ISSN 0280-5316
ISRN LUTFD2/TFRT--5802--SE

Improved Methods for GPS Navigation

Henrik Lundstedt

Department of Automatic Control
Lund University
October 2007

Lund University
Department of Automatic Control
Box 118
SE-221 00 Lund Sweden

Document name
MASTER THESIS

Date of issue
October 2007

Document Number
ISRNLUTFD2/TFRT--5802--SE

Author(s)
Henrik Lundstedt

Supervisor
Anders Görtz at Wayfinder Systems AB in Lund
Toivo Henningsson at Automatic Control in Lund
Karl-Erik Årzén Automatic Control in Lund (Examiner)

Sponsoring organization

Title and subtitle
Improved Methods for GPS Navigation (Förbättrade metoder för GPS-navigering)

Abstract
This Master's Thesis presents the development of methods for improved GPS navigation on Wayfinder Navigator, a mobile phone based navigation system developed by Wayfinder Systems. Focus has been put on off track detection, i.e. to detect a user diverging from the route given by the navigation system, and how the GPS position is presented to the user by the application. Improving off track detection includes both lowering the detection time and simultaneously maintaining robustness in order to avoid false alarms due to different types of error sources.
The work has resulted in two new off track detection algorithms, one using only GPS data and one using both GPS data and crossroad data, i.e. data describing roads crossing the route of navigation. The former has been implemented in the Wayfinder Navigator client.

Keywords

Classification system and/or index terms (if any)

Supplementary bibliographical information

ISSN and key title
0280-5316

ISBN

Language
English

Number of pages
61

Recipient's notes

Security classification

Contents

Preface

First of all, I would like to thank my supervisors, Toivo Henningsson at the Department of Automatic Control, Lund University and Anders Görtz at Wayfinder Systems for great support and guidance throughout this project. From Wayfinder Systems, I would also like to specially thank Stefan Bengtsson and Jesper Trägårdh for helping me defining the project and sharing important knowledge and experiences from the Wayfinder system, which has been essential for the result of this work.

Chapter 1

Introduction

In recent years the use of GPS[1] for positioning purposes has become increasingly common and today the technology is wide spread. Popular application areas are for example navigation, map making and military industry, i.e. areas where high precision positioning is required. Particularly the land navigation segment has expanded a lot the last years which can be explained by the general computer technology development. Modern GPS navigation systems feature graphical real time map navigation supported by voice instructions. Routes from start to destination are automatically calculated by the system. The user friendliness makes navigation easy and together with a general drop in prices this has led to an increasing demand for GPS navigation products.

It is possible to distinguish between two main types of land based navigation systems: on-board and off-board systems. On-board systems are independent systems with a large memory and processing capacity, which enables them to store both address databases and maps and to perform route calculations. They are typically embedded systems in cars or bought as stand alone products dedicated for car navigation. Advantages with these types of systems are high reliability and fast map processing, since data is stored on local memory. The main disadvantage is the need of frequent map updates as this information continuously changes.

The other approach is off-board navigation, which is a mobile solution usually implemented on mobile phones and PDAs[2]. The idea is to use the built in graphics and sound support of these devices to present the navigation. Memory and processing intensive tasks such as maintaining the map data and making route calculations are moved to a server. The mobile devices act as clients, communicating with the server to receive data, which makes the system flexible and less expensive than on-board solutions. The main drawback is the server dependency.

An important quality indicator for any land based navigation system is how it is experienced by the end user. Users driving a car, guided by a navigation system, must be presented with accurate and intuitive information about the ongoing navigation. If the information given by the navigation system does not match what is experienced by the user, this might lead to confusion and frustration in an already stressful environment, causing drivers to make mistakes. To keep focus and avoid irritation, one must be able to completely rely on the information given by the navigation system without being distracted.

[1] The Global Positioning System - A world wide satellite based navigation system

[2] Personal digital assistants

The aim of this thesis is to develop and implement methods that improve the car navigation experience for users of Wayfinder Navigator — an off-board navigation solution for mobile phones and PDAs developed by Wayfinder Systems. The thesis focuses on two important issues which strongly affect the navigation experience: off track detection and how the GPS position of the car is presented to the user. It will be shown that these issues are strongly connected.

Off track detection is concerned with determining if the driver during navigation diverges from the route leading to the destination, and presenting this information to the driver. The off track detection needs to be both fast and reliable. It needs to be fast to quickly give feedback to the driver. A driver coming off track must be alerted and get new instructions on how to continue the navigation as soon as possible. Clearly, this alert can only be given after the user is detected as being off track, which motivates a fast responding off track detection algorithm. On the other hand, off track should only be signaled if the user is in fact off track. Different error sources make the GPS position diverge from the route. The system must be robust enough to sustain such errors to avoid false off track signaling.

The second issue to be considered in this thesis is how the GPS position of the car should be presented to the driver. To obtain a good navigation experience for the drivers, it is desired to display the position on the route, as long as the driver is believed to be on track, even if errors in both GPS signal and map data bring the position outside the route.

1.1 Brief Description of Existing Implementations

The existing off track detection algorithm of Wayfinder Navigator relies solely on distance calculations between the current GPS position and the route. Off track is signaled if the distance of the user's GPS position from the route exceeds a fixed threshold during a fixed amount of time. This approach suffers from slow response since the threshold is set quite high. Still it is not enough to avoid false off track signaling, which also remains a problem.

In the existing implementation, the position of the car on the map is always presented as the position given by the GPS signal. Due to map and GPS errors this position is not always on the route. The discrepancy worsens the impression of the application and may cause great confusion if the position is displayed on the wrong side of a freeway or even on another road.

1.2 Objective

The goal of this Master's Thesis is to improve the car navigation experience for a Wayfinder Navigator user. This is to be done by developing a new off track detection algorithm that offers fast response to off track situations, but still is robust enough to avoid false off track signaling. It is also necessary to improve the consistency between how the GPS position is presented by the system and the real position experienced by the driver.

1.2.1 Goal Formulation

The following items state the goals set up for this project:

- False off track signaling should be avoided

- An off track situation should be detected and presented to the user within 2-3 seconds
- The user's GPS position should be presented as being on the route as long as the user is on track, so called snap-to-route.

1.2.2 Project Plan

As described, the existing off track detection implementation uses only the distance to the route to determine if the driver is off track. To achieve the goals of the project it is necessary to use more of the data supplied by the GPS signal. The first approach will therefore be to develop a new off track detection algorithm, which makes better use of the GPS signal data.

As a complement, it should be examined if use of crossroad information, i.e. roads crossing the route can be used to improve off track detection. A second algorithm using crossroad data should be developed to analyze this. Currently no support for crossroads is provided by the Wayfinder Navigator system. Comparing the benefits to the cost of implementing such support will be a part of this project.

The following project plan states in chronological order the major tasks to be performed during the project:

1. Use Matlab to design two new off track detection algorithms. One using existing GPS data, and one using crossroad data.
2. Perform simulations to compare the two methods and analyze the results.
3. Choose the most suitable solution and implement it in the Wayfinder Navigator J2ME[3] client.

1.3 Outline

First, background information on GPS is given in Chapter 2. In Chapter 3, the Wayfinder Navigator system is presented from both an implementation and user's point of view. Given this background information, Chapter 4 contains a problem discussion including the problems associated with the existing solution. Chapter 5 describes the methodology used during the development of the new algorithms and the final result, including both algorithm structures and design motivations. Chapter 6 presents the results of the simulations carried out during the development work. Finally, in Chapter 7, conclusions and future work are presented.

[3]Java 2 Platform, Micro Edition - a set of Java APIs for the development of software for resource-constrained devices, e.g. mobile phones

Chapter 2

The Global Positioning System

The Global Positioning System (GPS) is a worldwide localization system, built up by 29 satellites (24 operational and 5 backups). Unlike land based navigation system, which may suffer problems like dependency on weather or geographic situation, GPS will provide positioning of high accuracy at any time, anywhere in the world.

The system was originally developed by the United States Department of Defense (DoD) for military purposes. Today GPS is available for civilian use and receivers can nowadays be built with just a few integrated circuits. This has lead to an explosive development of new application areas, where GPS navigation is among the dominating. The following sections will briefly describe the most important parts of GPS. For further reading, see [1][2][3][4][5].

2.1 Trilateration - How to Determine a Position

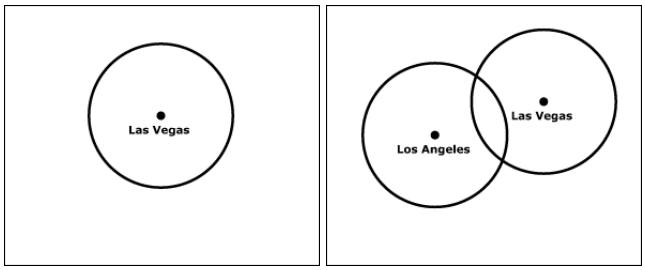
GPS uses distance calculations for position determination. The reason behind having 24 operational satellites in orbit is to guarantee that a minimum of four satellites are visible from any point on Earth. It turns out that this is the minimum requirement to obtain a precise three-dimensional location. These are calculated using a method known as *trilateration*, which is explained by the following example.

Assume that one knows the distance D to a fixed location. This means that the position can be anywhere on the circle with radius D and center at the fixed location as illustrated in Figure 2.1(a). If the distance to a second location is known the number of possible positions will be reduced to two, where the circles intersect, see Figure 2.1(b). Finally, if the distance to a third location is known, the exact position can be obtained, see Figure 2.1(c).

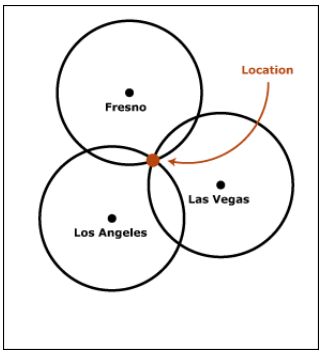
A similar discussion can be used to explain how GPS works, but instead of circles each satellite defines possible positions as the surface of a sphere, centered at the satellite. Finding the intersections of three spheres will actually give two points in space, but one of them can be rejected as a false location by additional calculations. In fact a fourth satellite will be required to achieve acceptable accuracy. This will be discussed below.

2.2 Measuring the Distance

GPS determines the distance between satellite and receiver by measuring the travel time of the GPS signal. Since radio waves travel with the speed of light, the distance can be cal-



(a) One known location (b) Two known locations



(c) Three known locations

Figure 2.1: Positioning using trilateration

culated if the time difference between transmitting and receiving the signal is known. To measure the time difference, each satellite is equipped with an atomic clock to provide a very accurate time reference. GPS receivers whose clocks are of less accuracy, continuously synchronize their clocks with the satellites to have exactly the same time. In order to synchronize, the receiver requires connection with a minimum of four satellites. If the receiver's clock is perfectly synchronized, the ranges from all four satellites will intersect in a single point, which will be the receiver's position. On the other hand, if clocks are not synchronized, the fourth measurement will not intersect with the first three. The receiver will then calculate a correction factor which will cause all measurements to intersect and use to it to correct its clock. This motivates the need of a fourth GPS satellite to determine the location.

When receiving the signal, the time codes of receiver and satellite are compared to obtain the time difference, which is used to calculate the distance. It is apparent from the above discussion that the accuracy of the distance, and thereby the accuracy of the position, is strongly dependent on how well the time difference can be measured. Precise time measurement is therefore crucial for GPS to work.

2.3 Satellite Positions

To be able to locate its own position, the GPS receiver not only requires the distance to the satellites, but also their positions. Since the satellites orbit at an altitude high enough to avoid most effects of the atmosphere, their movement can be described with fairly simple mathematical models. Using these models one can determine the position of a satellite at

any given time. This information, known as an *almanac*, is preprogrammed into every GPS receiver. To improve the accuracy further the satellites are continuously monitored using radar to spot errors compared to the expected orbit. These so called *ephemeris errors* are caused by gravity from the sun and the moon as well as solar radiation. The exact position is transmitted to the satellite, which includes the corrected position information in its GPS signal.

2.4 GPS Errors

Although GPS is designed to give optimal accuracy, there are still several types of errors affecting the result. Errors originate from a number of different sources and GPS receivers need to take them into account to achieve acceptable performance. The most important sources of error are breifly discussed below.

2.4.1 Atmospheric Conditions

As the GPS signal passes through the atmosphere, it is refracted and looses a bit of velocity. As a result the distance calculations will give errors, if assuming that the signal is traveling with the speed of light. One way to overcome the problem is through mathematical models of the atmosphere, which can be used to compensate this error. A more sophisticated solution is to use two different carrier frequencies of the GPS signal and observing the fact that low-frequency signals get refracted (slowed down) more than high-frequency signals. By comparing the two signals, the error can then be compensated for.

2.4.2 Errors Caused by the Satellite

Even in the satellites it is impossible to obtain perfect accuracy. The atomic clocks, although very precise are not perfect. In the same way, monitoring of the satellites' positions cannot detect all ephemeris errors. Finally there will always be noise acting on the GPS signal transmitted by the satellite.

2.4.3 Selective Availability

Selective Availability (SA) was intentional errors introduced in the civilian GPS signal by slightly altering the satellite's clocks and the orbital data. These factors together introduced a position error of roughly 100 meters, with the argument to protect security interests. By May 2000 a decision to end SA was announced by the U.S. government.

2.4.4 Multipath

A common error source in urban areas with tall buildings and trees is so called *multipath*. This happens when the GPS signal, before reaching the receiver bounces off a reflective surface. The reflected signals may confuse the receiver, giving erroneous measurements since the signal will travel longer than expected. In most cases both direct and reflected signals will arrive at the receiver. If so it is possible to only consider the direct signals, using signal processing. However, if all signals arriving at the receiver are reflected there is no way to completely remove the error. The multipath effect is illustrated in Figure 2.2.

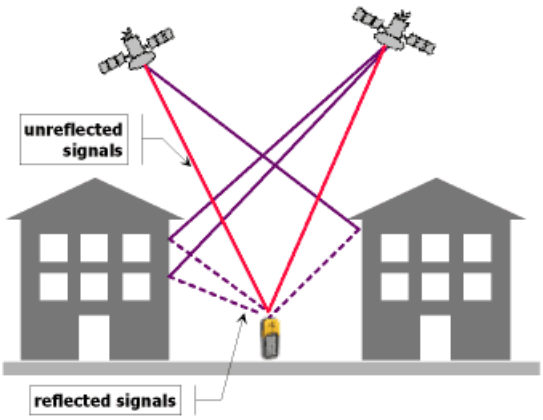


Figure 2.2: The multipath effect

2.5 Satellite Geometry

In addition to the different errors described above, there is yet another major factor determining the positional accuracy. That is the geometrical orientation of the satellites from which data is received. The general rule is that a greater angle between the satellites will lower the impact of the errors described above and vice versa. The quality of the satellite geometry is indicated by the Dilution Of Precision (DOP) value, which is included in the GPS signal. Many GPS receivers today have the ability to choose the satellites which will give optimal geometry. Measurements with too high DOP values can be ignored by the receiver itself or by GPS applications.

2.6 Differential GPS

Many of the errors discussed above can be diminished using a technique called Differential GPS (DGPS). This system relies on stationary GPS receivers, placed at known locations all around the world. These base stations continuously calculate their position based on the GPS signal from the satellites and compare it with their own physical location to calculate a correction factor. This information is then transmitted to DGPS enabled receivers in the region, which allow them to adjust their own calculations based on the error. The DGPS technology is commonly used in areas like construction industry, where very precise measuring is required.

2.7 Assisted GPS

Mobile phones with integrated GPS receivers are becoming increasingly common. These types of GPS receivers often have lower performing antennas than conventional receivers and strict demands on low energy consumption. This has led to the development of Assisted GPS (AGPS). This system differs from normal GPS by adding an *Assistance Server*, supporting the GPS receiver. Via a cellular network the GPS receiver communicates with the server, which supplies the receiver with precise GPS data such as satellite orbits and clock

information. The server can also be used to compute position solutions, since it has much more computing power. Delegating tasks to the server conserves battery power and speeds up computations, increasing the receiver performance which helps to compensate for the shortcomings.

2.8 Calculating the Distance Between GPS Coordinates

For navigation systems it is essential to be able to calculate the distance between GPS coordinates. A GPS coordinate is expressed in a geographical coordinate system, which is used to define any location on earth. If ignoring the elevation, it is defined by one latitude and one longitude component, see Figure 2.3.

