FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

FEUP

Geospatial Data Processing for GPS Navigation Systems

Tiago Ribeiro Mota Freitas

Report of Project

Master in Informatics and Computing Engineering

Supervisor: Dr. António Coelho, PhD

Co-Supervisor: Dr. Rosaldo Rossetti, PhD

March 2009

Geospatial Data Processing for GPS Navigation Systems

Tiago Ribeiro Mota Freitas

Report of Project
Master in Informatics and Computing Engineering

Approved in oral examination by the committee:

Chair: Prof. Eugénio Costa Oliveira, PhD

External Examiner: Dr. Nuno Lau, PhD (University of Aveiro)
Internal Examiner: Dr. António Coelho, PhD
Internal Examiner: Dr. Rosaldo Rossetti, PhD

20^{th} March, 2009

Abstract

Personal Navigation Assistants are a technology that is widely in use. PNAs depend on an updated digital map for accurate positioning and correct routing. Since updating digital maps is a complicated and costly task, it was necessary to find other means different from the usual ones (aerial imagery and use of special vehicles for field surveying).

A way of decreasing costs associated with map updating is by using passively collected GPS positions from PNA during navigation. By processing collected points it is possible to detect changes to the real road network.

Before using collected positions it is necessary to determine the correct positioning of points in the digital map (selection of the correct segment and position with in), which is called Map-matching. A method to map match points is presented that uses normal GPS information and by the use of topological information creates traveled routes. Processement of the matched points permits several detection of changes to be made that include new roads, new roundabouts, incorrect road geometry, incorrect authorised direction, incorrect manoeuvre prohibition and junction position correction.

The proposed method includes a database model, for archiving the collected data and results. Visualisation of output results is done through a GIS editor where a user can see the changes and correct the map.

An implementation of the system was created and the developed algorithms were tested with a small set of points.The results were satisfactory and demonstrated that the proposed methods are useful in recognizing changes and that this method can be used in order to reduce map update costs.

Resumo

Assistentes Pessoais de Navegação (PNA) são uma tecnologia que está amplamente divulgada. Os PNAS dependem de uma cartografia digital actualizada para posicionamento preciso do utilizador e possíveis direcções. Como a actualização de mapas digitais é uma tarefa complicada e dispendiosa, foi necessário encontrar meios que não utilizassem apenas imagens aéreas ou veículos especializados para fazer a sua actualização.

Uma forma de diminuir os custos associados à actualização de mapas é a utilização das posições recolhidas passivamente por dispositivos GPS, presentes nos PNA, durante a navegação. Através do processamento dos pontos recolhidos é possível detectar alterações à rede rodoviária.

Antes das posições recolhidas poderem ser utilizadas, é necessário determinar o posicionamento correcto dos pontos GPS no mapa digital (selecção correcta da estrada e posicionamento do ponto), através de um método chamado Map-matching. É apresentado um método que faz o mapeamento a partir da informação habitual vinda de um dispositivo GPS e pelo uso da informação topológica existente no mapa.

O processamento dos ponto mapeados permite a detecção de diversas alterações como novas estradas, novas rotundas, geometria incorrecta da via, direcção incorrecta de tráfego, manobras proibidas incorrectamente especificadas e a correcção da posição dos cruzamentos.

O método proposto inclui os algoritmos de detecção e um modelo de base de dados, para arquivar os pontos recolhidos e guardar as alterações encontradas. A visualização dos resultados é feita através de um editor de GIS onde o utilizador pode ver as alterações e fazer a correcção ao mapa.

Foi desenvolvida uma implementação do sistema para testar os algoritmos desenvolvidos com um pequeno conjunto de pontos. Os resultados foram satisfatórios e demonstraram que os métodos propostos são úteis no reconhecimento de alterações da rede viária e que este método pode ser utilizado para reduzir os custos derivados da actualização de mapas.

Acknowledgements

The development of this project was a major step in my life that was greatly accomplished with help and encouragement by everyone. A thank you to FEUP, and every teacher that accompanied me during my academic journey, for all the knowledge I could get that will certainly help me along my life. I have to thank NDrive and InfoPortugal for the great reception and support that they gave me, especially my supervisor Eng. Eurico Inocêncio and Eng. Egídio Moutinho for their cooperation, patience and knowledge. I am also grateful for the assistance and guidance that my supervisors Prof. António Coelho and Prof. Rosaldo Rossetti gave me during the development of this project. I give thanks to my family for the unconditional support, comfort and encouragement that they always gave me. To all my friends, that gave me advice or joy, a big thank you.

Tiago Mota Freitas

"You can't jump down the stairs in one leap,
however much you might wish to, and you even more surely can't jump up it,
but one step and then the next and there you are,
at the top or the bottom and not a bit out of breath or discomposed."

Elizabeth Aston, The Exploits & Adventures of Miss Alethea Darcy, 2005

Contents

List of Figures

Abbreviations

GPS	Global Positioning System
PND	Personal Navigation Device
PNA	Personal Navigation Assistant
HDOP	Horizontal Dilution of Precision
VDOP	Vertical Dilution of Precision
PDOP	Positional Dilution of Precision
CSV	Comma Separated Values
GIS	Geographic Information Systems
API	Application Programming Interface
SHP	ESRI Shapefile Format

Chapter 1

Introduction

A Personal Navigation Assistant (PNA) depends on an updated road network for accurate positioning calculation and correct routing. Unfortunately, updating maps is a difficult and expensive task.

Since PNA users use them to aid in navigation from one place to another, it should be possible to use information gathered by those devices to adjust the maps to the real road network.

This dissertation presents algorithms to process user's coordinates, retrieved from a PNA during trips on the road network, to detect possible changes and reduce costs of updating maps.

1.1 Scope

Navigation systems based on GPS technology are widely spread across the world. An increasing request to map companies is to provide accurate and complete digital maps of the road network [NG00]. As updating digital networks by the usual means (aerial imagery, field teams) is highly expensive, which in turn increases the map price to the consumer, it is necessary to develop new ways of detecting changes and implementing corrections to the map network with a low cost.

A solution that provides low cost information is the gathering of GPS points acquired by existing PNA devices on the road and their use to detect changes by comparing user positions with present digital maps.

NDrive Navigation Systems S.A. is a portuguese company in the field of navigation technologies that works in conjunction with InfoPortugal, a company that provides digital contents and maps, to supply the market with leading products in the navigation fields.

This project was developed for NDrive in order to reduce the expenses of map updates and to improve user satisfaction attained from the use of high quality and complete maps.

Digital maps are a representation of the real road network and are composed of polygons representing spaces, road segments, junctions and manoeuvres. Only a prepared application is able to "understand" the digital map and, by using the information stored, to give navigation directions and positioning information when coupled with a GPS device. GPS devices produce data with positioning information (latitude, longitude, altitude, etc.) that can be used in routing applications.

