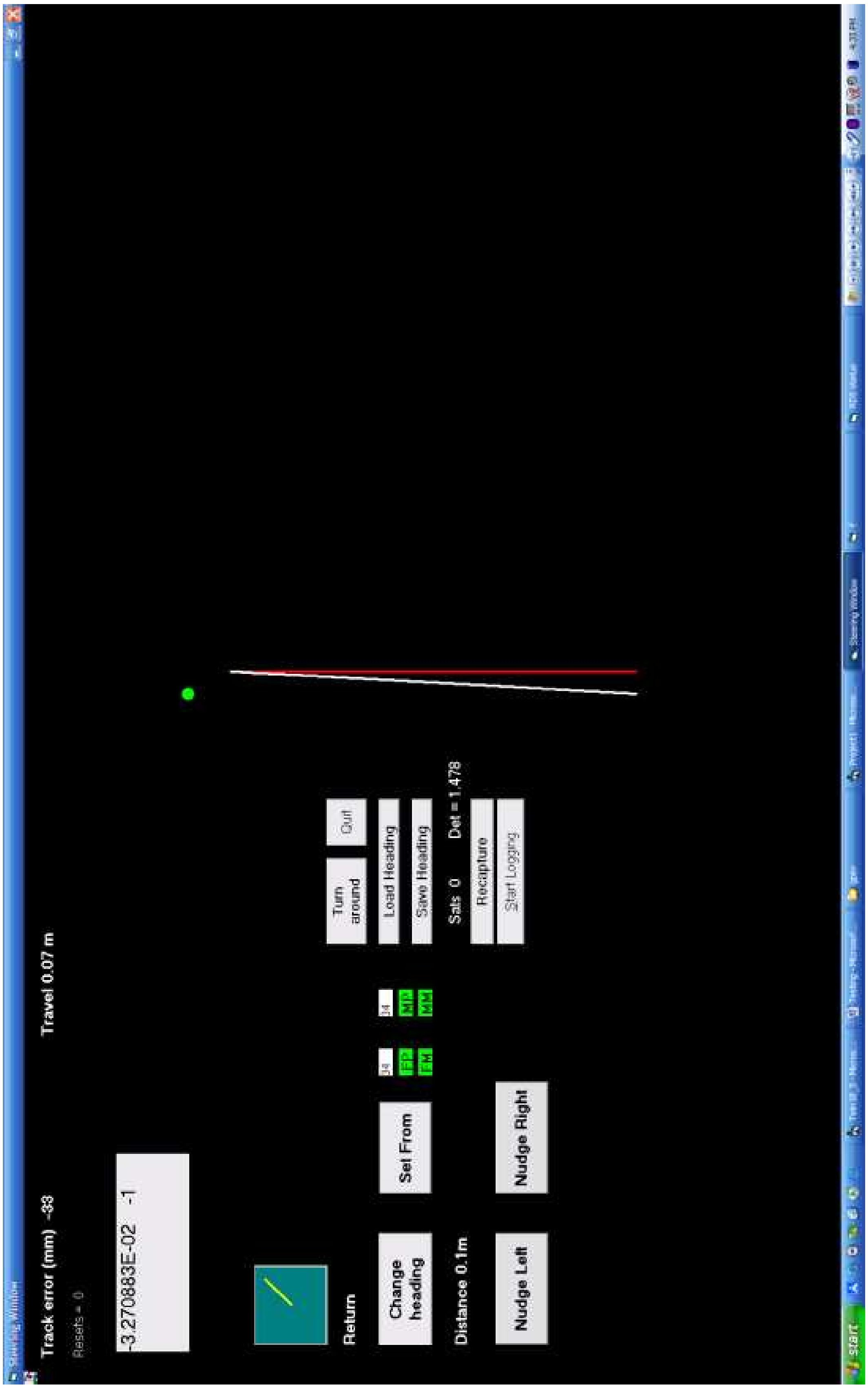


Figure D.11: Tyre path of tractor under guidance


Track error (mm) -33

Travel 0.07 m

Resets = 0

-3.270883E-02 -1

Distance 0.1m

Sats 0 Det = 1.478

Figure D.12: Screenshot of steering screen while guidance is operating

Note: screenshot was taken on a laptop at widescreen resolution.