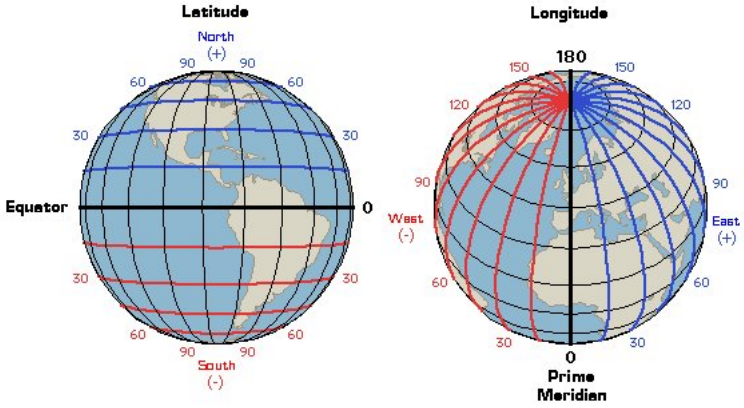


Figure 2.3: The latitude and longitude reference system

To calculate the distance between two positions, consider Figure 2.4, which illustrates an infinitesimally small distance dx from point A to B, with course C. The line from A to B is a so called *rhumb line*, which is a line on the surface of earth intersecting all meridians (longitudes) at a constant angle. If considered as a vector, it is composed of a north-south component, $dLat$, and a west-east component $dLon \cdot \cos Lat$. The $\cos Lat$ factor is the relative circumference of the respective parallel of latitude (equator = 1). From the figure, we see that:

$$dx = \frac{dLat}{\cos C}$$

The total distance D from A (Lat_A, Lon_a) to B (Lat_B, Lon_B) is found by integration:

$$D = \int_0^D dx = \int_{LatA}^{LatB} \frac{dLat}{\cos C} = \frac{Lat_B - Lat_A}{\cos C} \tag{2.1}$$

It turns out however, that the calculation of C is rather complicated. In practice this calculation is therefore mostly replaced by the *mid latitude* formula. This is an approximation which gives good results as long as the distance between both positions is not to large and both positions are far enough from the poles. The course C is calculated as:

$$C = \arctan\left(\cos Lat_M \cdot \frac{Lon_B - Lon_A}{Lat_B - Lat_A}\right) \tag{2.2}$$

$$Lat_M = \frac{Lat_A + Lat_B}{2}$$

Inserting equation 2.2 into equation 2.1 gives the distance D. Full coverage of this topic is found in [6].

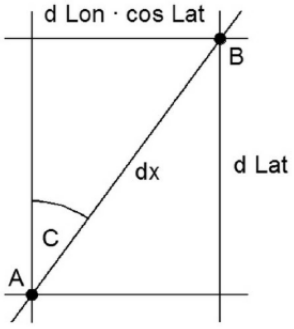


Figure 2.4: Rhumb line from point A to B

2.9 NMEA 0183

This sections describes the NMEA 0183 standard. Developed by the U.S. National Marine Electronics Association, it defines an electrical interface and data protocol for communications between marine instrumentation. It is commonly used by land based GPS receivers.

Data is transmitted in the form of sentences, built by ASCII characters. Each sentence fulfills the protocol rules:

- A sentence always starts with a dollar sign
- The next five characters define the message type
- Data fields that follow are comma-delimited
- The last data field is immediately followed by an asterisk character
- The asterisk is immediately followed by a two-digit checksum

The three types of NMEA messages used by Wayfinder Navigator are described below. A full reference to the NMEA 0183 standard is found in [7].

2.9.1 GGA - Global Positioning System Fix Data

The GGA message is a base message, supported by all GPS receivers. It contains time, position and fixation data, i.e. whether the receiver has contact with any satellites and the quality of the signal.

```
                                                    11
       1          2       3 4         5 6 7  8   9  10 |  12 13  14   15
       |          |       | |         | | |  |   |   | |   | |   |    |
$--GGA,hhmmss.ss,llll.ll,a,yyyyy.yy,a,x,xx,x.x,x.x,M,x.x,M,x.x,xxxx*hh
```

```
 1) Time (UTC)
 2) Latitude
 3) N or S (North or South)
 4) Longitude
 5) E or W (East or West)
 6) GPS Quality Indicator,
    0 - fix not available,
    1 - GPS fix,
    2 - Differential GPS fix
 7) Number of satellites in view, 00 - 12
 8) Horizontal Dilution of precision
 9) Antenna Altitude above/below mean-sea-level (geoid)
10) Units of antenna altitude, meters
11) Geoidal separation, the difference between the WGS-84 earth
    ellipsoid and mean-sea-level (geoid), "-" means mean-sea-level
    below ellipsoid
12) Units of geoidal separation, meters
13) Age of differential GPS data, time in seconds since last SC104
    type 1 or 9 update, null field when DGPS is not used
14) Differential reference station ID, 0000-1023
15) Checksum
```

2.9.2 GSA - GPS DOP and Active Satellites

The GSA message is a type of status message, describing which satellites are currently fixed to the GPS receiver. It also gives the DOP values which indicate the quality of the satellite geometry (see Section 2.5).

```
       1 2 3 4                       14 15  16  17 18
       | | | |                       |  |   |   |  |
$--GSA,a,a,x,x,x,x,x,x,x,x,x,x,x,x,x,x,x.x,x.x,x.x*hh
```

```
 1) Selection mode
 2) Mode
 3) ID of 1st satellite used for fix
 4) ID of 2nd satellite used for fix
 ...
```

```
14) ID of 12th satellite used for fix
15) PDOP in meters
16) HDOP in meters
17) VDOP in meters
18) Checksum
```

2.9.3 RMC - Recommended Minimum Navigation Information

The RMC message contains the recommended minimum data required for GPS navigation. The most important fields are the position, speed over ground and course over ground (bearing). These will all be used in the new off track detection algorithms.

```
                                                                  12
       1         2 3       4 5        6 7   8   9    10  11|
       |         | |       | |        | |   |   |    |   | |
$--RMC,hhmmss.ss,A,llll.ll,a,yyyyy.yy,a,x.x,x.x,xxxx,x.x,a*hh

 1) Time (UTC)
 2) Status, A = Data valid, V = Navigation receiver warning
 3) Latitude
 4) N or S
 5) Longitude
 6) E or W
 7) Speed over ground, knots
 8) Course over ground, degrees true
 9) Date, ddmmyy
10) Magnetic Variation, degrees
11) E or W
12) Checksum
```

2.9.4 GPS Coordinate Notation

The notation in which latitudes and longitudes of a GPS coordinate is given by the NMEA 0183 standard requires an explanation. For latitudes, the first two digits represent the integer part of the angle and the rest represent minutes of an angle. The following example shows how to convert a latitude L in the NMEA 0183 format to decimal format:

$$L = 5543.1127$$

This is converted as:

$$L' = 55 + \frac{43.1127}{60} \approx 55.7185$$

The format for longitudes is the same, except that the first three digits are required to represent the integer part as this takes values between 0 and 180.

Chapter 3

Wayfinder Navigator

To obtain a more theoretical model of the problem, it is necessary to have a better understanding of Wayfinder Navigator. This chapter will describe the parts of the system that are relevant for the scope of the thesis.

3.1 The Wayfinder Navigator System

As mentioned in the introduction, Wayfinder Navigator is an off-board navigation system, which means that memory and processing intensive tasks are handled by a central server. The Wayfinder server keeps updated digital maps, topological information describing how roads are connected and various other navigation data such as traffic information and points of interest. It is responsible for supplying the client with map and navigation data, which is continuously requested during navigation, and to carry out route calculations from start to destination. Communication between the server and the client is handled through a GPRS[1] or 3G connection.

GPS positions are fetched from a GPS receiver, which communicates with the mobile phone through a Bluetooth connection. Some new mobile phones, for example the Nokia N95, have built in GPS receivers, which makes the system more flexible and compact. Figure 3.1 gives an overview of the Wayfinder Navigator system.

3.2 Routing and Navigation

The key feature of Wayfinder Navigator is real time GPS navigation. To create a route for navigation, the user needs to make a *route request* specifying the start and destination of the route. Requests are handled by the Wayfinder server. Given a route by the server, the user can start to navigate. On the mobile phone screen, the route can be viewed as a red line on a digital map together with an arrow, marking the user's current position (see Figure 3.2). Real time turn instructions are presented to the user, both on the screen and via voice instructions. Navigation can be done using a 2D/3D map view or simply by having turn instructions displayed on the screen.

[1] General Packet Radio Service

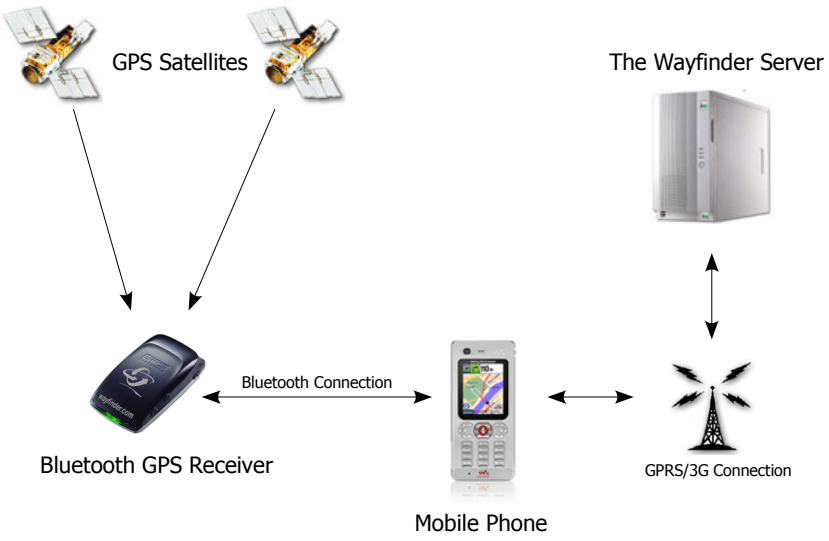


Figure 3.1: The Wayfinder Navigator system

3.3 Implementation

This section gives a simplified explanation of how the parts of Wayfinder Navigator which are relevant for the problem formulation are implemented.

3.3.1 Route Format

If successfully handling a route request, the server will respond with a binary data stream, containing all information about the route needed for the client to perform the navigation. The route is basically represented by a set of points with straight lines between them. All points contain a base set of attributes used during navigation. Some points contain additional information, for example where the driver should perform an action. All this data is accessed through an internal iterator.

For the scope of this project it is enough to know that each point has a position and that the route format is designed for forward iteration. This means that there is no natural way to step backwards in the route. A line between two points is called a *segment*. The angle of a segment is defined as the compass course of a vehicle driving along the segment.

3.3.2 The Navigation Engine

Having a route given by the server, navigation can be initiated. The different steps carried out during one iteration of navigation will be called *the navigation engine*, which are illustrated by Figure 3.3.

GPS positions are continuously fetched from the GPS receiver with a rate of one position per second. Given a GPS position, we want to know where on the route the driver is. This is done by selecting the segment of the route that best matches the position. In Wayfinder Navigator, two different algorithms for selecting segments have been implemented. The first

(a) 3D Navigation (b) 2D Navigation

Figure 3.2: Different navigation modes of Wayfinder Navigator

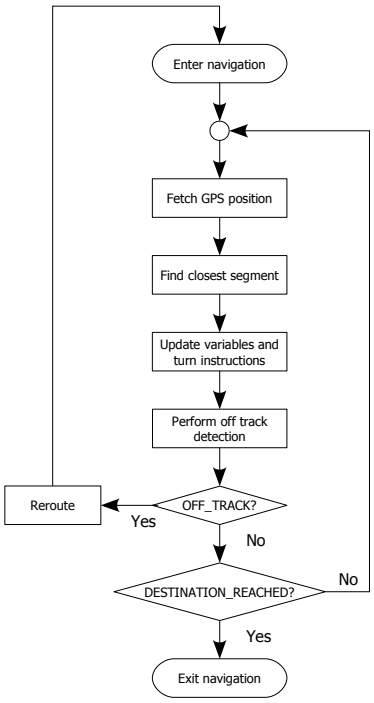


Figure 3.3: Steps of the navigation engine

solution is to simply pick the segment closest to the current position. For each segment candidate, the position is projected perpendicular onto the segment (see Figure 3.4). Sometimes positions cannot be projected perpendicular. In this case they will either be projected to the beginning or the end of a segment. From Figure 3.4 we see that positions P_3 and P_4 will both be projected on the same point $P'_{3,4}$. The distance between the projected and the actual position is the distance to the segment candidate. The closest segment is selected as the current

segment. A distance to a segment candidate must be strictly shorter than the current shortest distance in order for the segment to be selected as the new current segment. The projected position tells where the user is on the route and is used to generate turn instructions and display distances.

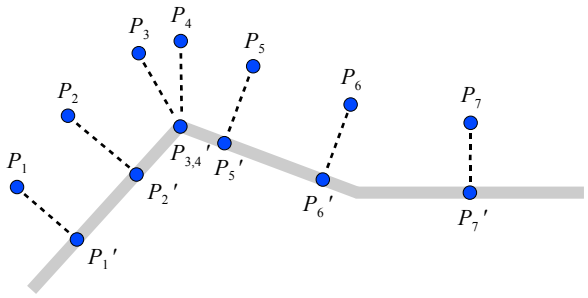


Figure 3.4: Positions P_1 - P_7 projected onto the route

The second segment selection algorithm is more sophisticated. It uses the same projection and distance calculations as the first, but in addition a penalty proportional to the angle difference between the segment and the current bearing is added to the distance if the error is big enough. The effect is that segments that do not match the bearing of the GPS signal, will be less likely picked as closest segment. A penalty is also added to positions that correspond to the beginning of a segment. The idea behind this extra penalty is to get better consistency between turn instructions and that this is in fact more important than knowing the exact position of the car. The off track detection algorithm is independent of the segment selection strategy. To avoid confusion, we will in this thesis always assume that the simple segment selection strategy is used.

After finding the closest segment, turn instructions and other status variables can be updated and off track detection can be performed. If the system finds the driver off track it will make a reroute, i.e. requesting a new route from the server. If the driver is on track a check is made to see if the destination is reached. This check will either cause the navigation to terminate, or start the next iteration by fetching a new GPS position.

3.4 Existing Off Track Detection Algorithm

The existing off track detection algorithm has the simple structure presented in Figure 3.5.

This algorithm is primarily designed to avoid false off track signaling by a single or a few erroneous positions, which explains the high number (six) of consecutive off track positions required to signal off track. 60 meters is chosen with concern to the accuracy of old GPS receivers as well as map errors. The problems caused by this design are discussed in the next chapter.

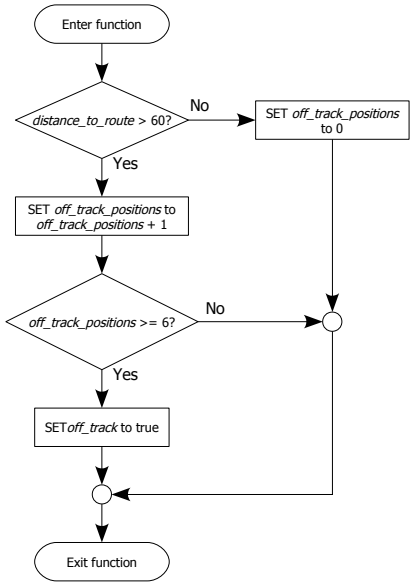


Figure 3.5: Existing off track detection algorithm

Chapter 4

Problem Discussion

Clearly, the existing off track detection algorithm does not work satisfactorily. In this section the problems associated with off track detection and the drawbacks of the current solution will be discussed.

4.1 Signal Disturbances

As described earlier several different sources of error disturb the accuracy of the GPS signal. Fortunately many of the effects of such errors can be minimized, but even with modern GPS receivers problems still exist. The general position error is characterized by a slow drift around the true position. This so called *wandering* is illustrated in Figure 4.1, where GPS positions have been collected during 24 hours. The average of all positions has been calculated and is used as a reference position, from which the distance to all other positions have been calculated. The experiment which was carried out (indoors) on a Sirf-Star III Bluetooth GPS receiver shows that 95.3% of the GPS positions remain within a distance of 15 m from the reference point.

The speed given by the GPS signal has a constant error added to it, which means that for low speeds the error will be relatively higher. The speed also affects the bearing, which remains undefined if the receiver is standing still or moving too slowly.

A perhaps more serious source of error is multipath, which is difficult to remove. A multipath error usually results in a sudden shift in position, which slowly converges to the correct position. The magnitude and direction of the error is dependent on how signals are reflected against different objects, which means it is hard both to predict and reproduce.

4.2 Map Errors

Another important source of error to consider is errors associated with the digital maps used for navigation. One problem is plain map errors, which usually result in roads or streets being shifted from their true positions with a fixed distance. Another problem is incorrect or faulty road digitization, i.e. errors in the digital approximation of streets and roads. A digital road is represented as a set of points, connected by straight lines. If this approximation is not made accurate enough, i.e. too few points are used, the differences between the real and approximated road may become significant. This problem can cause trouble if the digitized

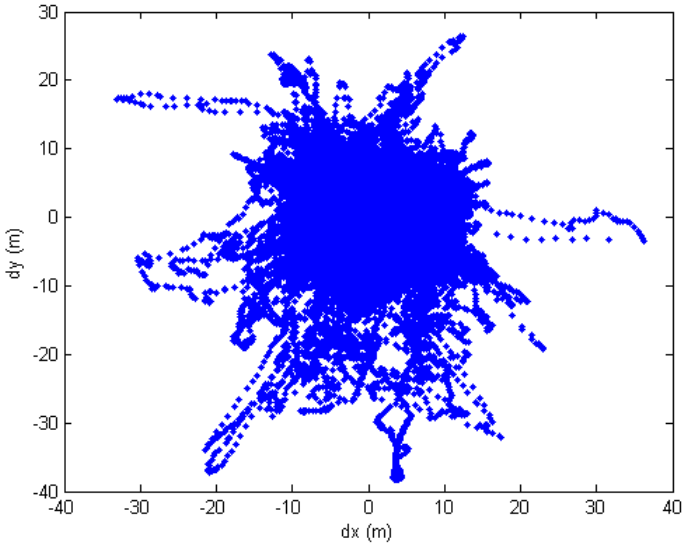


Figure 4.1: Experiment showing wandering GPS positions during 24 hours

road or street differs too much from the true location. Although unusual, cases also exist where low resolution digitization of freeways give errors of up to 200 meters. In general these types of map errors seem to be more frequent on locations with low traffic density.