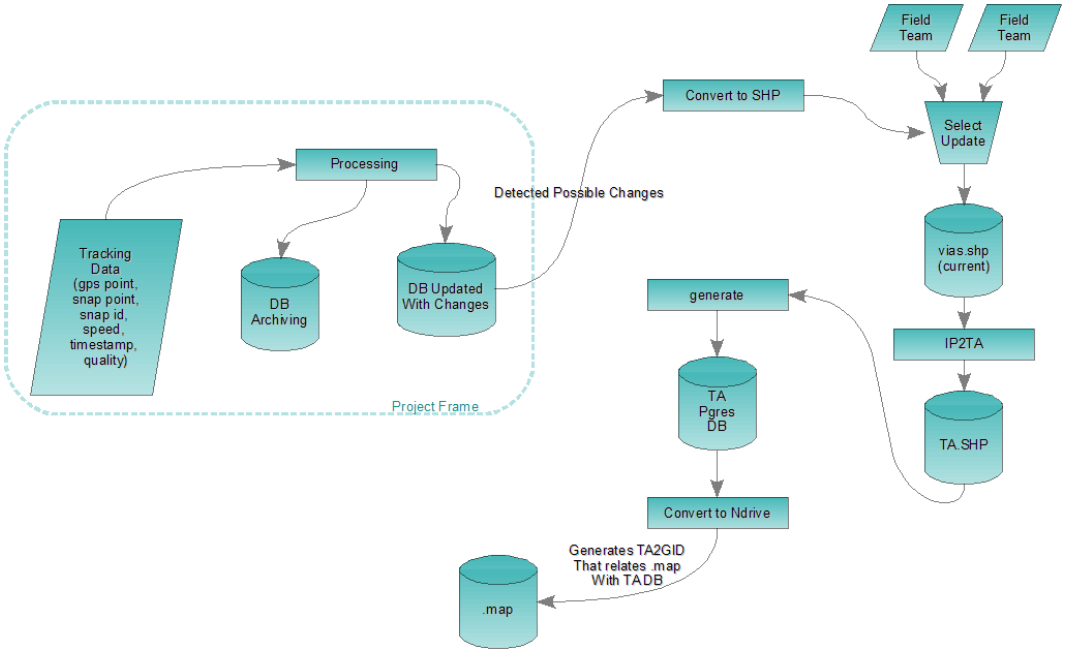


Figure 1.1: Project Scope

Figure 1.1 introduces the scope of this application and how this project could be integrated to the work flow of the company. This project will complete the way map updates are integrated into the existing digital map (vias.shp) in conjunction with field team data and then adapted to other uses.

TeleAtlas, one of the leading companies in the field, provides some of the maps that NDrive uses on the navigation software. NDrive currently uses databases in the TeleAtlas format to generate data files for use with navigation applications. This project is developed with the TeleAtlas format in mind and adapts to its existence.

1.2 Project

A PNA with an outdated map is hard to use since positioning information, derived from GPS position and a map-matching algorithm, might be wrong and routing from A to B

might not work as expected if there were a big change in the road network.

Present Personal Navigation Assistants are using maps that were created based on data collected from field teams using special vehicles and aerial image. Although this is a reliable form of retrieving network information it is also a very expensive one.

A possible solution to lower costs is to use information retrieved by PNA's that is being discarded. By saving PNA's position along the trip it is possible to later compare the map with the road network and try to fit the map.

Since GPS is not accurate enough it is necessary to use many trips to accurately detect a possible change in the network topology.

This project will try to develop algorithms to detect road network changes and generate possible corrections that might be used in conjunction with aerial images to adapt the map.

1.3 Objectives

The main goal of this project is to detect changes in network topology and present a possible correction to the digital geometry.

All the algorithms developed should only use the usual data from GPS devices (position, direction, speed, time, quality).

Provided GPS points do not include segment relation, it is necessary to compute the correct location on map by using a matching algorithm. With snapped points available it should be possible to detect the following changes:

- New roads;
- New roundabouts;
- Incorrect prohibited maneuvers;
- Incorrect traffic direction;
- Junction position;
- Wrong road geometry.

The system outputs results to a database and visualisation is possible by using a GIS editor/viewer.

The user, by analyzing the output of the application on the viewer, should be able to correct the digital map.

1.4 Dissertation Structure

The next chapter (2) presents an overview of the present algorithms and technologies as well as the selection of tools for the implementation of the algorithms.

Chapter 3 presents the developed algorithms for changes detection. For each detection a clarification of the problem is described followed by the description of the algorithm.

Implementation and Results, chapter 4, demonstrates a practical implementation of the algorithms introduced in chapter 3 and includes the achieved results with the implemented algorithms.

The last chapter (5) is a summary of the complete project and introduces guidelines for future work.

Appendix A and B are the configuration files of the developped applications that includes the variable values used in each algorithm.

Chapter 2

State of the Art

This chapter exposes the investigation process that preceded the developed methodology. A general presentation of the current state of navigation technology and a selection of the technologies that were used.

2.1 Personal Navigation Assistants

Personal Navigation Assistants are devices that provide a user with a map, current position and directions [WBK00]. There are three ways of determining user's location [WBK00]:

- By Dead-Reckoning using devices such as accelerometers it is possible to update user's position;
- Ground-based beacon that broadcasts its location to nearby users;
- a form of radio/satellite positioning system like GPS or the European's Galileo.

Obtaining a position is not complicated but since GPS positions have high accuracy errors [YW04] [Zho04] [sYpKsC05], it's very important to use algorithms to adapt (snap) user's location to a position in a map.

Another source of errors is the digital map [NG00] [MHA04] [QON06]. Road centrelines in digital maps may have a high degree of errors since a representation of a centreline does not define the boundaries of the road [sYpKsC05].

Personal Navigation Assistants completely depend on the accuracy of the map to provide the user with good directions (during route navigation). Since updating a digital map is an unstoppable task, it was necessary to improve the way that research of new areas, and even mapping, is done.

The following sections present some of the current advances that permit fast and correct map-matching of points to locations and even automatic map creation.

2.2 Road Network Updating

PNAs are useful because of the digital representation of the world and the possibility to get routes from an origin to a destination.

Changes to the road network confuses the route the PNA gives and sometimes prevents the correct map-matching. Road network updating is the process of rectifying the digital map so it better represents the road network. Since it is a highly expensive process when it is done by field teams surveying the area, some processes were developed that would allow corrections to maps through other means.

2.2.1 Aerial Imagery

Acquisition of data to update digital maps can be obtained from aerial imagery using processes that are time-consuming, costly and error-prone like digitising with a hand-operated tablet [TZ03].

Some algorithms that use computer vision to process aerial photos had been developed in the past and proved effective although dependent on the quality of the data [EBPB07].

2.2.2 GPS Data Processing

The number of PNAs in use has increased in the last years. By using users' collected data it is possible to detect changes to an existing map network at a fraction of the costs [Fri05].

2.2.2.1 Correction of Spatial Alignment

Work has been done in the direction of correcting spatial alignment (geometry) of streets by the use of existing data sets composed of GPS points. Focus was on the automatic adaption of the road network by comparison of user positions and spatial averaging against the existing digital maps. The algorithms presented on [Fri05] use modification of position of nodes and shape points (defining nodes of a segment) to bend and adapt segment to the real road network. This work introduced generally good results on the tested segments.

2.2.2.2 Automatic Map Processing System

The work of Tom and Zantout introduced a way of updating a digital map based on GPS data. The features extracted from GPS data are intersections, lanes and segments. The results proved that it was possible to update a digital map using GPS data and user info [TZ03].

2.2.2.3 Segment Grouping

An alternative approach to the classification of unknown roads use passively collected data from navigation systems. This approach introduces the possibility of using an artificial neural network to determine the geometry of the new roads. This work covered only the detection of some of the inputs for the determination of the road group (road, minor roads, local streets) using a Snap-Drift Neural Network [EBPB07].

2.2.2.4 Lane information

This paper presents an approach that allows detection of number of lanes and lane positioning of a vehicle on a road by the use of GPS positions. Lane information can be used in various applications like lane departure warning or lane level navigation to aid in the correct placement for an exit. This approach used centreline detection to compare lane positions [RLW99].

2.2.2.5 TomTom MapShare

TomTom Mapshare is a system that depends on the user for detection of map changes. Users by using their TomTom PNA are able to add changes to the digital map through a simple interface, as seen in figure 2.1.

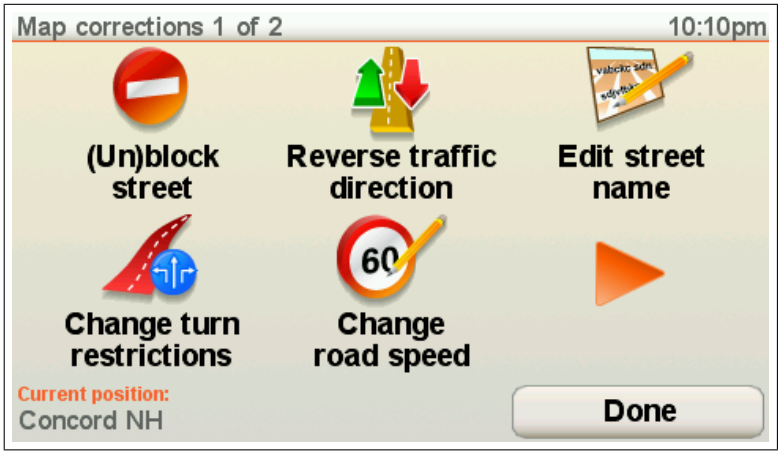


Figure 2.1: Mapshare Interface

Mapshare permits users to add changes in order to:

- Block / unblock street - allows marking a street as blocked to traffic from a direction;
- Reverse traffic direction - reverses allowed traffic direction;
- Edit street name - change the street name;

- Add missing Points of Interest - add new points of interest;
- Edit Points of Interest. - edit existing points of interest (location, name);
- Edit Speed limits - change speed limit;
- Change Turn restrictions - correct possible maneuvres;
- Report other error - through this interface it is possible to communicate more complex problems and these will not be visible from Mapshare:
  - Existing Street - a street that shouldn't exist;
  - Missing Street - a new street not included on the map;
  - City - Report a city error;
  - POI - Report a POI error;
  - Highway entrance/exit - Report an entrance or exit to the highway;
  - Zip Code - Errors with zip codes;
  - Roundabout - Problems with roundabouts;
  - Other - Allows a text to be written to explain the problem.

Every change made to the digital map by the user is applied instantly and is used during routing, speed alerts and navigation.

When the user connects the PNA to the Internet the introduced changes are shared, if the user allows it, and other users' changes are downloaded to the device. There are different levels of trust on obtained corrections. It is possible to receive changes verified by TomTom, changes submited by trusted sources, by many people or by some people. [tom09b]

Mapshare does not allow addition of new roads or changes to the physical geometry so to really have an updated map it is necessary to buy it.

2.2.2.6 TomTom IQ-Routes

> "This new improved technology calculates routes based on the real average speeds measured on roads every day compared to speed limits. This uses historical data that TomTom users have been adding to over the years. It will always provide users with the smartest route hour-by-hour, day-by-day, saving them time, money and fuel." [tom09a]

IQ-Routes is a new feature of TomTom devices that allows reduction of time spent on daily traffic by the use of statistical speed information for street segments obtained from users.

By using IQ-Routes it is possible to reduce fuel consumption by avoiding usual traffic jams that leads also to a state of increased stress.

This system uses speed for daily intervals of five minutes and differentiates week days from weekend. [tom09a]

Conventional calculation of speed is made based on road specifications and speed limit. Usually conventional speed is overrated leading to unfavourable routes through a degraded road or passing on a jammed junction.

2.3 Map-Mathing GPS to the real road network

Map-matching, or snapping, is the process of relating a GPS coordinate to a position on a digital map with a reference to a precise street and a position within it. Although a human can match points to a map easily, this is a complex task to be implemented on a computer [Gre02].

Different levels of difficulty for the problem exist [Gre02]:

- Limited known route: map-matching points to a limited subset of the digital map is easier; an example is the management of a bus fleet where every bus should be on the usual route;
- Route Following: if the map-matching uses routing information and the GPS points follow the route it is easy since only a subset of the segments need to be compared;
- No knowledge is assured; this is the most common application; this occurs when there is no knowledge that may reduce the number of possible segments.

Paper [QON07] categorises map-matching algorithms into three groups:

- Geometric: algorithms that solely use the information derived from geometry definitions;
- Topological: uses information from defined geometry and topological information of the maps (connected segments, distance, prohibited manoeuvres, speed, authorised traffic directions);
- Probabilistic: use of probabilities in the calculation of confidence around the points;
- Advanced Map-matching algorithms: algorithms that implement all or any of the above definitions and also use any other complex method for matching maps like the use of Kalman filters, Dempster-Shafter's mathematical theory, fuzzy logic model, etc.

Map-matching algorithms can also be categorised into offline and online processes [YW04].

Offline algorithms have no time restraint to process points and generates a complete route. Online algorithms are efficient processes that have incomplete information (offline algorithms may access points in the future during the snapping process) and have to process matching in real-time. Online algorithms are used on PNA to snap user position during routing and free navigation.

Yin and Wolfson 2004 [YW04], created an algorithm that uses Disjktra's shortest path to select the smallest weight path. Weight is calculated from segment distance to points and the lowest total is the traveled path.

Li et al 2005 [LLZ05] defined an offline topological map-matching algorithm that uses buffer bands with the format of an ellipse around processing points. The ellipse format depends on the direction of the user and segment length. This topological process uses heading, length and closeness to snap user points.

Zhou 2004 [Zho04] includes an overview of some base map-matching processes:

- Point-to-Point: this process is the most simple one and it is also the one that produces the worst results [sYpKsC05]; points are snapped to the closest segment shape point;

- Point-to-Curve: this method uses distance from point to the segment curve to calculate the best segment;

- Curve-to-Curve: since GPS points compose a complete route it is possible to compare a path (curve) to a segment in order to calculate the best match.

The previous processes have also been improved by the use of other algorithms that increased the accuracy of the results [WBK00].

Another offline map-matching algorithm was presented in [MHA04] that features a fast process that relies only on position coordinates and on the network topology. This process selects N near candidates to every point and selects the best path. Although this is a fast process, it may produce routes including segments that the user did not use in order to satisfy the creation of a route.

Greenfeld 2002 [Gre02] also implemented an offline map-matching algorithm based on point to curve matching that uses topological information, distance and direction to calculate the best match for each point.

Quddus et al 2006 [QON06] developed a way of evaluating integrity of map matching algorithms based on various error sources associated with position fixes and map data using fuzzy logic. The proposed method may be applied with any type of matching method.

2.4 Technology

The following sections present the possible technology options selected during the development of this project.

2.4.1 Spatial Database

InfoPortugal stores the digital map of Portugal in shape files that are converted to a Postgresql Database with PostGIS spatial extension.

Postgresql is an open source object-relation database system in active development and frequent releases. It is a mature project with more than 15 years that focuses in being a standards complier and operates on all major operating systems [pos09b].

Postgis is an Open Source spatial extension to the Postgresql that adds support for geographic objects. It follows the ”Simple Features Specifications for SQL” by OpenGIS and has been certified as compliant with the ”Types and Functions” profile [pos09a].

Since the existing TeleAtlas database is in Postgresql, it was a requirement to continue using it to archieve the traced data and detections.

2.4.1.1 Viewer

uDig is an open source application being actively developed in Java with the goal to provide users a complete Java solution for GIS data access [udi09].

uDig provides users with editing and visualisation of geographical objects with an intuitive and simple interface.