On top of this comes the inaccuracy connected to the representation of the route. In order to save memory and minimize calculations, routes are often even harder filtered (represented by fewer points) than the roads. The additional errors introduced by this filtering are most present when short segments with big angular differences represent the road, for example at roundabouts and various types of crossings.

4.3 Problems with Existing Algorithm

The main problem with the existing algorithm is the long response time to off track situations. Since the algorithm relies solely on distance calculations, the off track threshold distance is set with a big margin to be able to avoid false alarms caused by signal disturbances and map errors. Since GPS signals are fetched with a rate of one sample per second and six consecutive off track positions are required, the off track detection time is typically 10-15 seconds. The problem becomes especially severe in situations where the driver goes off the route but still keeps in a close vicinity e.g. driving parallel to the route. Since the distance to the route is within the threshold, the system will not signal off track.

The second serious drawback is the risk of false alarms, meaning that the system might signal off track even though the driver is actually on the route. The current design with a big distance margin gives a fairly robust handling of disturbances and map errors but at the same time it clearly deteriorates the off track detection performance. Still it suffers from problems with multipath and roads with highly inaccurate digitization.

Finally, the existing system always displays the position derived from the GPS signal as

the driver's position. As described earlier, this approach may cause great confusion if the GPS gives a position which is not on the route.

Chapter 5

New Algorithms

The design of an off track detection algorithm is a compromise between speed and robustness. An algorithm with low thresholds will give a fast response, but may be sensitive to disturbances, whereas a very robust algorithm with high thresholds will sustain errors better but react slower on real off track situations. The goal is to design a solution which performs well in both of these aspects.

This project has resulted in two alternative off track detection algorithms. The first approach is to improve off track detection using existing GPS data. The benefits of such a solution is that it does not require a lot of implementation work, since it only makes use of the existing framework. The second approach relies on crossroads information for off track detection. This solution is believed to work better than the first one, but requires server cross road support, which does not exist at present. Analyzing the performance of the new algorithms will give a hint if such support is worth to implement. This chapter describes the fundamental steps carried out during the development of the new off track detection algorithms, as well as the design of the new solutions.

5.1 Simulation Environment

To be able start the design and development of new algorithms it is necessary to establish a navigation simulation environment. Such an environment has been created by a simplified Matlab implementation of the navigation engine, described in Section 3.3.2. The simulation routine is fed with the following inputs:

- **GPS data** which contains GPS positions with coordinates, bearing and speed (1 sample/s)
- **Route data** which defines the segments of the route
- **Cross road data** which defines the segments of cross roads and how they are interconnected

Upon execution, the routine steps through each time step, calculating the distance and angle error to the route and to cross roads. For the distance calculations, the mid latitude formula explained in Section 2.8 is used. The distance and the angle error outputs are then used to perform the off track detection.

5.2 Test Cases

To test and analyze the performance of different algorithms, GPS data have been logged while driving a car with a GPS receiver. Parts of this data have been selected and matched with short routes to form a number of test cases which the off track detection algorithm must be able to handle. In some cases the data has been modified to simulate different types of errors. The test cases can be divided into two main categories:

- **Off track test cases** where the user drives away from the route, after some distance.
- **On track test cases** where the user stays on the route.

5.2.1 Off Track Test Cases

The off track test cases are used to test the response performance of an algorithm, i.e. how fast it reacts to off track situations. Examples of off track test cases are:

- Driving off the route by a 90 degree turn
- Driving off the route by a 45 degree turn
- Driving off the route via exit on freeway
- Missing a turn and continuing straight forward

5.2.2 On Track Test Cases

The on track tests are used to check the robustness of the algorithms to map and GPS errors. Examples of on track test cases are:

- Turning through a crossing
- Driving through a roundabout
- Driving on a road with map errors
- Driving with GPS signal disturbances

5.3 J2ME Implementation

Implementation on mobile phones on the J2ME requires special consideration. First of all, since J2ME is designed for resource-constrained devices it does not support all API:s of ordinary Java. This includes for example various trigonometric operations, which will have to be implemented by the developer. Other important issues to keep in mind is the limited memory and processing capacity as well as the battery consumption aspect. Minimizing the use of floating point arithmetic and large data structures is therefore important. These considerations have been taken into account in both the design and implementation steps of this project.

5.4 Testing

In addition to simulation, the implementation has also been tested. This can be done either by simply driving with the application or by using a GPS simulator. Such a simulator can be used to load previously logged GPS data, which enables realistic testing of the application.

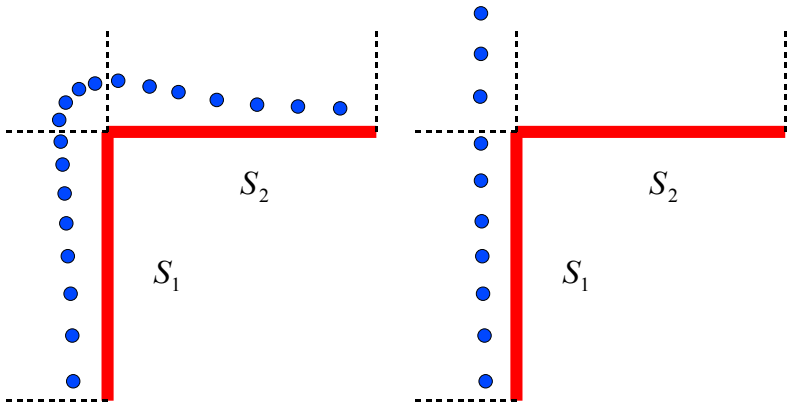
5.5 Off Track Detection Using Existing Data

Observing more indicators than just the distance to the route, it is possible to improve performance. This section describes an algorithm which makes better use of existing data and motivates its structure.

5.5.1 Angle Error as Off Track Indicator

One potential indicator that can be used to improve off track detection is the angle error, i.e. the angular difference between the current segment and the driver's bearing. A significant angle error would probably mean that the user is heading away from the route.

Unfortunately this indicator suffers from some quite serious problems. As described earlier, a GPS position is always matched to a single segment of the route. Crossings and small roundabouts are usually digitized using few segments, giving the digital route sharp turns with big angular differences between consecutive segments. Matching the GPS position to the wrong segment could give a big angle error, even though the user is on track. In a similar way, a user heading away from the route can have a small angle error. Examples of the two situations are illustrated below.



(a) Follow route through sharp turn (b) Missing turn

Figure 5.1: Segment matching problems

Figure 5.1(a) shows a situation where the driver follows the route through a sharp turn. Map errors and/or disturbances cause the GPS position to be slightly offset from the route. The dashed lines mark the matching regions of segment S_1 and S_2, i.e. the regions from which positions will be projected onto the different segments. Positions ending up between

the two matching regions will have the same distance to both segments. Remembering that the distance needs to be strictly lower than the previous shortest distance for a new segment to be set as current segment, we see that segment S_2 will not be set as the current segment until positions enter its matching region. This will cause large angle errors for positions between the matching regions, as these are matched to segment S_1.

Figure 5.1(b) shows a situation where the driver instead of following the route, misses the turn instruction and continues straight forward. Since GPS positions never enter the matching region of segment S_2, all positions will be matched to S_1 and hence have a very small angle error, even though the driver is clearly heading off track.

These examples show the difficulties of using the angle error as an off track indicator. Large angle errors must be tolerated to avoid false off track signaling, which increases the reaction time. Using the extended segment selection strategy described in Section 3.3.2 would do little to solve these problems. Since this algorithm adds penalties to positions projected to the beginning of segments, this would only lead to more positions being matched to segment S_1. One way of overcoming the problems in the second situation would be to skip the strictly smaller distance requirement and accept S_2 as the current segment when the distance from the current position is the same as to S_1. Using both the distance and angle error for segment selection would be useful in the first situation.

5.5.2 Normalized Distance To Route Derivative Indicator

In order to overcome the problems described in the previous section, a new indicator has been designed. To explain this concept, consider a GPS position P and its perpendicular projection onto the route P'. Set $d_\perp$ as the distance between P and the route and $d_\parallel$ as the distance traveled along the route from the start to the projected point P'. Define the change of these values between two consecutive positions as $\Delta d_\perp$ and $\Delta d_\parallel$, see Figure 5.2.

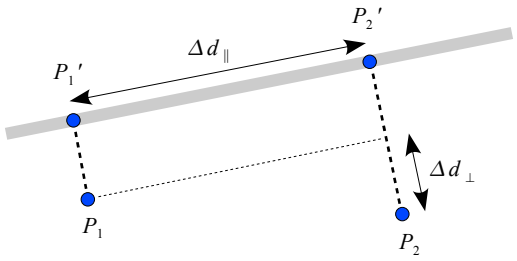


Figure 5.2: Illustration of $\Delta d_\perp$ and $\Delta d_\parallel$ for two consecutive GPS positions P_1 and P_2

Finally define the distance to route derivative, normalized with the derivative of the distance traveled along the route:

$$D(t) = \frac{\Delta d_\perp}{\Delta d_\parallel + 1\,\mathrm{m/s}} = \frac{d_\perp(t) - d_\perp(t-1)}{d_\parallel(t) - d_\parallel(t-1) + 1\,\mathrm{m/s}}$$

The denominator is increased by one to avoid division by zero. This value is independent of segment angles and serves as a good off track indicator. In theory, any nonzero positive value would indicate that the user is heading off track.

5.5.3 Complements

Even though the value proposed in the previous section is a good off track indicator it is not enough to handle all situations. As shown in Section 4.1, significant uncertainty is present in the GPS signal when the receiver is standing still. A drifting position may cause the indicator to signal off track, which makes it necessary to require a minimum speed of the vehicle to perform off track detection.

We also need to handle the case of a user driving the route in the wrong direction. Due to the route format, which is designed primarily for forward iteration, there is no practical way to move backwards in the route. The angle error will therefore be used to check if the driver is heading in the wrong direction.

5.5.4 Algorithm Structure

Based on the indicators presented in the previous sections, the first off track algorithm has been designed. To avoid the uncertainty of the position, speed and bearing previously discussed, the receiver must reach a minimal speed limit. This is tested at the very beginning of each iteration. Only if this test succeeds, we continue with the off track detection.

At first, a test is made to see if there exist strong off track indicators. This means either one of the following:

- A high value of the normalized distance to route derivative D, indicating that the driver is heading away from the route with a large divergence.
- A very large angle error, meaning that the vehicle is driving along the route in the wrong direction.
- The vehicle is very far from the route. Let this limit be defined by `max_distance_to_route`.

The idea is to primarily use the D value to check if the driver is diverging from the route. Positions far from the route can be accepted as long as the D value is low, in order to avoid false alarms due to shifted or faulty digitized roads. `max_distance_to_route` is set as an upper limit for the size of these errors. It also works as a backup if off track has not been detected earlier.

The most difficult situation to detect is vehicles leaving the route with a small divergence. To make such detections a lower threshold for D has to be used. A lower value may cause false alarms, particularly in crossings with map errors. To overcome these problems a second distance limit has been defined: `short_distance_to_route`. This limit is set to cover combined map and GPS errors. It is thus possible to use a lower limit for D if the distance to the route is above this limit. A check to see if these conditions are fulfilled is done if no strong off track indicators are found.

To be able to reach the goal of 2-3 seconds off track detection time stated in the goal formulation, the number of consecutive off track positions has been set to three. This gives a theoretical minimal detection time of 2 seconds.

5.6 Off Track Detection Using Crossroad Data

The second approach is to use information about cross roads for off track detection. This method relies on the assumption that a driver will always have to exit the route via a crossroad. Knowing the positions of crossroads will make it possible to perform a fast and robust off track detection. The crossroad format used during this project defines crossroads in the same way as routes, i.e. by points connected by straight line segments.

5.6.1 Penalty Function

To determine if the driver is on the route or in fact driving on a crossroad, the concept of penalty functions has been developed. A penalty function assigns a penalty to a position to find out how bad this position matches a route or a crossroad segment. Each position P is given a distance penalty $Q_d(P)$ for the distance between the position and the route/crossroad, and an angle penalty $Q_a(P)$ for the angle difference between the route/crossroad and the position's bearing. These are defined by:

$$\begin{aligned} Q_d(P) &= 10 \cdot \min(1, f_1(d(P))) \\ Q_a(P) &= 10 \cdot \min(1, f_2(a(P))) \end{aligned}$$

where $d(P)$ is the distance to the route/crossroad and $a(P)$ the angle error. f_1 and f_2 scale the values to achieve balance between the penalties, i.e. they should have the same impact. Each penalty is limited to a maximum of 10 points in order to ignore extreme values of $d(P)$ and $a(P)$. The total penalty Q for a position P is given by the sum of the angle and distance to route penalties:

$$Q(P) = Q_d(P) + Q_a(P)$$

We see that $Q(P) \in [0, 20]$.

5.6.2 Algorithm Structure

Using penalty functions, we can determine how well a GPS position is matched both to the route and crossroads in the vicinity. If the crossroad with best (lowest) penalty matches the current position better than the route, i.e. it has a lower penalty, this may signal that the driver is on the cross road instead of the route. We can thus use this penalty difference as an off track indicator.

When applying the penalty function to crossroads, only those in a close vicinity to the current position should be considered to minimize the computational effort. In the Matlab implementation crossroads starting from the current, previous and next segment are compared to the route. Crossroads too far from the current position will be ignored.

Comparison of the route with the best cross road includes both the relative and the absolute penalty function difference. This is needed to maintain robustness in the algorithm. Map errors and signal disturbances make it likely that GPS positions sometimes are better matched by crossroads than the route. A common example is driving on freeways with GPS positions shifted perpendicular to the road. This may cause freeway exits to match the GPS positions better, leading to false alarms.

As with the previous algorithm a minimum vehicle speed is required and the number of consecutive off track positions required for off track has been set to three in order to fulfill the goal formulation.

5.6.3 Implementing Crossroad Support

The crossroad data representation used in this project is designed for drawing crossroads in the navigation view and thus far from optimal when it comes to off track detection. If implementing crossroad support as a part of the route format, various issues need to be considered.

First of all the representation of crossroads needs to be as compact as possible to minimize the size of the route. A minimal representation would be to just represent the points on the route from which crossroads start (and perhaps on which side of the road). This information could be used to ignore extreme positions far from the route if we know that there are no crossroads in the vicinity, typically on freeways or main country roads. However if positions diverge from the route and we have crossroads in the vicinity, we would not know if positions originate from a disturbance/map error or the fact that the vehicle is on a crossroad. To achieve the full potential of crossroads they will have to be represented by one or several segments.

Using this representation, the next thing to consider is the length of crossroads. They need to be long enough to give the off track algorithm time to detect an off track situation. Clearly, the length required will depend on the orientation of the crossroad in relation to the route. Crossroads running perpendicular away from the route can be much shorter than crossroads running almost parallel. This can be solved by representing crossroads to a limit where they exceed a fixed distance to the route. A maximum crossroad length can be set to cut off crossroads running parallel to the route for too long.

Finally comes the question how to access the crossroad data in the route format. It seems natural to store the representation of crossroads in the point from which they originate. Looking only at the crossroads starting from the current segment will not be enough since the current route segment can be changed while driving on a crossroad and we would then loose the crossroad reference. The algorithm needs to look at crossroads originating from both previously passed and future segments. A distance threshold can be used to ignore crossroads too far away. References can be kept to crossroads starting from segments already passed and gradually dropped when they exceed the distance limit. In a corresponding way, the algorithm can step forward from the current segment and access crossroads within the distance limit. By tuning the distance limit, we can reduce the number of potential crossroads and thereby make the algorithm more efficient.

Chapter 6

Simulations

This chapter presents the results of some of the simulations carried out during this project. GPS data for the simulations has been gathered using a Sirf-Star III Bluetooth GPS. 18 test cases (9 off track and 9 on track) have been selected. These are denoted Test Case A-R and are presented in Figures A.1 to A.18. When referring to the newly developed algorithms, these will be called Algorithm A (uses existing data) and Algorithm B (uses crossroads). The normalized distance derivative will be denoted D. The presentation of each test case includes the following items:

(a) **Map** - A graphical overview of how the car is driving. The bold line represents the route and the thinner lines show different crossroads. GPS positions are displayed as dots, where every 10th dot (10th second) is numbered. Positions where the different algorithms detect off track are marked by P_E (existing algorithm), P_A (Algorithm A) and P_B (Algorithm B). Important to note in these plots is that GPS positions are given by latitude and longitude coordinates, which means that distances and angles can be different from what they appear in the plot (see Section 2.8).