This project benefits from this application by using it to visualise the output generated by the system and the digital map.

Another tool used for the visualisation of GIS data is Google Earth.

Google Earth is an application that allows exploring the world by the use of aerial imagery. This application also allows adding overlays with geographic objects. This tool was used for the visualisation of the results.

2.4.1.2 Programming Language

C++ and Python were the possible languages for this project. Python, although an easy programming language, does not offer the speed and all the possibilities of C++. Python also restricts the way the user programs the code by implementing a forced indentation.

For this project the selection pended to C++ for the complete and easy to use existing API for access to the Postgresl database.

2.5 Summary

Personal Navigation Assistants that rely on a GPS device for location information are of common use nowadays. PNA devices depend on the accuracy of GPS devices and digital maps to provide the user with good information. Work has been done in order to solve some of the problems caused by the accuracy errors in GPS or the lack of complete information in digital maps.

Digital maps have to be updated frequently to provide PNAs with reliable and accurate data. Different forms of maps updates exist that allow reduction of costs, like the use of aerial imagery, processing of GPS data sets and user feedback. To extend existing digital maps there are developments that allow detection of lane information from GPS data. Correction of spatial alignment of roads is also possible by comparing GPS positions with the currently defined segments.

The usual need of map matching GPS positions to a location in a digital map had a large improvement over some of the first algorithms. Existing algorithms in the field can be classified in several groups depending on degree of knowledge (known route, route follower, no knowledge), complexity (geometric, topological, probabilistic, advanced) and time needs (offline, online).

Technology selection depended on existing constraints. Postgresql with the PostGIS spatial extension is a standards compliant database with an open source GIS extension that follows some of the specifications by OpenGIS. For visualisation of detected changes it is used uDig, an open source spatial editor with a simple interface.

Chapter 3

Methodology

This chapter introduces the proposed methodology that includes the complete process from the collection of points to the visualisation phase.

Each created algorithm and its internal processes are presented individually.

3.1 System Flow

The proposed methodology is composed of three main parts:

- Data collection: acquires tracking data from existing users' PNAs;
- Application Processing: crunching of data to reveal possible changes to road network;
- Visualisation: present the possible changes so it is possible to take actions.

Figure 3.1 exposes the different phases of the methodology.

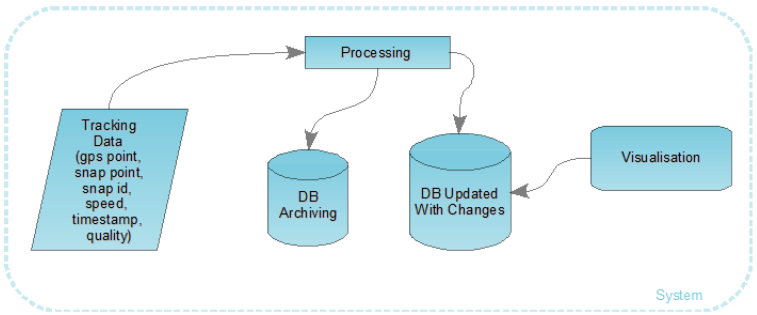


Figure 3.1: System Flow

3.1.1 Data Collection

By using user GPS positions it is possible to lower costs with the detection of possible alterations to the road network. User GPS coordinates are recorded by the PNA the user utilises and are uploaded to the main server when the user connects his device to the Internet.

3.1.2 Application Processing

The collected data is then processed and saved by the developed applications that integrate the created algorithms and generates database tables with the results.

In order to ease the visualisation it should be possible to convert the output data format to the one used by map companies, like Shapefiles.

3.1.3 Visualisation

For the visualisation of altered geometries the company may use a GIS editor (i.e. uDig) and shape files obtained from the database. After visualisation, the system user decides the modifications the digital map needs and then may rerun the application.

3.2 Database

A database to hold the acquired and generated data is used. Figure 3.2 presents the proposed database.

Tables NW (network), JC (junctions), MN (manoeuvre) already exists on a TeleAtlas database. The other tables are the ones needed by the developed algorithms.

Each proposed method uses some of the tables:

- New roads: uses tables NewRoads and NewRoadsUGeom. NewRoads contains records of the detected new roads from statistical information whereas NewRoadsUGeom contains only user's geometry for a new road;
- Roundabouts: table RoundaboutsVotes contains the records of voting for each junction; RoundaboutsUGeom is only user detected possible roundabouts and RoudaboutsGeom contains the final geometry;
- Wrong road geometry: geometry uses tables GeomChangeUser for user's geometry of a possible changed segment and table GeomChange for a probable geometry change;
- Wrong Direction: table WayUsage is the counter of direction use for each segment; Table WrongWayLine and WrongWayPoint contains the path of an user in possible wrong direction; WrongWay are the final detected wrong directions;