(b) **Response Times/False Off Track** - The algorithms' response times for each off track test case and if false off track is signaled or not for each of the on track test cases. The response time is defined as the time difference between the point where the driver leaves the route and the point where the algorithm signals off track. Defining the point where the driver has left the route is somewhat subjective. The definition used during the simulations is the first point where the car has entered and is aligned with a crossroad. This may give very short response times if off track positions start to be registered as the car turns away from the route.

(c) **Distance to Route** - The shortest distance in meters between the car's GPS position and the route.

(d) **Angle Error** - The difference between the bearing of the GPS position and the current segment angle in degrees.

(e) **Speed** - The car's speed in meters per second.

(f) **Normalized Distance Derivative** - The normalized distance to route derivative indicator presented in Section 5.5.2.

(g) **Penalty Function** - Value of penalty function for the route and best crossroad.

(h) **Off Track - Existing Algorithm/Algorithm A/Algorithm B** - The off track signal for each algorithm, giving 1 (true) if the position is an off track position and 0 (false) if not.

6.1 Comments

Comments on the results of some particularly interesting test cases are covered by this section.

6.1.1 Test Case A

This test case has been created to illustrate the problem with segment matching when missing a turn, discussed in Section 3.3.2. We see that the angle error remains small until positions start to be matched to the second segment at $t = 25$. The D value rises immediately after the car has passed the crossing. This is used by Algorithm A to make a fast detection. The angle penalty for the route will remain low until $t = 25$ so the total route penalty will grow only due to the increased distance penalty. This slows down the detection speed of Algorithm B.

6.1.2 Test Case B

This test case illustrates a typical off track situation where the driver goes off track by a 90 degree turn. Both algorithms perform well and detect off track almost immediately. For both Algorithm A and B, off track positions start to be registered already during the turn. Since the off track point, i.e. the point where the driver has left the route is defined as the first point after the turn is completed, the off track detection times of Algorithm A and B are less than 2 seconds, which normally is the theoretical minimum detection time. Because of the low speed of the vehicle it takes a long time to reach a distance of 60 meters from the route. The increase in performance compared to the existing algorithm is therefore very large.

6.1.3 Test Case C

In this test case, the driver exits a traffic route via an exit. This off track situation is more difficult to detect than the previous, since the driver diverges more slowly from the route. This means that the D value and the penalty difference between the route and the crossroad also grow slowly. Still both new algorithms perform significantly better than the existing one.

6.1.4 Test Case D

This test case is somewhat similar to the previous. Again, the driver exits the route with a small divergence, but here the speed is lower than in the previous example. The lower speed mainly affects the result of Algorithm A. The D value is too low to reach the strong off track indicator level, which means the GPS position must reach the lower distance from route limit before off track positions can start to be registered due to D. Since the car is moving slowly, this will take time and the performance of Algorithm B will be much better. This situation is an excellent example of when use of crossroad data can boost performance.

6.1.5 Test Case E

A difficult situation for an off track indicator is a driver exiting a freeway. This test case covers an example where the driver exits a freeway, enters a roundabout and turns left. We see that the existing off track algorithm takes very long to detect off track, since the car stays close to the route long after it exits the freeway. The D value stays low when the car is on the exit ramp since this runs almost parallel to the route. When the driver enters the roundabout, the D value increases and off track is detected by Algorithm A. In this example the crossroad is too short to let Algorithm B detect off track. From the penalty plot we see that the crossroad penalty is lower than the route penalty for a while but the difference is not enough to overcome the threshold required for off track. This illustrates the risk of having too short crossroads.

6.1.6 Test Case J

In Section 5.6.2 the problem of having perpendicular shifted GPS positions when driving on freeways was mentioned. This test case has been created to simulate these types of situations. The GPS positions have intentionally been moved out 15-20 m perpendicular to the road to match a freeway exit. The probability that a similar situation would happen somewhere on a long route along a freeway is relatively high and it is probably one of the most difficult types of situations for a crossroad off track detection algorithm when it comes to avoiding false alarms. We see from the penalty plot that the penalty is significantly lower for the crossroad. This test case therefore serves as an important benchmark for tuning parameters of Algorithm B.

6.1.7 Test Case K

The purpose of this test case is to simulate a very inaccurate digitization of a freeway. The curved road has been approximated with a straight line, which gives a distance inaccuracy of up to 180 meters. This can be considered as an extreme case of faulty road digitization. The existing algorithm gives a false alarm due to the large distance from the route. Algorithm A will not signal off track since the D value remains low during the entire period. The test case is also easily handled by Algorithm B since no crossroads exist in the vicinity, which means that the driver cannot be off track, regardless of the route penalty.

6.1.8 Test Case L

GPS errors caused by multipath (discussed in Section 2.4.4 and 4.1) can be difficult to handle for an off track algorithm. From the collected GPS data, there is only one good example of multipath effects. This test case presents the example. Looking at the map and distance to route plots, one can see a sudden shift in the GPS position at $t = 37$. The error remains large for a while and then slowly diminishes. Sudden shifts may be difficult for Algorithm A to handle but since the position is shifted not only perpendicular to the route but also along the route, the D value stays low and Algorithm A handles the disturbances well. From the penalty plot we see that no crossroads are close enough to be considered when the disturbances occur. Algorithm B will thus have no problems to sustain the shifted positions.

6.1.9 Test Case M

In this case the issue of wandering GPS positions, previously discussed in Section 4.1 is addressed. The map plot shows a car stopping at a red light at $t = 15$. While standing still, the GPS position starts to drift perpendicularly away from the route. After a while the car starts to drive but soon stops a second time. Again, the position starts to drift in the same direction. Finally the car continues and makes a turn. The GPS position converges to the true position. A wandering GPS position may cause trouble for both of the new algorithms. They may both give a high value of D and match a crossroad. This test case clearly motivates the need for having a minimal speed limit requirement to perform off track detection.

6.1.10 Test Case N

This test case illustrates the second problem associated with segment matching, discussed in Section 3.3.2. The car stops at a traffic light at $t = 15$. The GPS position drifts away and the position remains shifted throughout the turn. From the angle error plot, we see that large angle errors will be given during the turn and it would therefore be risky to use this as an off track indicator. In contrast the D value stays low during the turn and by using this value we avoid the risk of false alarms.

6.2 Summary

	Existing Algorithm	Algorithm A	Algorithm B
Test Case A	11 s	3 s	5 s
Test Case B	20 s	1 s	0 s
Test Case C	14 s	5 s	5 s
Test Case D	32 s	11 s	3 s
Test Case E	53 s	31 s	-
Test Case F	12 s	2 s	3 s
Test Case G	9 s	3 s	2 s
Test Case H	65 s	58 s	-
Test Case I	20 s	15 s	13 s

Table 6.1: Response times

Both of the new algorithms have been developed with a focus primarily on avoiding false alarms, which is covered by the on track test cases. The performance of the algorithms is therefore best analyzed by looking at the off track detection speed. An overview of the response times from the off track test cases are given by Table 6.1. From these results it is clear that the response times can be significantly reduced by using either of the new algorithms. Adding that robustness is maintained we see that both Algorithm A and B will give large performance improvements compared to the existing algorithm.

By comparing the response times of Algorithm A and B wee see that in many of the test cases both algorithms perform similarly. These are test cases where the driver quickly

diverges from the route, often through a sharp turn. Probably, these cases also represent the majority of the off track situations occurring during real navigation.

Another common situation is off track through a freeway exit. From test case E we have seen that the crossroad representing the exit must be longer in order for Algorithm B to detect off track. The same holds for test case H. Test case I shows that Algorithm B can give a performance improvement for these types of situations if the exit is represented well enough.

The best performance improvement by Algorithm B compared to A is given in situations where the driver diverges slowly and with a relatively small angle from the route. This is illustrated by test case D, where the response time for Algorithm B is almost four times faster than for Algorithm A. As we have seen, crossroads are also very useful when it comes to maintaining robustness, since extreme positions can be ignored if no crossroads are present. Test cases K and L are good examples of such situations.

To summarize, we see that Algorithm B generally performs better than Algorithm A even though A may perform better in some situations. However, it should be pointed out that Algorithm B has been designed robust enough to handle situations like the one in test case J, which clearly has an effect on the off track detection time. As mentioned earlier, only one example of multipath exists in the test cases and it is possible that the robustness of Algorithm A needs to be increased to sustain larger disturbances. Hence, the performance difference between the algorithms may be increased.

Chapter 7

Conclusions and Future Work

7.1 Conclusions

The goal of this Master's Thesis has been to improve the car navigation experience for a Wayfinder Navigator user by improving the off track detection and the presentation of the car's position to the user. In accordance with the project plan, two new off track algorithms have been developed (one using only GPS data and one using crossroads) and simulations have been carried out to test their performance. The goals set up for the project included avoidance of false off track signaling and an off track detection time of 2-3 seconds. Simulations show that false off track signaling is avoided in all test cases. The off track detection time goal can be reached in situations where the driver quickly leaves the route, but if the driver diverges slowly, the position and map inaccuracies make it hard to make a fast off track detection if robustness is to be maintained.

In general, the off track detection performance has been improved significantly compared to the previous solution. The main reason for this improvement is better knowledge about the GPS signal behavior, including errors. This knowledge has been established through logging and analyze of GPS data from realistic environments. With better knowledge of which GPS behavior can be expected during navigation, it has been possible to develop more sophisticated algorithms. We have also seen that Algorithm B, which uses additional knowledge about crossroads generally performs better than Algorithm A. The connection between additional knowledge and improved off track performance is thus strong.

As with any type of testing, there is no guarantee that an algorithm will work during all possible circumstances, even though it passes a set of test cases. The result will be highly dependent on how well the test cases can cover different types of navigation situations, including disturbances. The GPS data collected to create the test cases have been gathered while driving in a limited area and with one type of GPS receiver, which makes it difficult to capture all kinds of behavior. Some disturbances can be created in simulations while others, such as multipath and the effect of map errors are hard to reproduce. To get a better characterization of these types of errors, more data will have to be gathered from areas where the errors are expected to be present.

Conclusions have also been drawn from the development work. As a recommendation for similar types of projects, it is probably a good idea to gather all measurement data at the very beginning of the project and make a thorough review. Perhaps statistical methods

can be applied to build up an error model for use throughout the project. In this project, some data was gathered at the very beginning and used to create a first solution. Later in the project, more data was introduced, new discoveries were made and the initial approach had to be altered. This scenario could have been avoided if all data would have been available from the very beginning. In a similar way, different suggestions and opinions on how the problem can be solved should be gathered at the project startup.

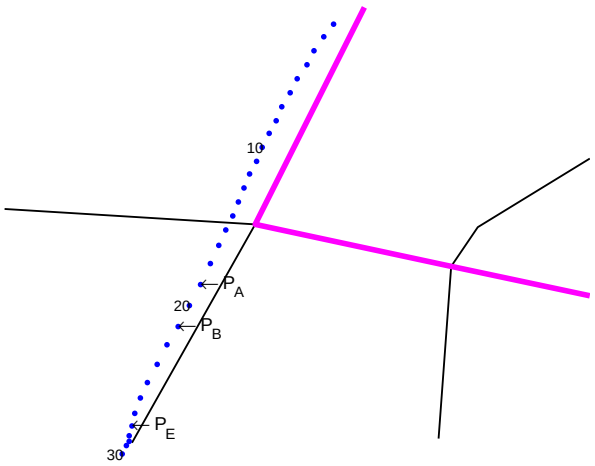
7.2 Future Work

For the most adjacent future, further testing will follow to investigate the behavior of the new implementation, which includes Algorithm A together with snap to route. In a longer perspective, more issues regarding the off track detection could be interesting to examine. Some examples are presented below:

- For Algorithm A, an interesting extension would be to add prediction of future positions. Doing so would make it easier to handle for example the multipath effect by ignoring positions which diverge too much from the prediction.
- For Algorithm B, it could be worthwhile to develop the penalty functions further. The current penalty function is linear, but perhaps it would be wiser to have a different statistical distribution, for example by making the impact of relative error changes stronger. Better error characterization could help to accomplish this.
- The number of consecutive off track positions required for off track could be altered depending on the margin to the off track threshold. If the threshold was reached with a very fine margin, the number of required positions could set be higher to reflect the uncertainty and avoid false off track. The opposite could be applied if the driver is heading away from the route with high speed and off track indicators are far beyond the required limits.
- The quality of the map data is an important error source and therefore important for the off track detection performance. The map quality varies between different regions and by safeguarding against severe map errors in a region with high map quality, performance will be lost. The optimal solution would be to alter the off track detection sensitivity, depending on the map quality of the current area.

Appendix A

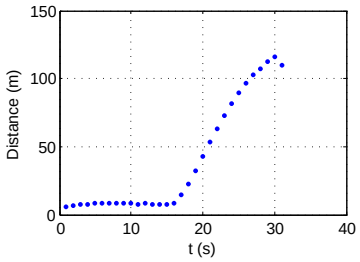
Simulation Results



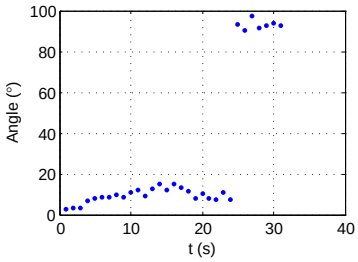
(a) Map

Existing Algorithm	11 s
Algorithm A	3 s
Algorithm B	5 s

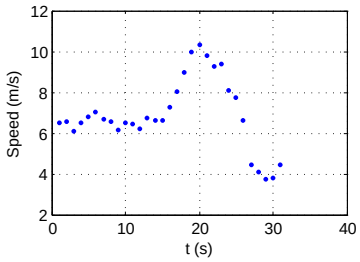
(b) Response Times



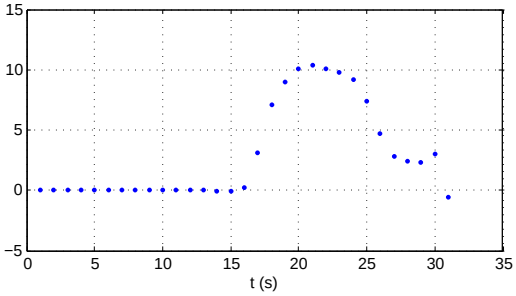
(c) Distance to Route



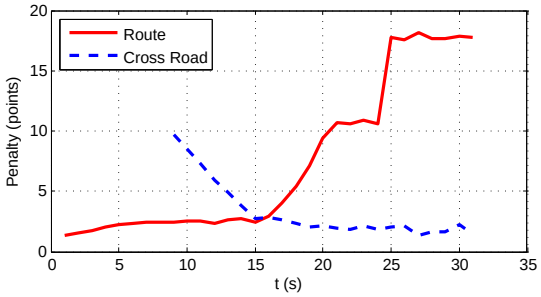
(d) Angle Error



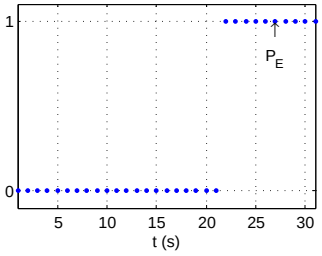
(e) Speed



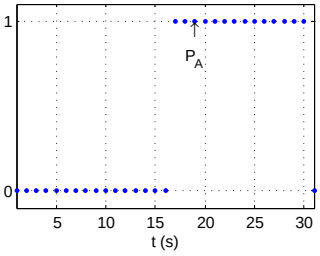
(f) Normalized Distance Derivative



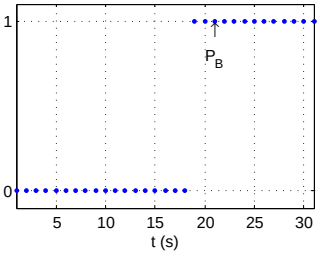
(g) Penalty Function



(h) Off Track - Existing Algorithm

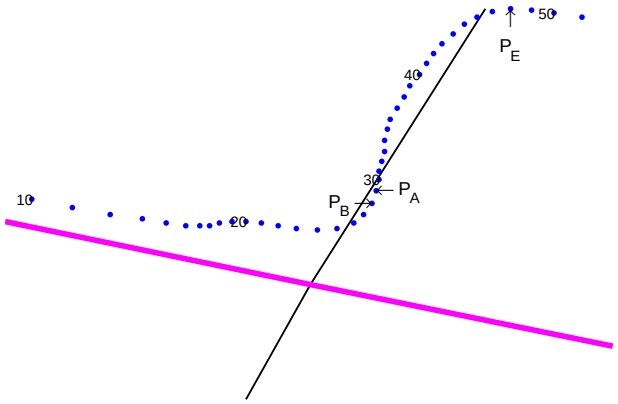


(i) Off Track - Algorithm A



(j) Off Track - Algorithm B

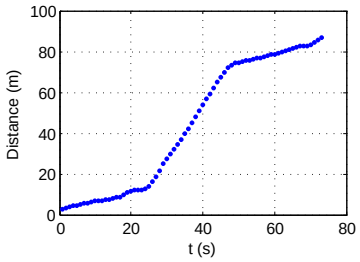
Figure A.1: Test Case A



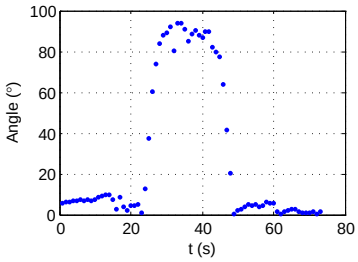
(a) Map

Existing Algorithm	20 s
Algorithm A	1 s
Algorithm B	0 s

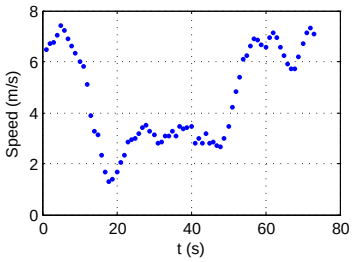
(b) Response Times



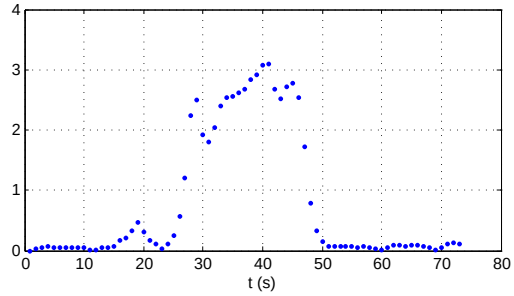
(c) Distance to Route



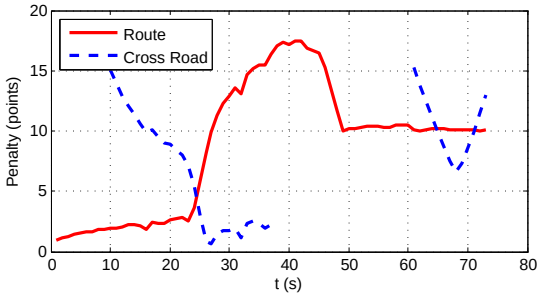
(d) Angle Error



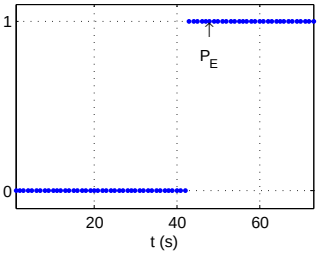
(e) Speed



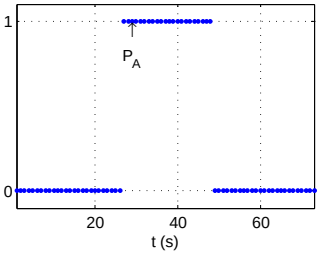
(f) Normalized Distance Derivative



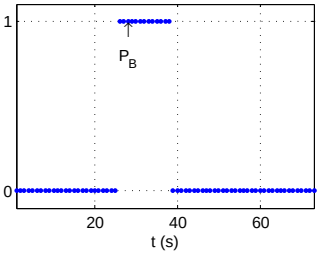
(g) Penalty Function



(h) Off Track - Existing Algorithm

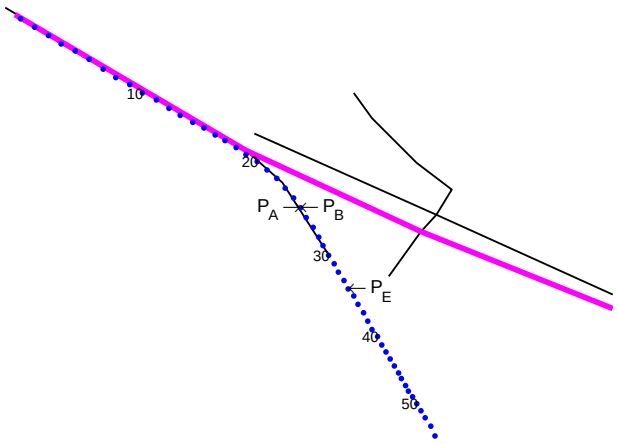


(i) Off Track - Algorithm A



(j) Off Track - Algorithm B

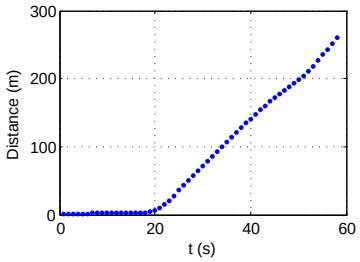
Figure A.2: Test Case B



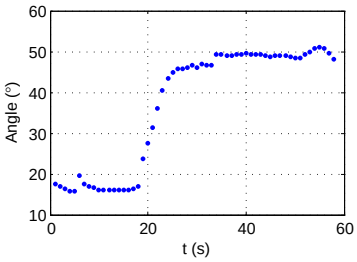
(a) Map

Existing Algorithm	14 s
Algorithm A	5 s
Algorithm B	5 s

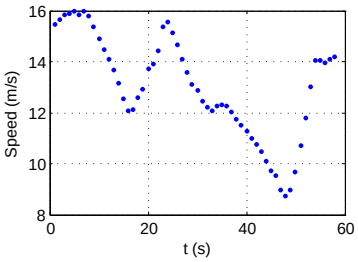
(b) Response Times



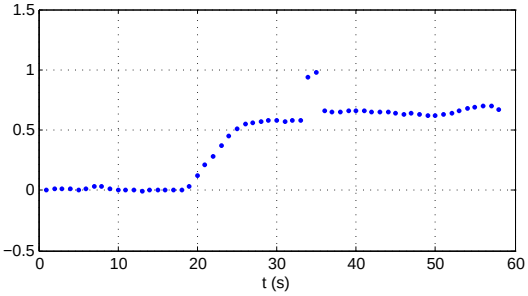
(c) Distance to Route



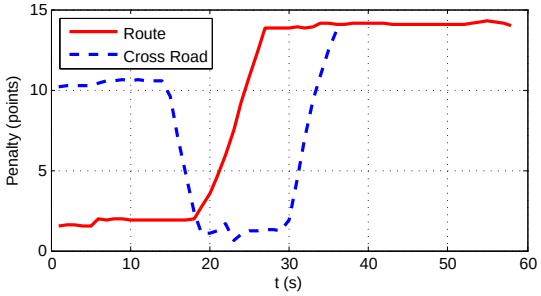
(d) Angle Error



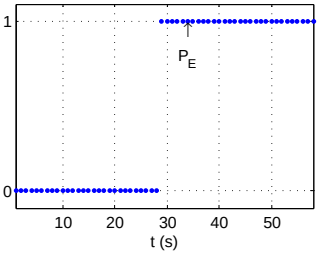
(e) Speed



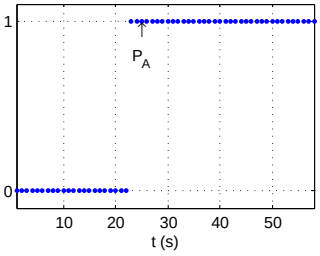
(f) Normalized Distance Derivative



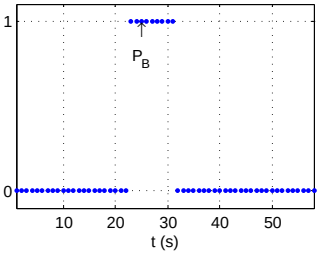
(g) Penalty Function



(h) Off Track - Existing Algorithm

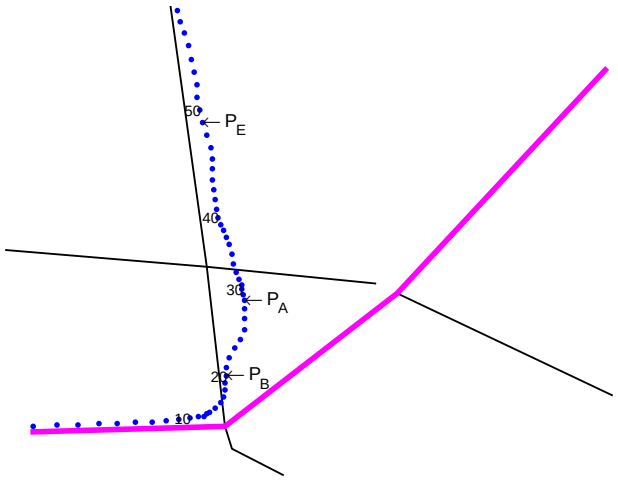


(i) Off Track - Algorithm A



(j) Off Track - Algorithm B

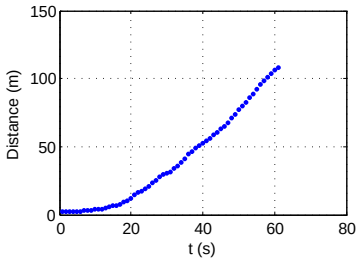
Figure A.3: Test Case C



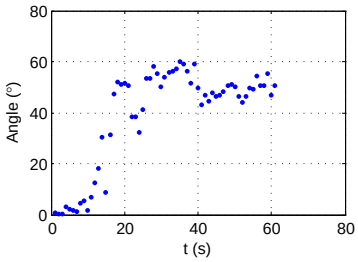
(a) Map

Existing Algorithm	32 s
Algorithm A	11 s
Algorithm B	3 s

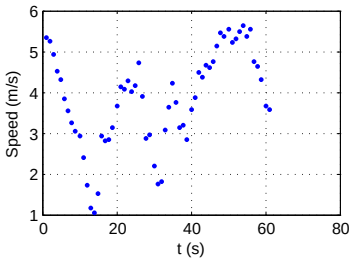
(b) Response Times



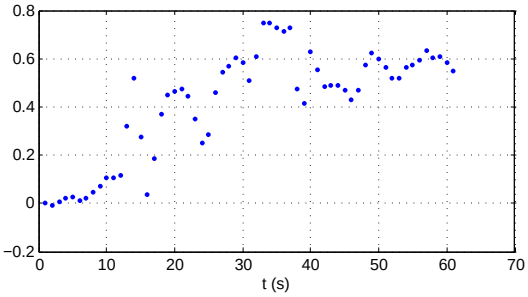
(c) Distance to Route



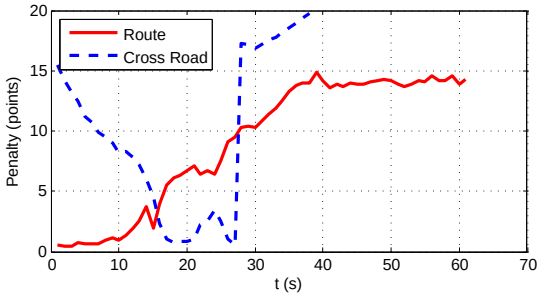
(d) Angle Error



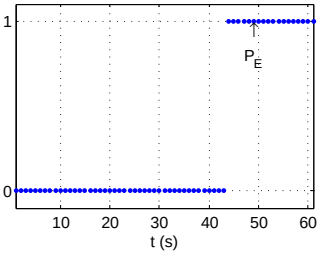
(e) Speed



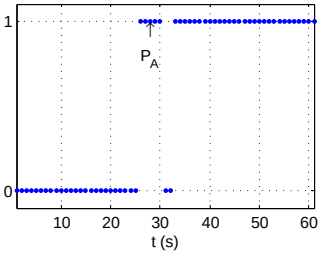
(f) Normalized Distance Derivative



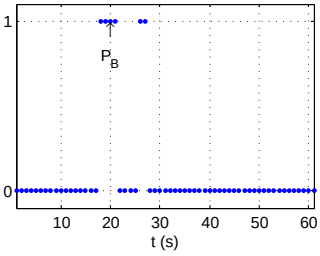
(g) Penalty Function



(h) Off Track - Existing Algorithm

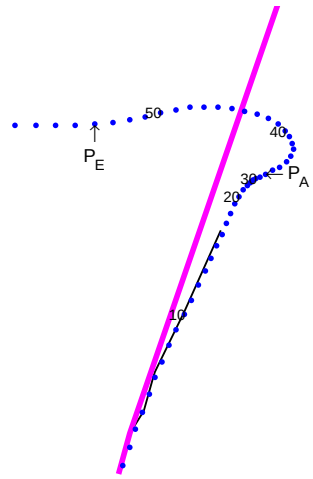


(i) Off Track - Algorithm A



(j) Off Track - Algorithm B

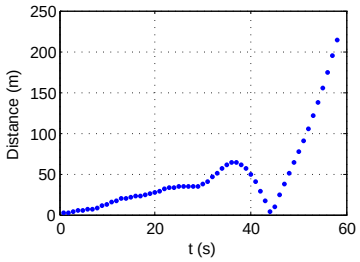
Figure A.4: Test Case D



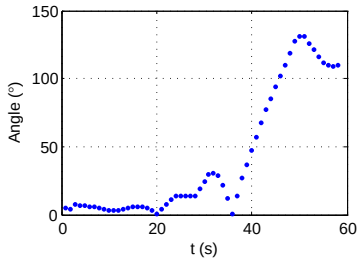
(a) Map

Existing Algorithm	53 s
Algorithm A	31 s
Algorithm B	-

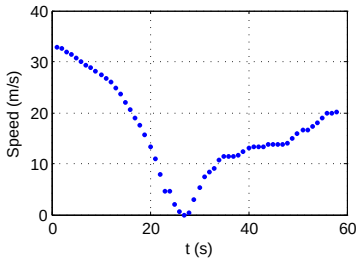
(b) Response Times



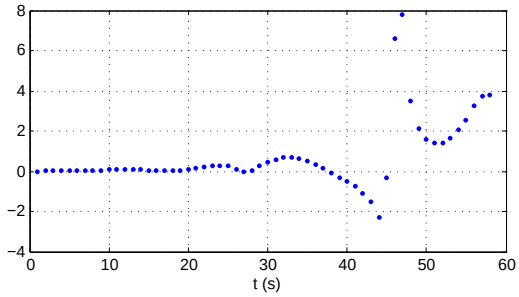
(c) Distance to Route



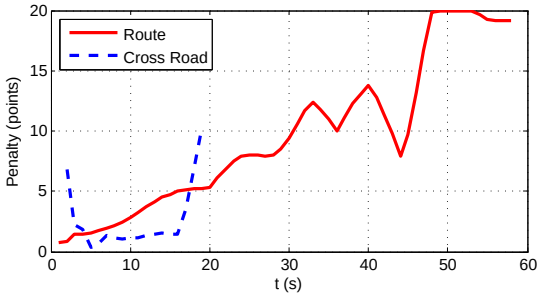
(d) Angle Error



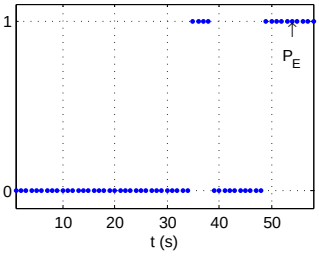
(e) Speed



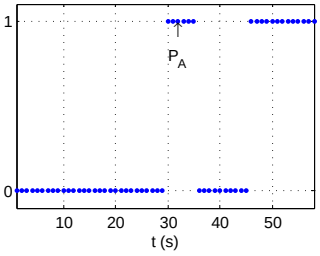
(f) Normalized Distance Derivative