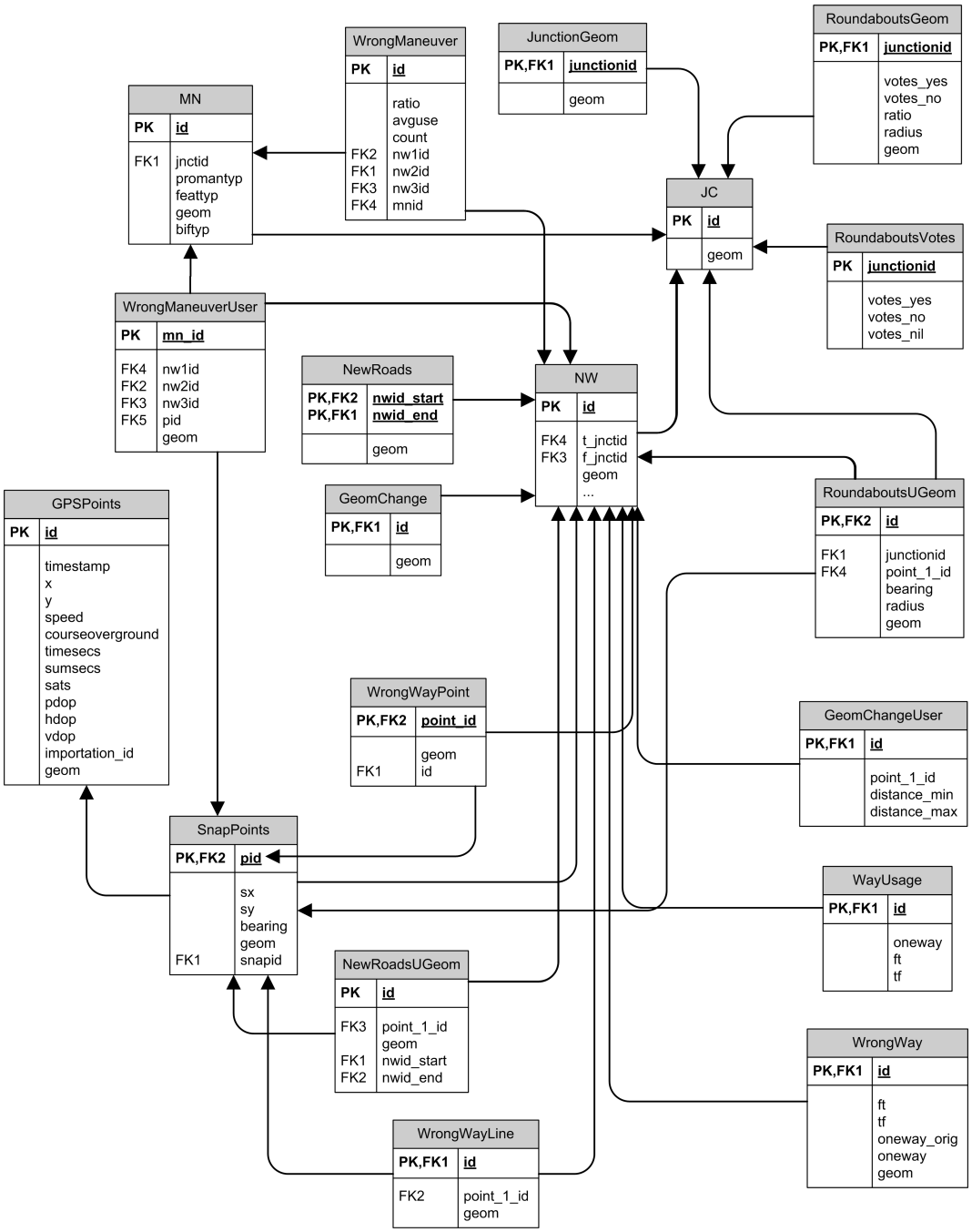


Figure 3.2: Database Relations Diagram

- Wrong Manoeuvre: WrongManoeuvreUser are the user's path that compose a possible wrong manoeuvre and WrongManoeuvre contains final detected wrong manoeuvres;
- Junction Position: uses JunctionGeom to save the new geometry.

This database can be implemented on a database manager with GIS extensions like Postgresql with PostGIS. Geom field on each table is the spatial attribute.

3.3 Map-Matching

Map-Matching is an important problem in navigation technology that consists in the correct determination of the right road segment, and position within, for each point. The process of matching GPS points to a digital map may also be called snapping.

After applying snapping, it is possible to obtain the complete user's route with aggregated topological information.

Since coordinates from GPS contain a variable error at all times, it is hard to select the correct segment. The simple approach, of selecting the closest segment to the point, does not work [Zho04].

Figure 3.3: Road segments and GPS points

Figure 3.3 shows gathered GPS points. It can be seen that the algorithm that snaps points to the closest segment would not yield the best results.

For this project it was required that the algorithm would snap points to a location even if the specified direction was prohibited in order to detect changes in direction or authorised manoeuvres.

Figure 3.4 shows the correct snapping process applied to points. This is an actual result of the developed algorithm.

Figure 3.4: Map-Matching - Problem

3.3.1 Algorithm

A possible approach to produce correct snaps is the use of an adaptive algorithm that tries to correct itself. The presented algorithm uses a method of segment retrying, when it is impossible to continue matching.

A set of points is being snapped continuously and the segment that is selected is always the best candidate. Although the best candidate is the most probable, it is also possible that it is not the right one. After selecting a wrong segment and verifying that it is impossible to continue snapping through the selected path, the algorithm will try to go to the last junction and will decrease weight of the wrong segment and will try the next best segment.

If the algorithm is not capable of producing a correct result it will stop and snap the points to the GPS position. This is important to allow execution of some of the detections.

Figure 3.5 is a diagram of the proposed algortihm. Before snapping a point to a segment it is necessary to select possible candidates by querying the existing map for existing segments within a radius centred on the point (2). Although the system snaps points even if in the wrong traffic direction, in order to reduce map-matching errors, when a two way road is composed by two one way segments the segment selected is the one with the right direction.

Candidates are then evaluated by their weight value which is calculated using equation 3.1 (4). When a point has no available candidates it is snapped to the original GPS position and the process is restarted (19). Every point that is processed is added to a list of points (6) and every change of segment on a junction is added to a list of alternatives (16). Point's candidates are then reordered (9) according to dynamic weight values that depend on the last snapped segment and position, as defined in section 3.3.1.2.

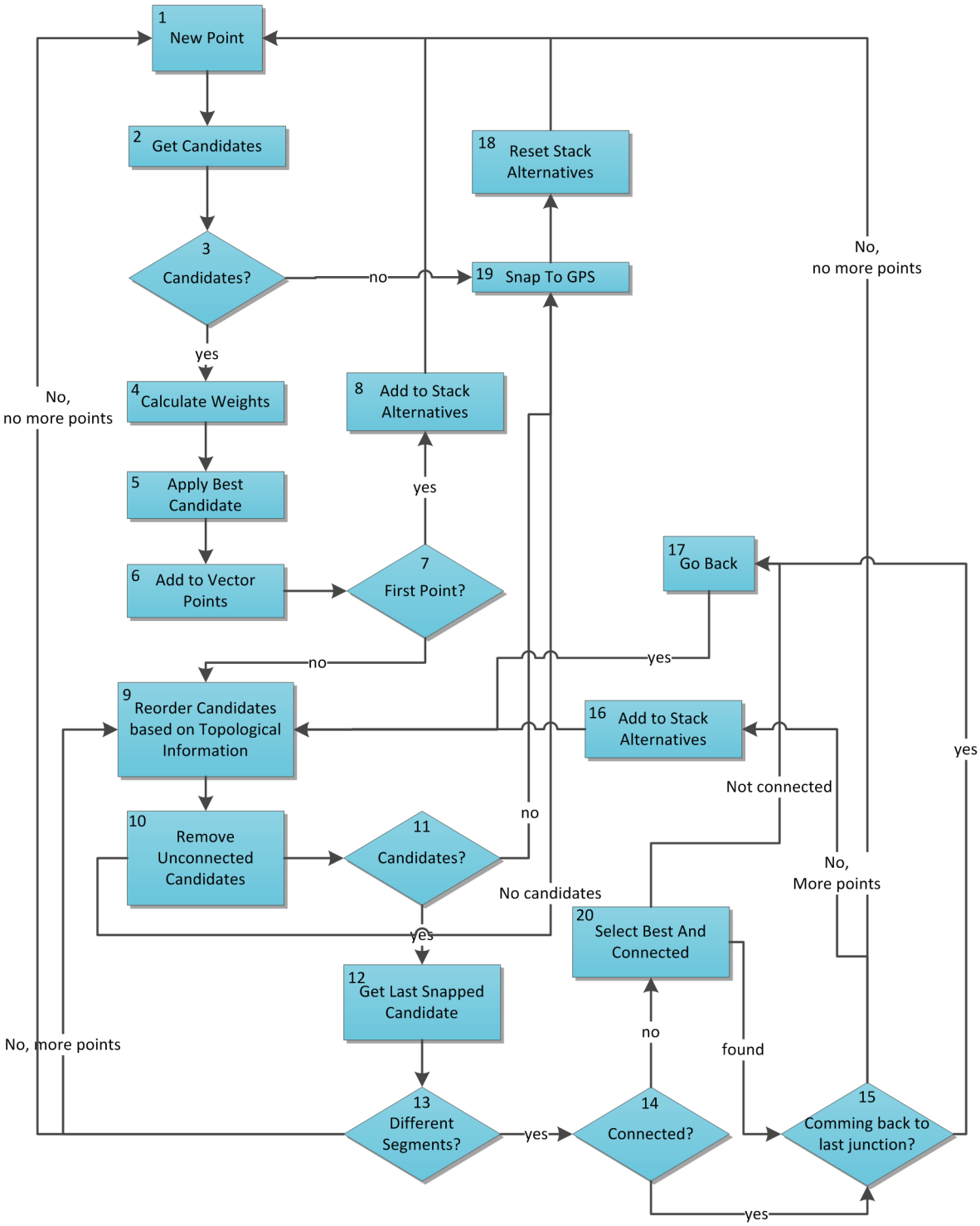


Figure 3.5: Map-Matching - Algorithm

In order to improve performance, impossible segments (ones that are not reachable from the list of segments for the previous point) are removed from candidate's list (10).