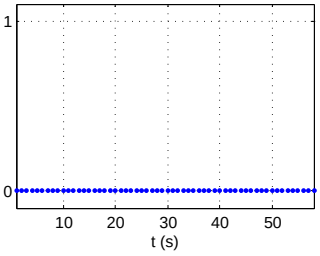
(g) Penalty Function



(h) Off Track - Existing Algorithm

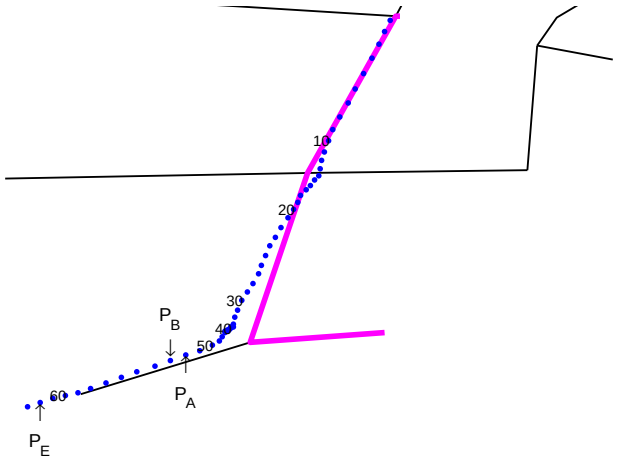


(i) Off Track - Algorithm A



(j) Off Track - Algorithm B

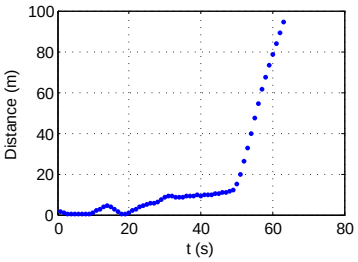
Figure A.5: Test Case E



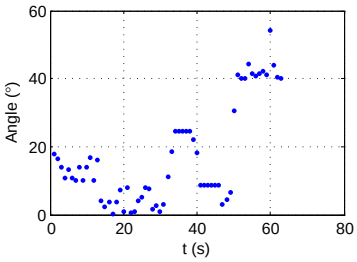
(a) Map

Existing Algorithm	12 s
Algorithm A	2 s
Algorithm B	3 s

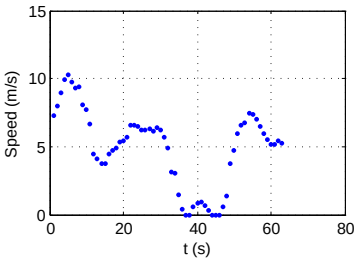
(b) Response Times



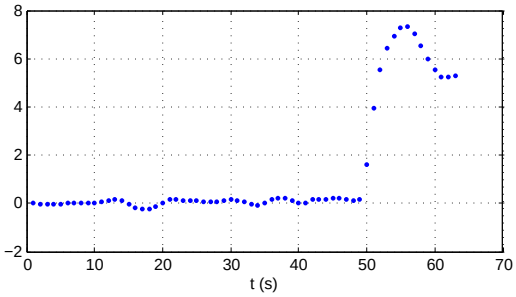
(c) Distance to Route



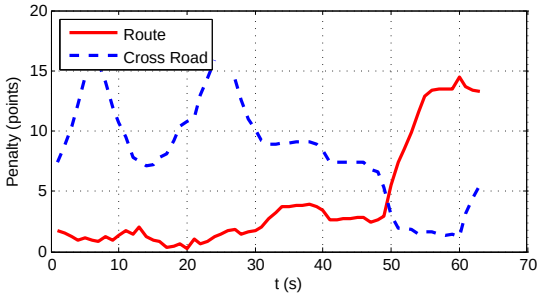
(d) Angle Error



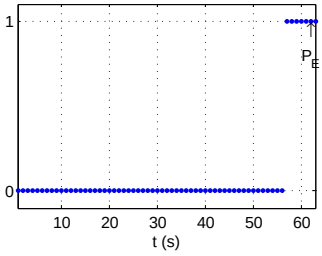
(e) Speed



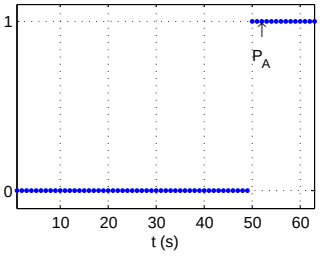
(f) Normalized Distance Derivative



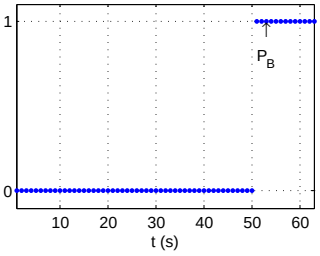
(g) Penalty Function



(h) Off Track - Existing Algorithm

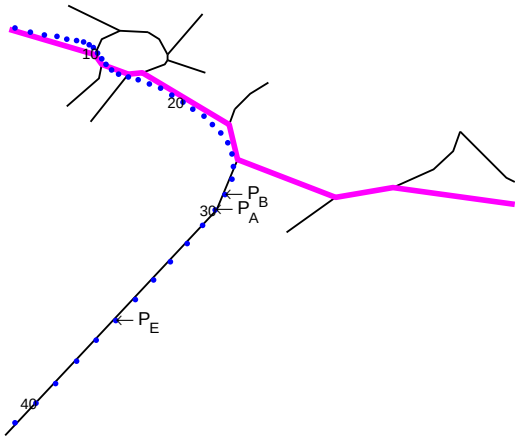


(i) Off Track - Algorithm A



(j) Off Track - Algorithm B

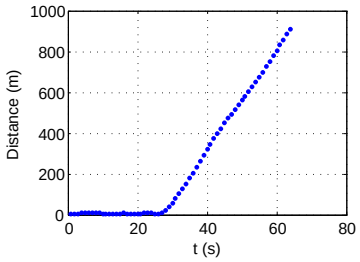
Figure A.6: Test Case F



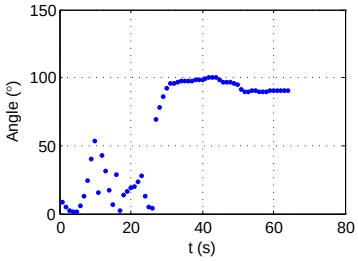
(a) Map

Existing Algorithm	9 s
Algorithm A	3 s
Algorithm B	2 s

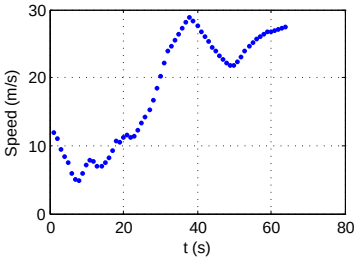
(b) Response Times



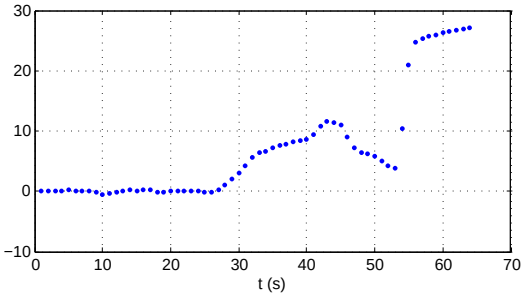
(c) Distance to Route



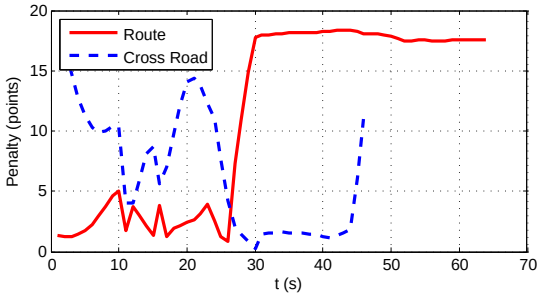
(d) Angle Error



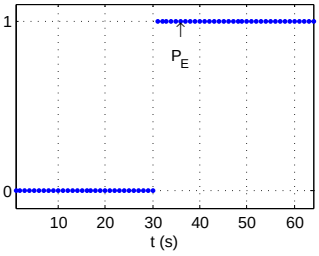
(e) Speed



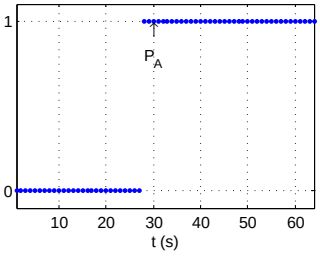
(f) Normalized Distance Derivative



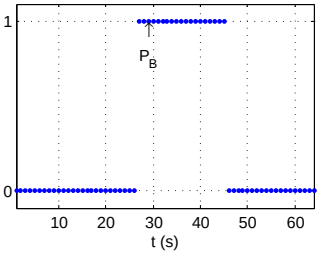
(g) Penalty Function



(h) Off Track - Existing Algorithm

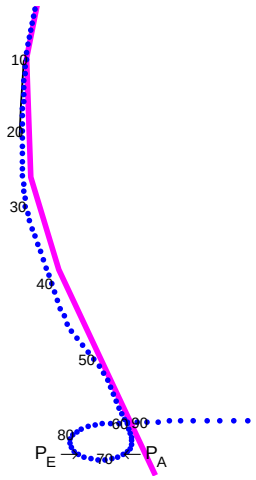


(i) Off Track - Algorithm A



(j) Off Track - Algorithm B

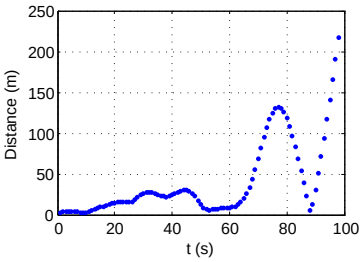
Figure A.7: Test Case G



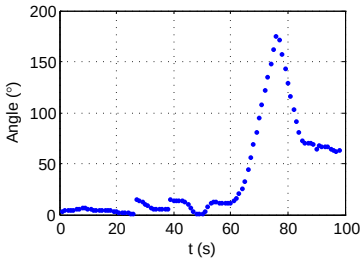
(a) Map

Existing Algorithm	65 s
Algorithm A	58 s
Algorithm B	-

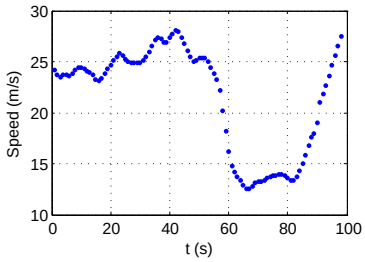
(b) Response Times



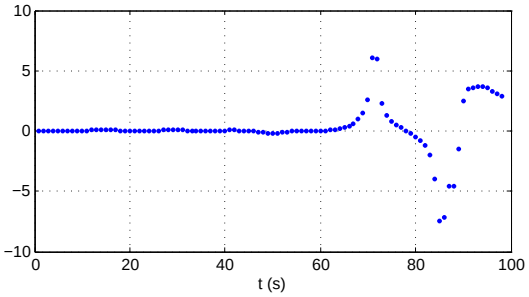
(c) Distance to Route



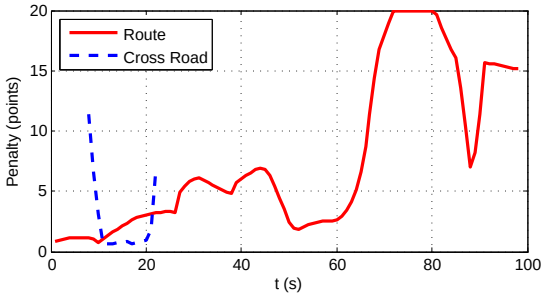
(d) Angle Error



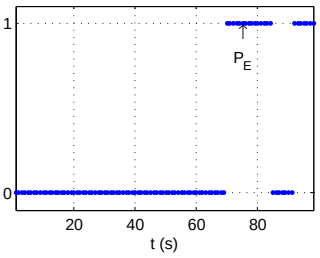
(e) Speed



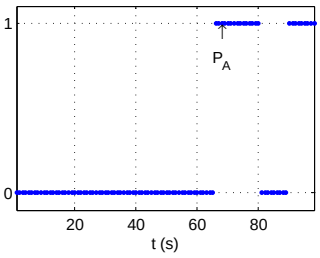
(f) Normalized Distance Derivative



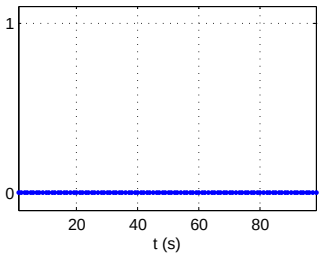
(g) Penalty Function



(h) Off Track - Existing Algorithm

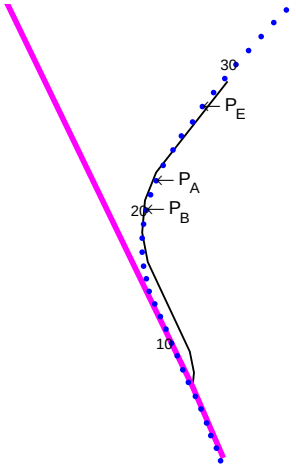


(i) Off Track - Algorithm A



(j) Off Track - Algorithm B

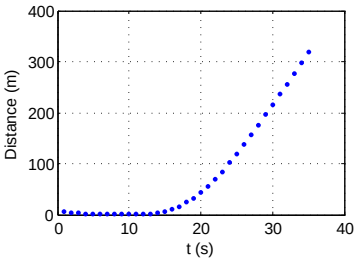
Figure A.8: Test Case H



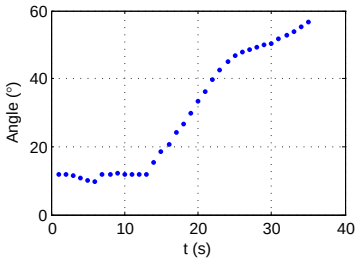
(a) Map

Existing Algorithm	20 s
Algorithm A	15 s
Algorithm B	13 s

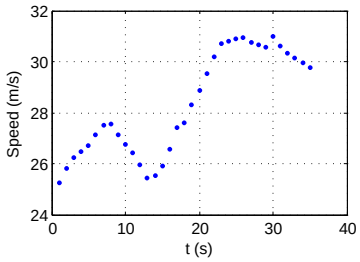
(b) Response Times



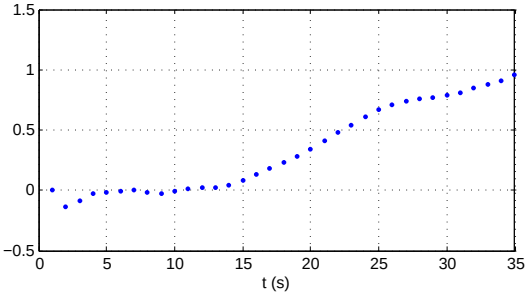
(c) Distance to Route



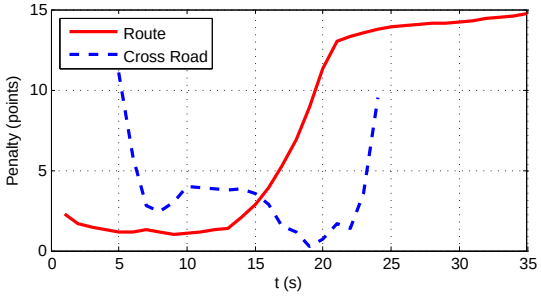
(d) Angle Error



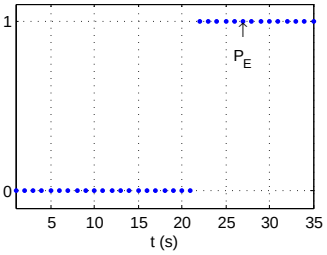
(e) Speed



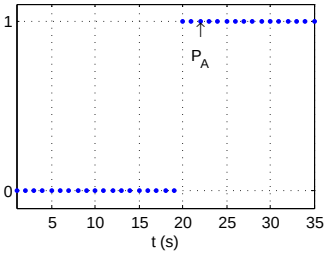
(f) Normalized Distance Derivative



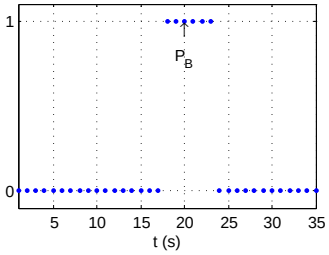
(g) Penalty Function



(h) Off Track - Existing Algorithm

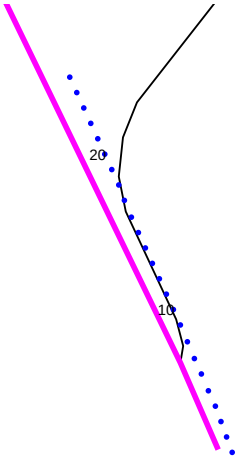


(i) Off Track - Algorithm A



(j) Off Track - Algorithm B

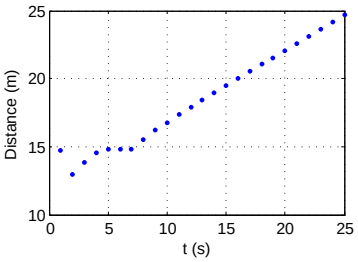
Figure A.9: Test Case I



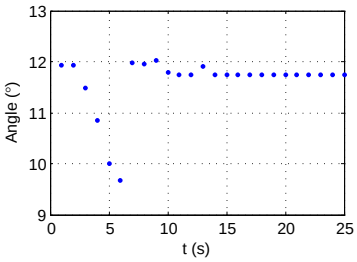
(a) Map

Existing Algorithm	No
Algorithm A	No
Algorithm B	No

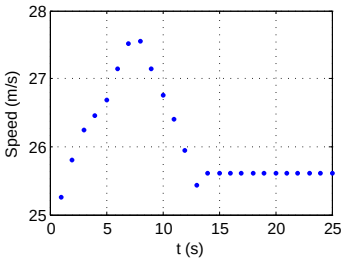
(b) False Off Track



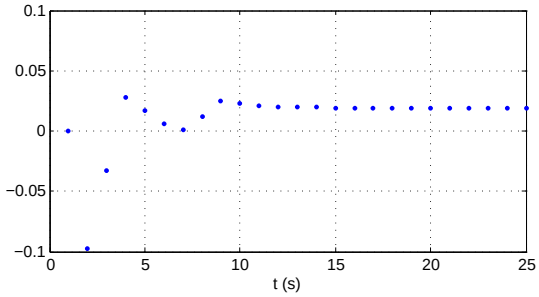
(c) Distance to Route