After selecting the best candidate it is important to verify that they are connected (14). If segments are not connected then it is necessary to select a lower weight connected segment from the list (20). When there is no segment that connects to the current previous segment, it is required to go back (17) to the last junction and alter the selected segment.

The process of going back is explained in detail on the next diagram 3.6.

After detecting a wrong selected segment, the algorithm "go back" 3.6 is performed with information of the previous alternatives, points and candidates. The algorithm starts by checking if there was improvement since the last call by checking if the current point's id is greater than last id (2) and, if it is, it resets the backtrack counter (3). The algorithm picks the last processed junction and increments the number of "go back" executions of the candidate (4). If the limit of passes is reached, the candidate is removed (7). The maximum and minimum candidates' weight are calculated on the point referenced by the last processed junction (9). In order to reduce the possibility of returning to the same candidate on next call of "go back", the affected candidate has its weight decreased by a defined percentage (10).

If there were no more candidates on the list for the referenced point, or the number of "go back" calls reached a limit without improvement (by the use of backtrack counter), or the best candidate snap position distance to the GPS point is greater than a maximum allowed value, or the calculated maximum weight is lower than the minimum weight value plus a percentage of the difference of maximum and minimum weights (11), then the candidate is removed from all of the points after the junction (12), and points are snapped to GPS position.

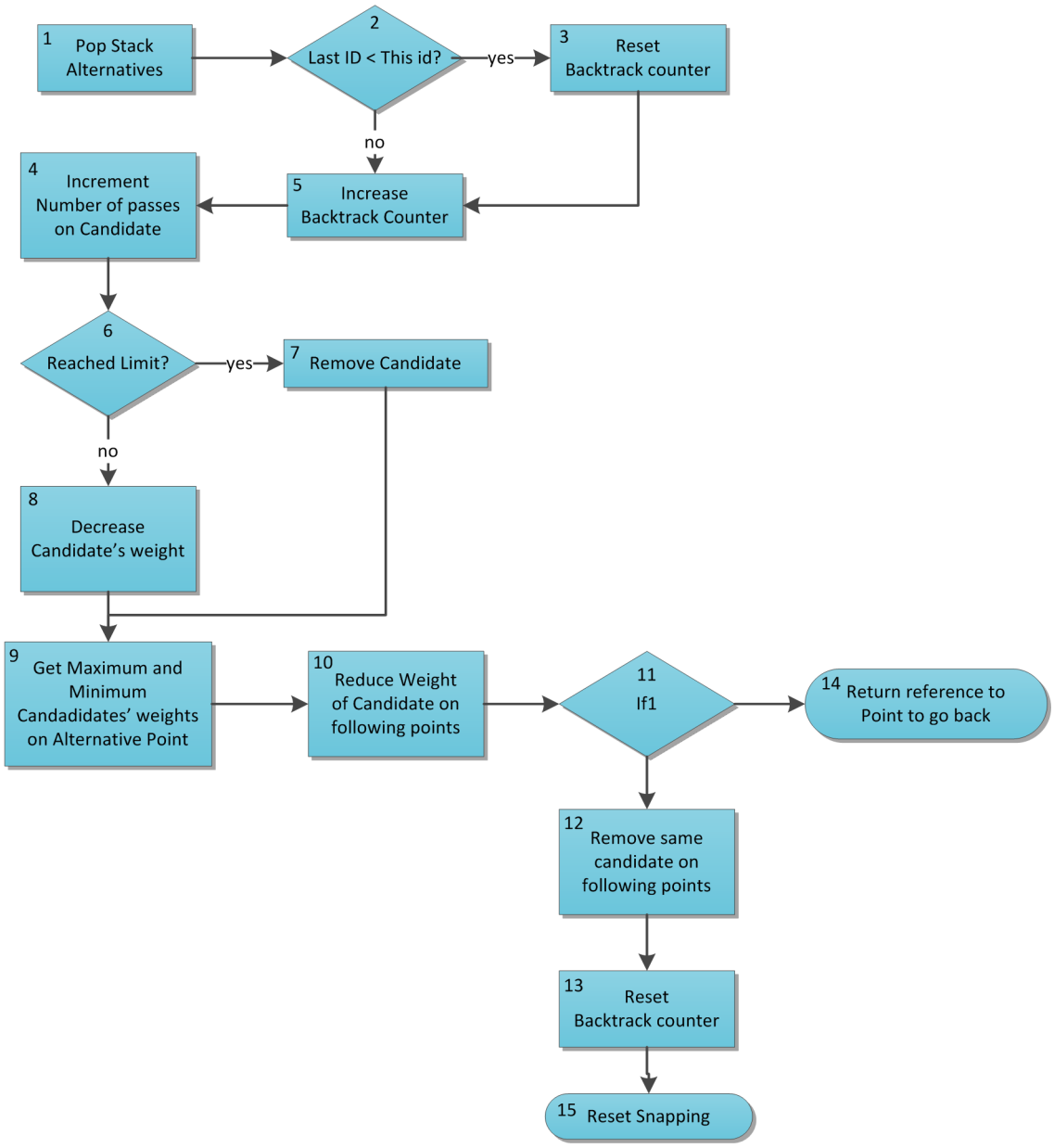
If it is possible to continue snapping, then a reference to the point from where to proceed is returned to the map-matching algorithm (14).

3.3.1.1 Weight Calculation

Candidate weight value for each point is calculated by comparison of azimuth (section 3.3.1.3) of movement and the azimuth of the possible snap location.

$$Weight_{C_n,P} = W_{Distance}.(1 - Distance/MaxDistance) + W_{Azimuth}.(\cos(Az_P - Az_{C_n}))^{W_{PAzimuth}} \tag{3.1}$$

$Weight_{C_n,P}$ = Base Weight of Candidate Segment n for P
$W_{Distance}$ = Constant Weight of Distance
$Distance$ = Distance from P to projected location in Candidate Segment
$MaxDistance$ = Constant Maximum Distance



If1: No Candidates ||
Backtrack counter = Maximum ||
Best Candidate Distance > Maximum ||
MaxWeight < MinWeight + DiffWeight * limitProportion

Figure 3.6: Map-Matching - Algorithm - Go back

$W_{Azimuth}$ = Constant Weight Azimuth
Az_P = Azimuth of Point
Az_{C_n} = Azimuth of Projected Point on Candidate Segment
$W_{PAzimuth}$ = Constant Weight Azimuth Cosine Power

For best results, if azimuth value obtained from GPS is not trustworthy, like in situations of very low speed, different weights should be applied.

This base weight will be later modified during weight comparison.

3.3.1.2 Comparison of Weights

During comparison of candidates some modifiers are applied to weights that are based on topological information. Like in situations where it is impossible to change from segment A to B due to a long distance to junction or in case there is no connection between them.