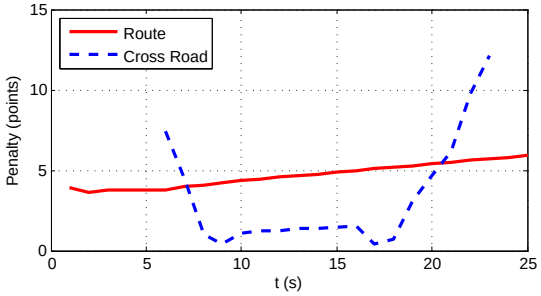
(d) Angle Error



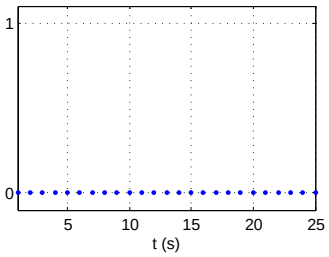
(e) Speed



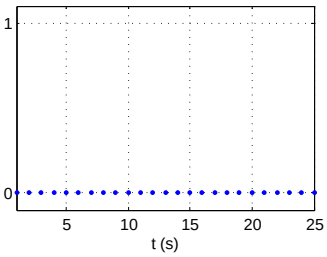
(f) Normalized Distance Derivative



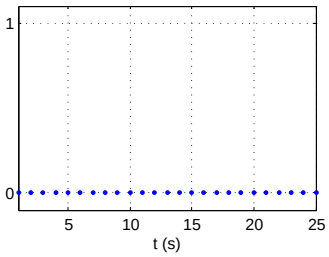
(g) Penalty Function



(h) Off Track - Existing Algorithm

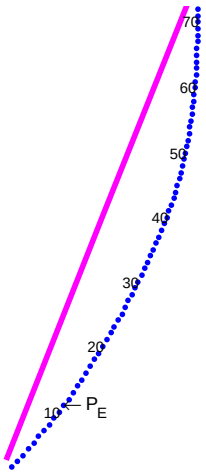


(i) Off Track - Algorithm A



(j) Off Track - Algorithm B

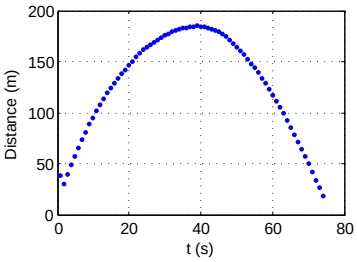
Figure A.10: Test Case J



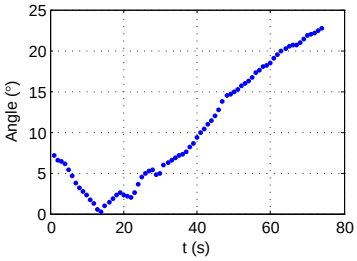
(a) Map

Existing Algorithm	Yes
Algorithm A	No
Algorithm B	No

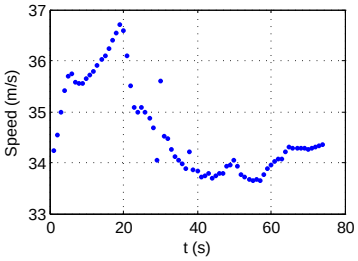
(b) False Off Track



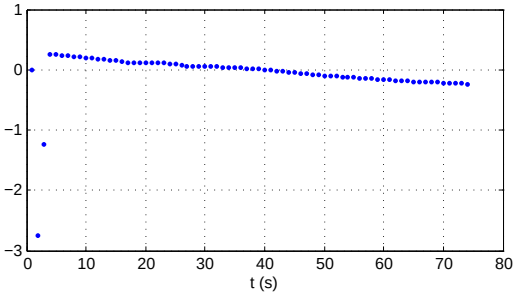
(c) Distance to Route



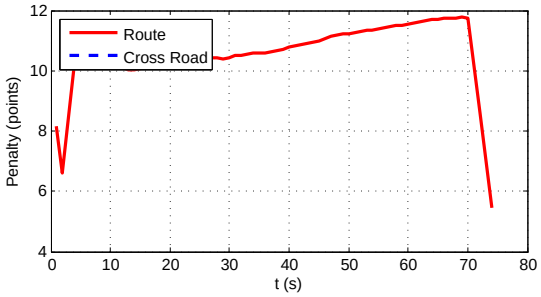
(d) Angle Error



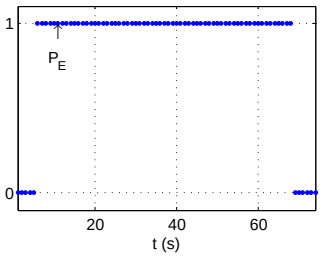
(e) Speed



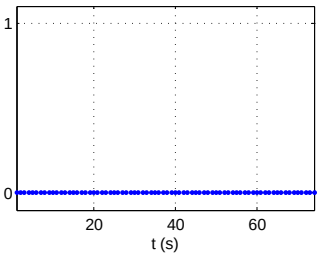
(f) Normalized Distance Derivative



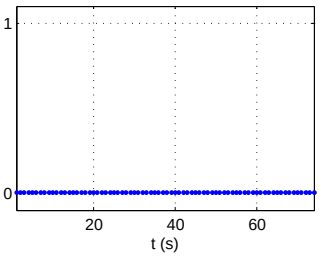
(g) Penalty Function



(h) Off Track - Existing Algorithm

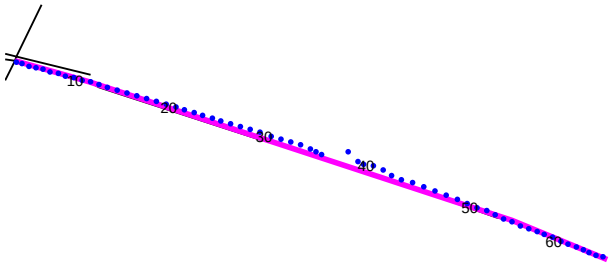


(i) Off Track - Algorithm A



(j) Off Track - Algorithm B

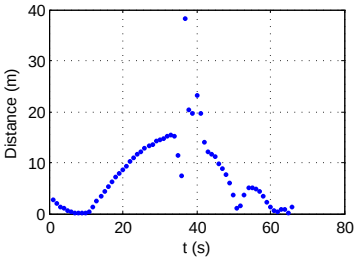
Figure A.11: Test Case K



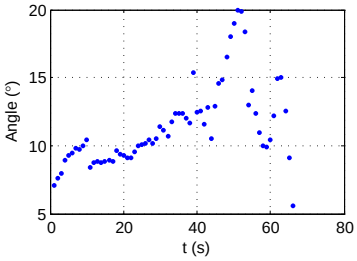
(a) Map

Existing Algorithm	No
Algorithm A	No
Algorithm B	No

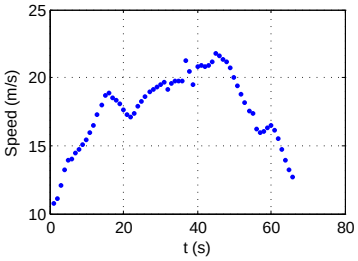
(b) False Off Track



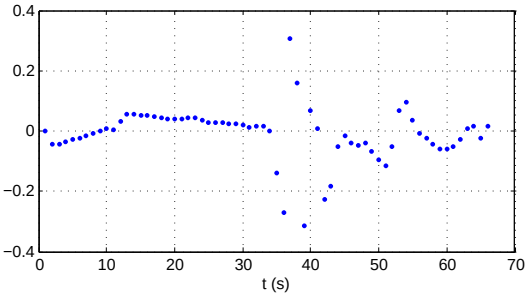
(c) Distance to Route



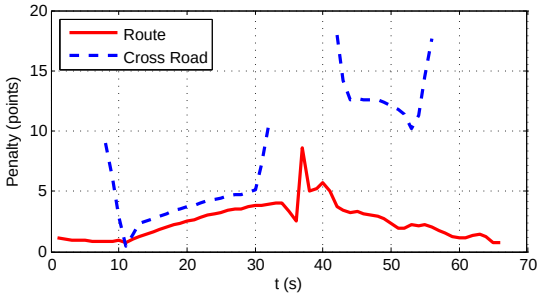
(d) Angle Error



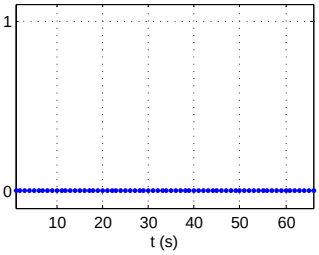
(e) Speed



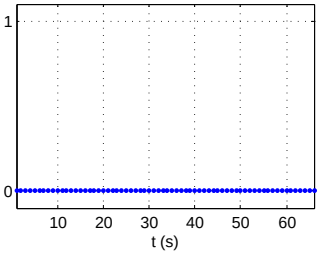
(f) Normalized Distance Derivative



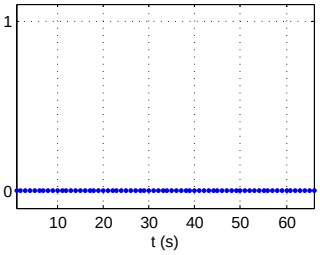
(g) Penalty Function



(h) Off Track - Existing Algorithm

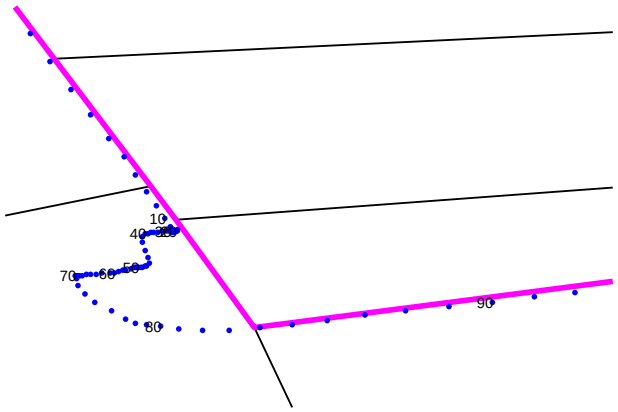


(i) Off Track - Algorithm A



(j) Off Track - Algorithm B

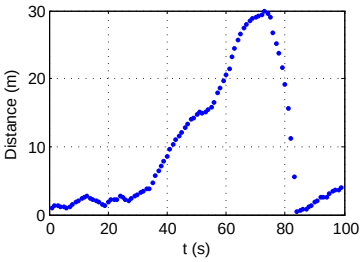
Figure A.12: Test Case L



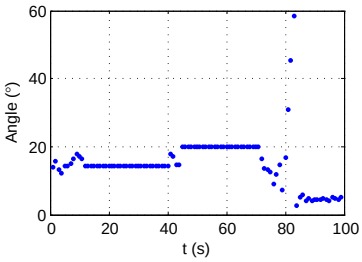
(a) Map

Existing Algorithm	No
Algorithm A	No
Algorithm B	No

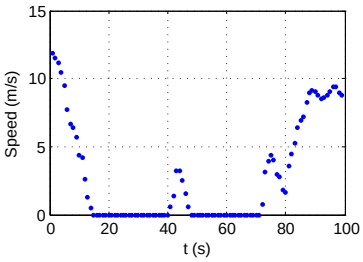
(b) False Off Track



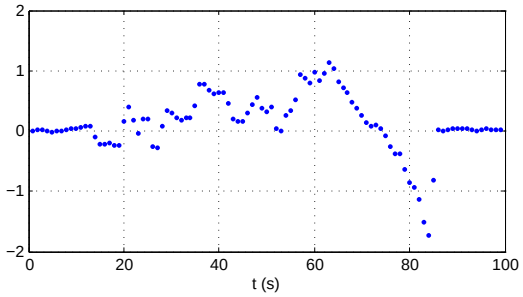
(c) Distance to Route



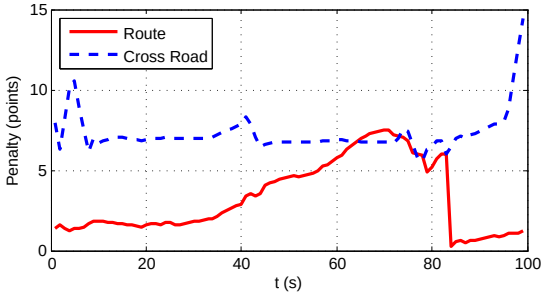
(d) Angle Error



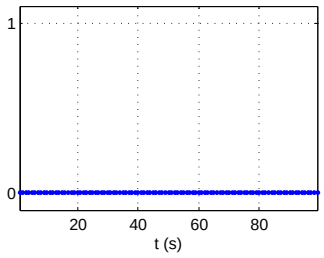
(e) Speed



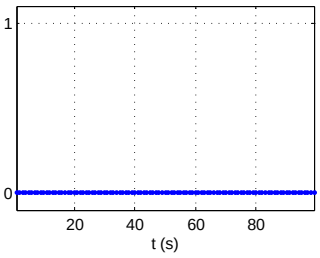
(f) Normalized Distance Derivative



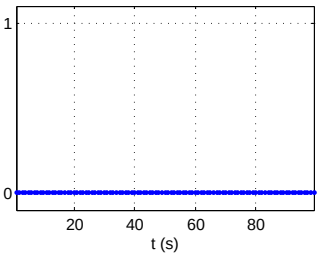
(g) Penalty Function



(h) Off Track - Existing Algorithm

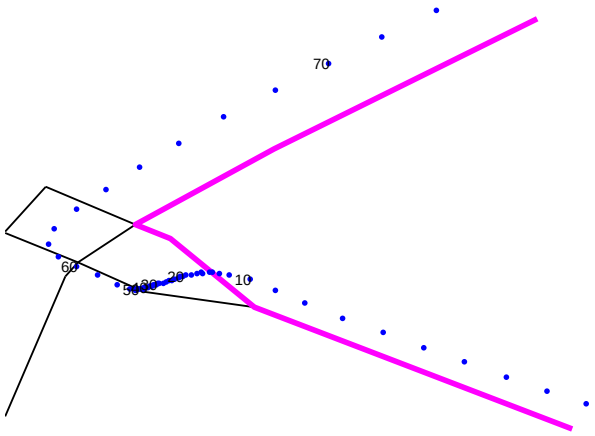


(i) Off Track - Algorithm A



(j) Off Track - Algorithm B

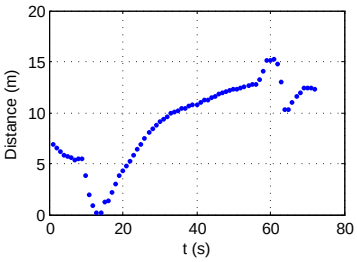
Figure A.13: Test Case M



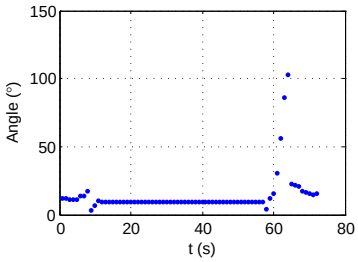
(a) Map

Existing Algorithm	No
Algorithm A	No
Algorithm B	No

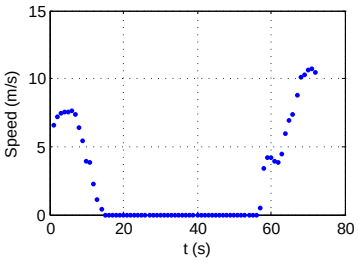
(b) False Off Track



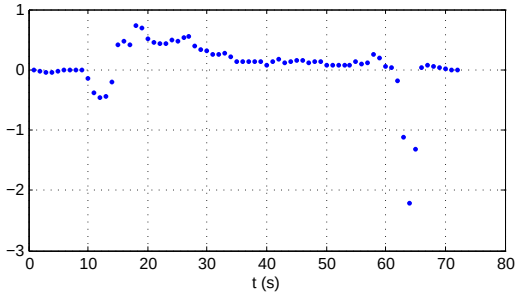
(c) Distance to Route



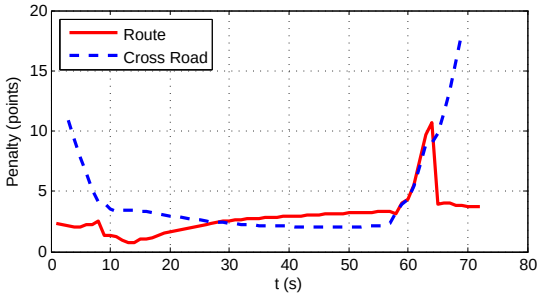
(d) Angle Error



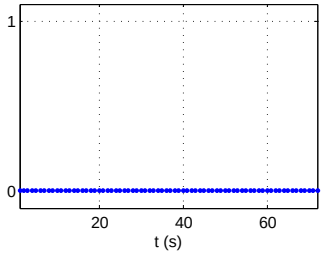
(e) Speed



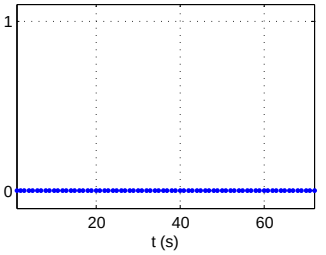
(f) Normalized Distance Derivative



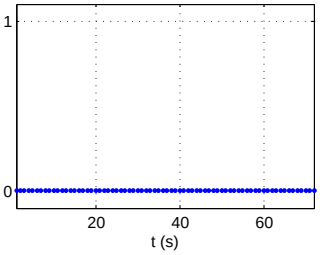
(g) Penalty Function



(h) Off Track - Existing Algorithm

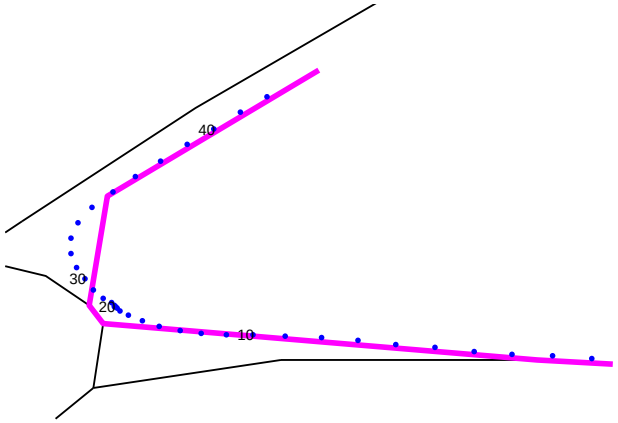


(i) Off Track - Algorithm A



(j) Off Track - Algorithm B

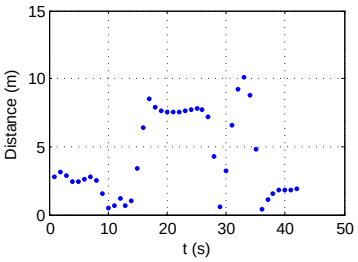
Figure A.14: Test Case N



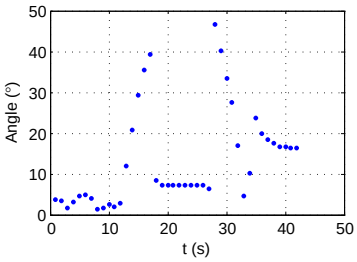
(a) Map

Existing Algorithm	No