Before being able to calculate the comparable weight it is necessary to determine other variables.

$$C_nOpVal = \cos(\Delta Azimuth_{C_{n-1},C_n} - \Delta Azimuth_{P_{n-1},P_n})^{W_{PCompareDirection}} \tag{3.2}$$

$\Delta Azimuth_{P_{n-1},P_n}$ = Difference in GPS Azimuth
$\Delta Azimuth_{C_{n-1},C_n}$ = Difference in Segment Azimuth
C_nOpVal = Comparison of Direction between GPS points and Network Segment Points
W_{OpVal} = Constant Weight Value for OpVal modifier

$$ReachableDistance_{SP} = Speed + CRDistance + Error \tag{3.3}$$

$Speed$ = Movement Speed
$CRDistance$ = Constant Added Reachable Distance
$Error$ = Error calculated from last snapped point

$$C_nReachDist = 1 - (DistanceJunction / ReachableDistance_{SP}) \tag{3.4}$$

$C_nReachDist$ = Reachable Distance Modifier, equal to 1 when comparing same segment
$DistanceJunction$ = Distance from Point to Junction of (C_{n-1}, C_n)
$ReachableDistance_{SP}$ = Maximum Reachable Distance from Point

Now the actual calculation of the Comparable Weight:

$$ComparableWeight = (1 + W_{OpVal}.C_nOpVal).(C_nReachDist.C_nWeight) \quad (3.5)$$

W_{OpVal} = Weight of Difference of Direction
C_nOpVal = Candidate Segment Difference of Direction
$C_nReachDist$ = Candidate Segment Reachable Distance
$C_nWeight$ = Candidate Segment base Weight (see 3.3.1.1)

3.3.1.3 Azimuth Calculation

Calculation of azimuth (direction of movement) between two points is done based on [Gre02] equation.

$$Az_{P_1,P_2} = \arctan(\frac{x_2 - x_1}{y_2 - y_1}) \quad (3.6)$$

if $(y_2 - y_1 > 0)$ add π
if $(Az_{p^1,p^2} < 0)$ add 2π

This azimuth is a value from 0^o to 360^o where north is indicated by a 0^o and south with 180^o.

3.4 Detection of New Roads

New roads are constructed frequently and usually alter possible routes near them. By using user positions along these new streets it is possible to obtain an approximate geometry and information about the existence of new ones.

After detection of a new road, it is required that a base geometry is generated.

Although this detection will only obtain geometry information (without toponymy data), it will allow map companies to pinpoint locations in need of an update.

3.4.1 Algorithm

This algorithm is composed of two phases:

- Phase One - collection of possible new roads by detecting jumps from unconnected segments and obeying to certain restrictions;

- Phase Two - in this phase information generated from phase one is analysed and determination of the new road geometry is made by use of average positions (section 3.10) along the track.

3.4.1.1 Phase One

Detection of segment jumps is made by detecting an increasing distance from last snapped segment without a change to the next connected segment or a jump from a segment without a connecting junction.

Part of this job is done by the snapping module that snaps the point to GPS if it there is no better match.

Figure 3.7 is the diagram representation of the algorithm. This algorithm processes each snapped point until there is a snap to GPS (1); this defines the start segment id (2). Try to go back and detect the last correctly snapped point and add all points to the new road. Continue processing until points are snapped to a segment again (3); this defines the end segment id (4) and save to database the possible new road (5).

3.4.1.2 Phase Two

Generation of the final geometry for a new road is made by spatial averaging all user made tracks against one trace.

To reduce possibility of an incorrect selection of the first segment it is suggested to test distance from the selected possible track to other tracks.

This algorithm (figure 3.8) starts by getting possible new roads (1), with a high number of travels, and user geometries . Then, picks a reference geometry (4) and calculates a spatial average (section 3.10) for each point of the geometry (5). To better generate geometry it is suggested to select a stepping distance independent of the number of points.

Create a new line that contains all the averages (6) and apply a simplification algorithm like Douglas Pecker (7). Save the new line to databaes (8).

Results can be seen on section 4.2.2.

3.5 Detection of Roundabouts

Construction of roundabouts is a common practice to reduce traffic issues. New roundabouts ought to be detected to reduce user confusion during navigation.

A car during a roundabout deviates from junction and then corrects its position. By comparing user direction against segment direction it is possible to detect an existing and unmarked roundabout.

Figure 3.9 is an example of a nonexistent roundabout on the digital map. Blue lines represent the digital map and the red dots are the acquired GPS positions.

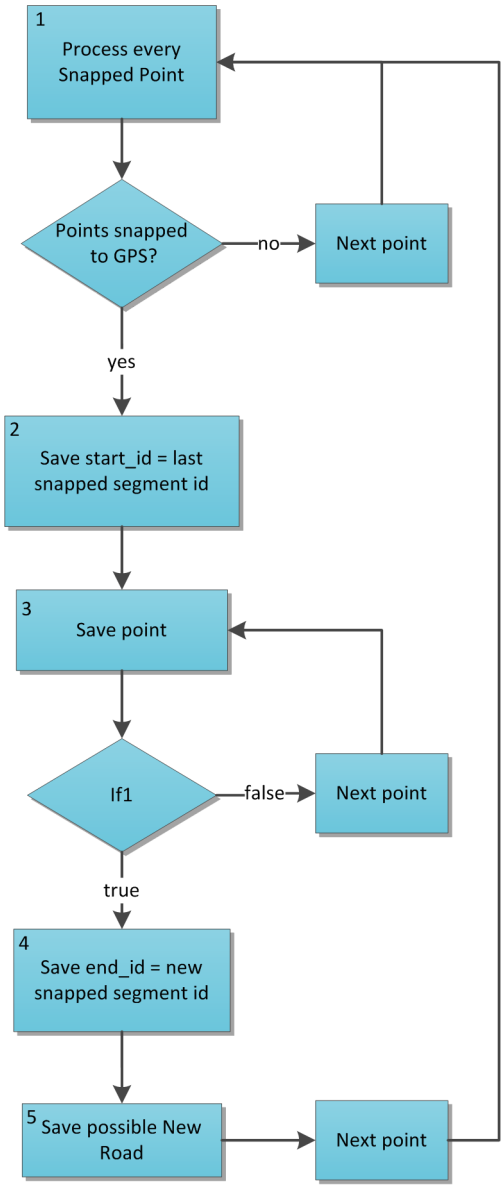


Figure 3.7: Detection of New Roads - Phase One diagram

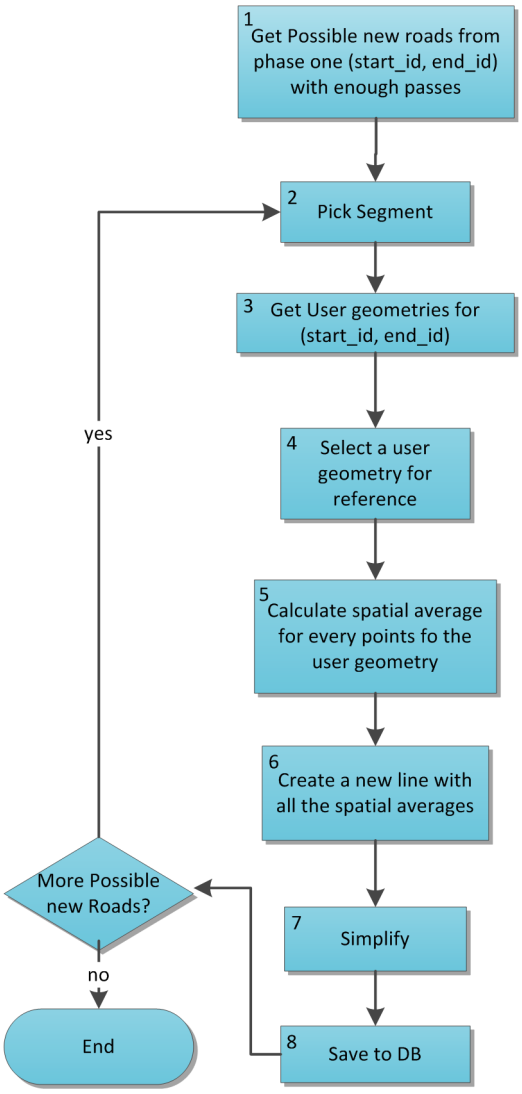


Figure 3.8: Detection of New Roads - Phase Two diagram

Figure 3.9: Detection of Roundabouts - Problem

3.5.1 Algorithm

This detection depends on two phases:

- Detection of possible user traces that might represent a roundabout by comparison of segment curves with user direction. Calculation of a possible radius;
- Selection of true roundabouts and averaging of roundabout radius and saving of possible roundabouts to database.