Algorithm A	No
Algorithm B	No

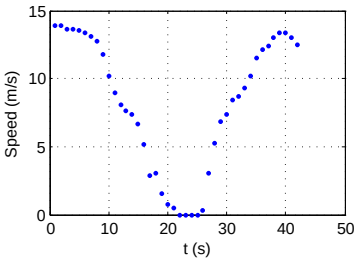
(b) False Off Track



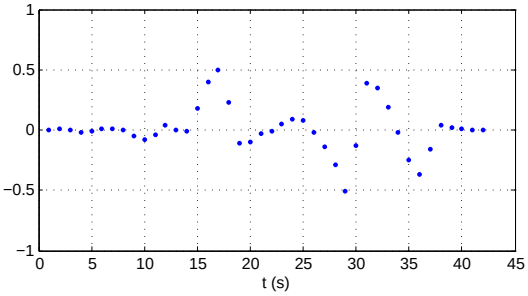
(c) Distance to Route



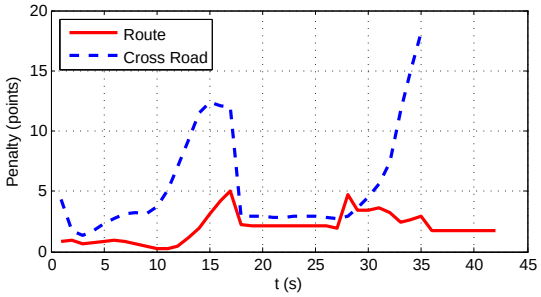
(d) Angle Error



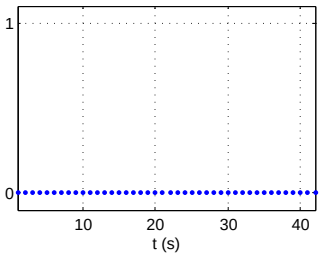
(e) Speed



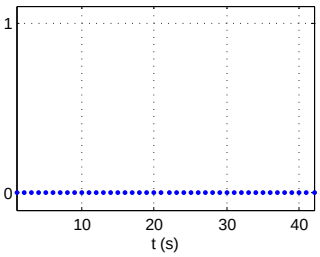
(f) Normalized Distance Derivative



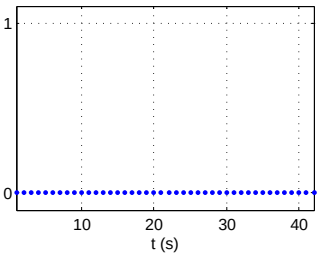
(g) Penalty Function



(h) Off Track - Existing Algorithm

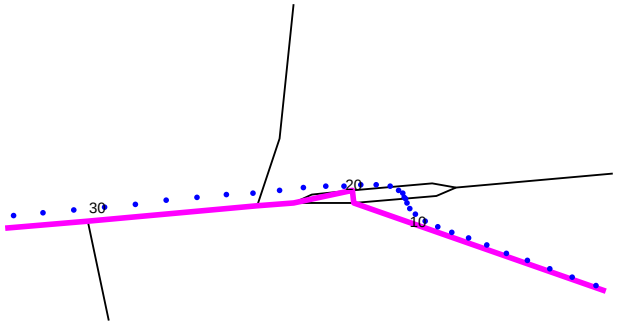


(i) Off Track - Algorithm A



(j) Off Track - Algorithm B

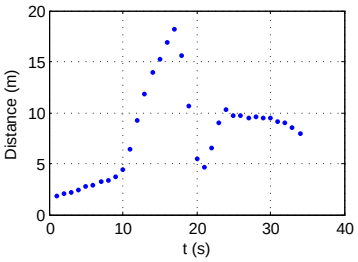
Figure A.15: Test Case O



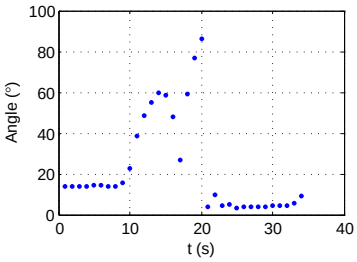
(a) Map

Existing Algorithm	No
Algorithm A	No
Algorithm B	No

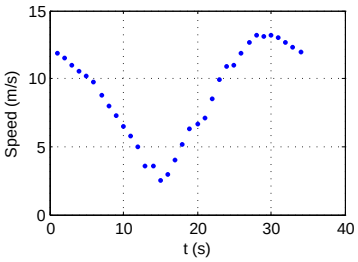
(b) False Off Track



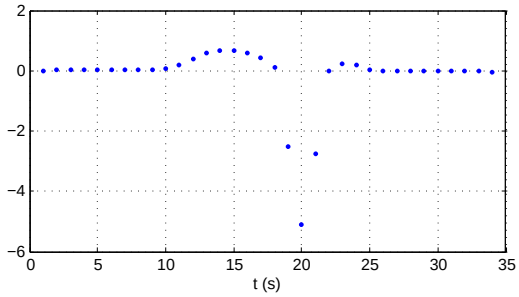
(c) Distance to Route



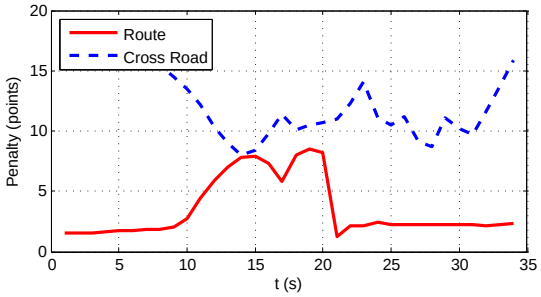
(d) Angle Error



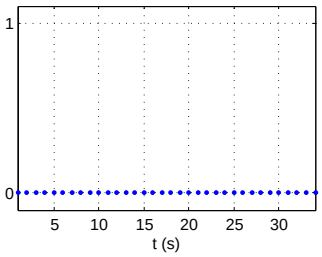
(e) Speed



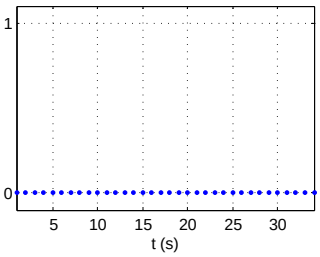
(f) Normalized Distance Derivative



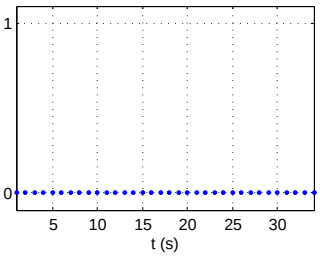
(g) Penalty Function



(h) Off Track - Existing Algorithm

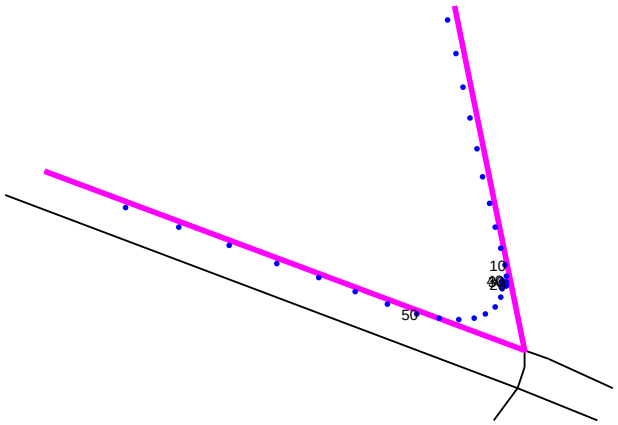


(i) Off Track - Algorithm A



(j) Off Track - Algorithm B

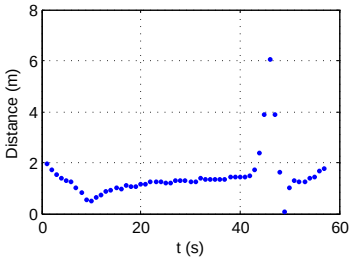
Figure A.16: Test Case P



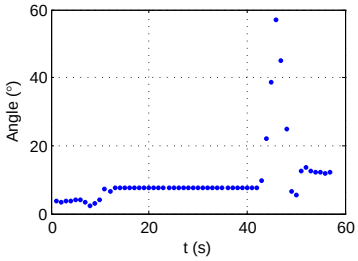
(a) Map

Existing Algorithm	No
Algorithm A	No
Algorithm B	No

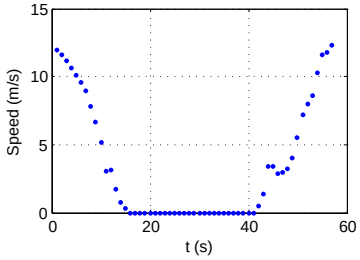
(b) False Off Track



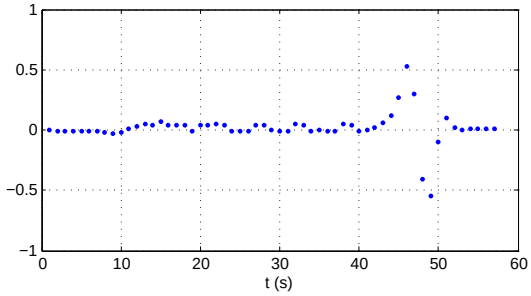
(c) Distance to Route



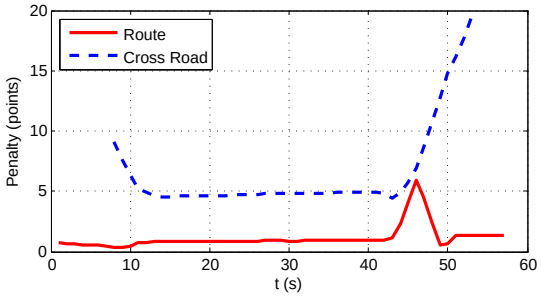
(d) Angle Error



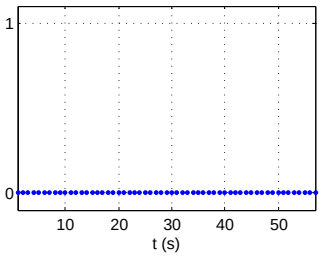
(e) Speed



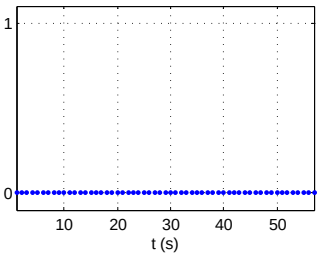
(f) Normalized Distance Derivative



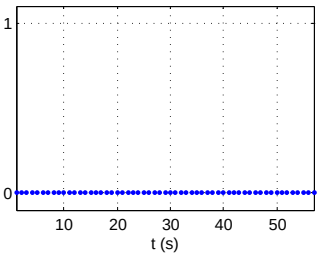
(g) Penalty Function



(h) Off Track - Existing Algorithm

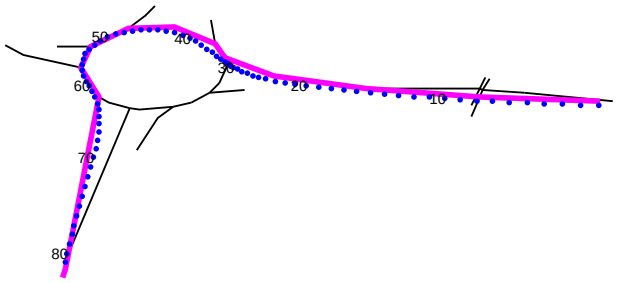


(i) Off Track - Algorithm A



(j) Off Track - Algorithm B

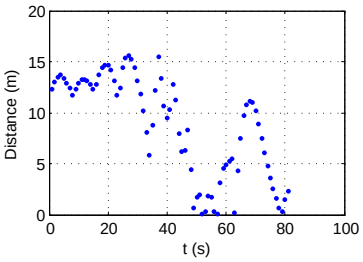
Figure A.17: Test Case Q



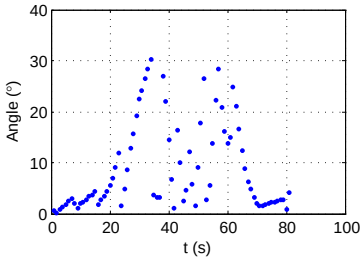
(a) Map

Existing Algorithm	No
Algorithm A	No
Algorithm B	No

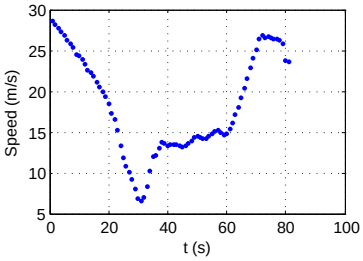
(b) False Off Track



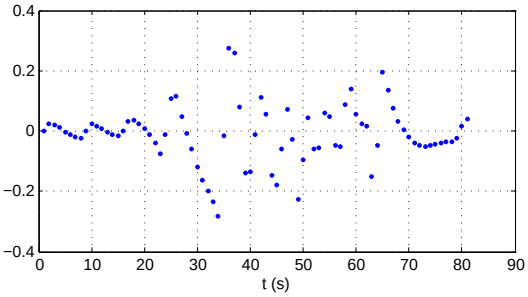
(c) Distance to Route



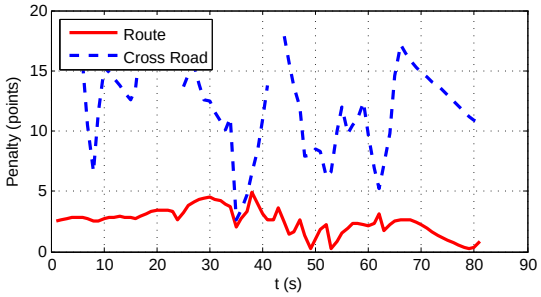
(d) Angle Error



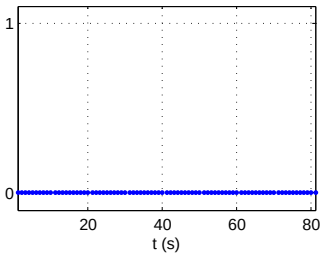
(e) Speed



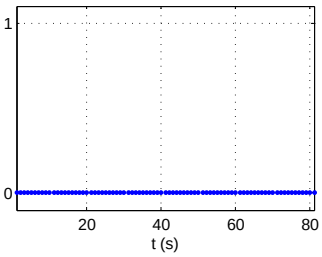
(f) Normalized Distance Derivative



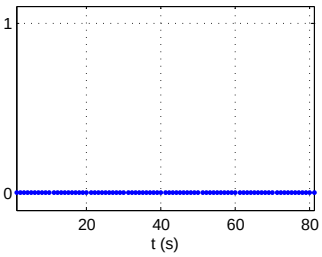
(g) Penalty Function



(h) Off Track - Existing Algorithm



(i) Off Track - Algorithm A



(j) Off Track - Algorithm B

Figure A.18: Test Case R

Bibliography

[1] Introduction to the Global Positioning System for GIS and TRAVERSE
Available at: http://www.cmtinc.com/gpsbook/index.htm
May 2007

[2] Trimble - GPS Tutorial
Available at: http://www.trimble.com/gps/whatgps.shtml
May 2007

[3] The Evolution of GPS
Available at: http://illumin.usc.edu/article.print.php?articleID=137
May 2007

[4] GPS World
Available at: http://www.gpsworld.com/gpsworld/
June 2007

[5] All About GPS
Available at: http://www.kowoma.de/en/gps/index.htm
June 2007

[6] A Short Guide to Celestial Navigation
Available at: http://www.celnav.de/
June 2007

[7] NMEA 0183 - Standard For Interfacing Marine Electronic Devices
Version 3.01
January 1, 2002