3.5.1.1 Phase One

Constitutes detection of possible user tracks that differ from snapped segments. By comparing user direction to snapped segment direction it is possible to detect a path on a roundabout.

Right turns ought to be ignored since it is difficult to detect differences between a turn to the right and the first exit to the right of the roundabout.

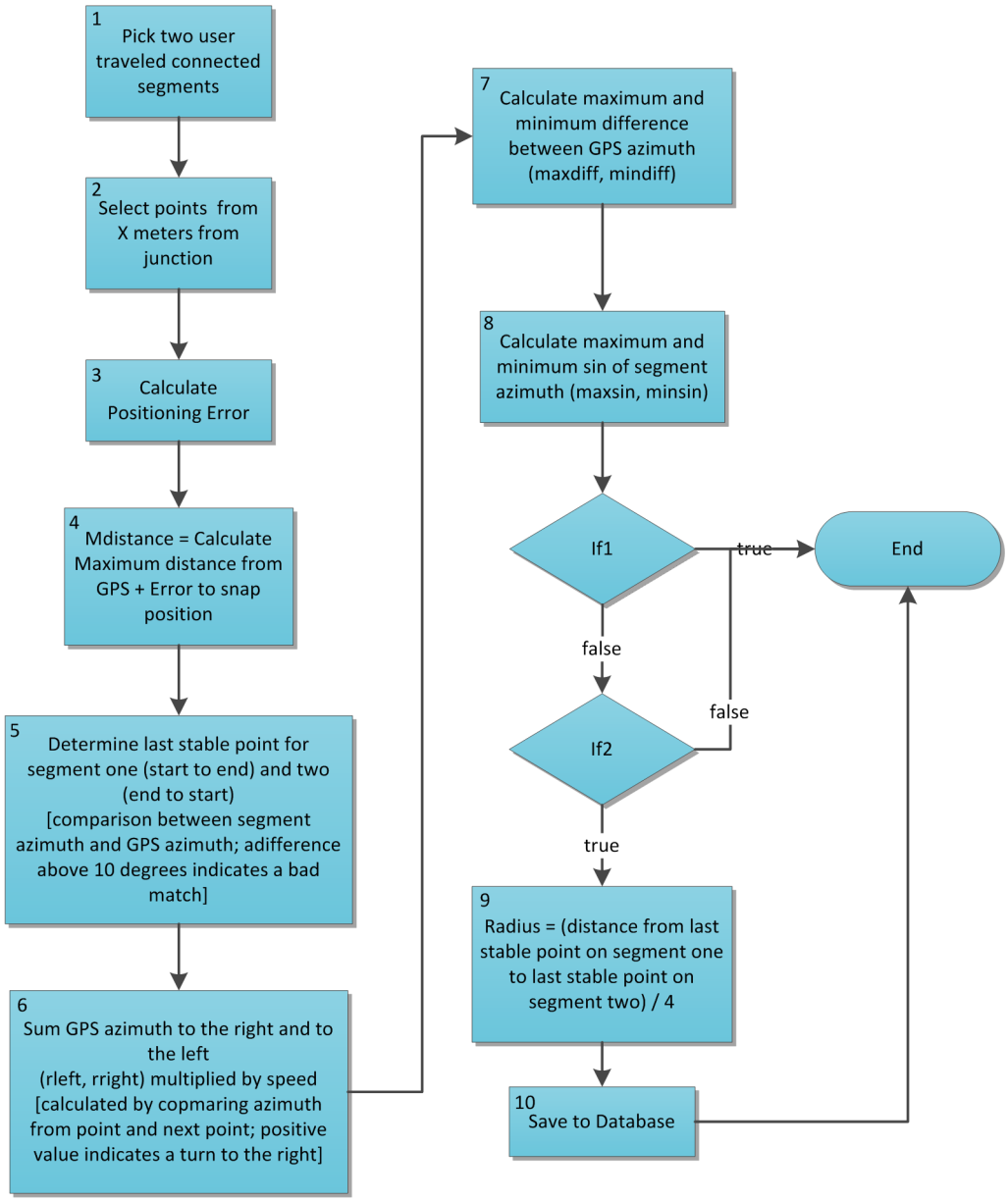
In figures 3.10 and 3.11 it is possible to verify that there is no visible difference on the traveled path (black line) when on a roundabout or on a normal junction. Blue represents analysed segments and yellow is the referenced angle. Gps points are pinned red.

This algorithm consists in the analysis of every junction to check for a possible roundabout. Every pass on a junction accounts as a vote (yes/no/ignored).

Figure 3.12 represents the algorithm of roundabout detection. The algorithm starts by the selection of two user traveled connected segments and processing X meters of each in the direction of the junction (1). Calculation of positioning error by subtracting GPS position to the snapped position (3). For each segment, detect the last good match of azimuth

Figure 3.10: Detection of Roundabouts - Algorithm - Turn to the Right 1

Figure 3.11: Detection of Roundabouts - Algorithm - Turn to the Right 2



If1: Compare last GPS azimuth on segment one with last GPS azimuth on segment two, Turn to the right?

If2: MaxDistance – MinDistance > Minimum Difference Distance &&
MaxDistance > Minimum Maximum Distance &&
ABS(maxsin – minsin) > Minimum Difference Sin &&
MaxDiff > Minimum Maximum Azimuth Difference &&
ABS(rleft / rright) < Maximum Proportion turn left / turn right

Figure 3.12: Detection of Roundabouts - Phase One diagram

between points and snapped points (5). A bad match is indicated by an angle difference above 10^{o}. Sum GPS azimuth to the right and to the left multiplied by speed (6). This will be used to compare if the user went more to the right or to the left. Calculate maximum and minimum azimuth difference between points (7). This indicates sharp turns. Determine the maximum and minimum sin values of segment azimuth (8). Compare the last GPS azimuth on segment one with last GPS azimuth on segment two and if it is positive and below a defined value (turn to the right) then cancel this roundabout and vote ignored (If1).

Now the comparison that determines the existence of a possible roundabout is composed of a difference in distance from GPS position to snap position higher than a specified value, a maximum distance from GPS position above a defined value, module of difference from maximum and minimum detected sin above a determined value, maximum azimuth difference above specified and module of ratio turns below a value (If2).

If all the above was true then the radius of the possible roundabout should be calculated by the use of last good match of segment one and two divided by 4 (9), and the information saved to the database (10).

3.5.1.2 Phase Two

In this phase occurs a selection of the most probable segments by comparing positive votes with negative ones and the generation of a representing geometry.

Only junctions with a minimum number of votes should be processed to guarantee a minimum truth.

As presented on figure 3.13, detection of roundabouts starts with the selection of possible roundabouts with a high number of votes (yes, no) from phase one (1) and calculate ratio yes/no (3). If above a specified value then it proceeds with the calculation of the average radius for the roundabout and generation of the geometry (5).

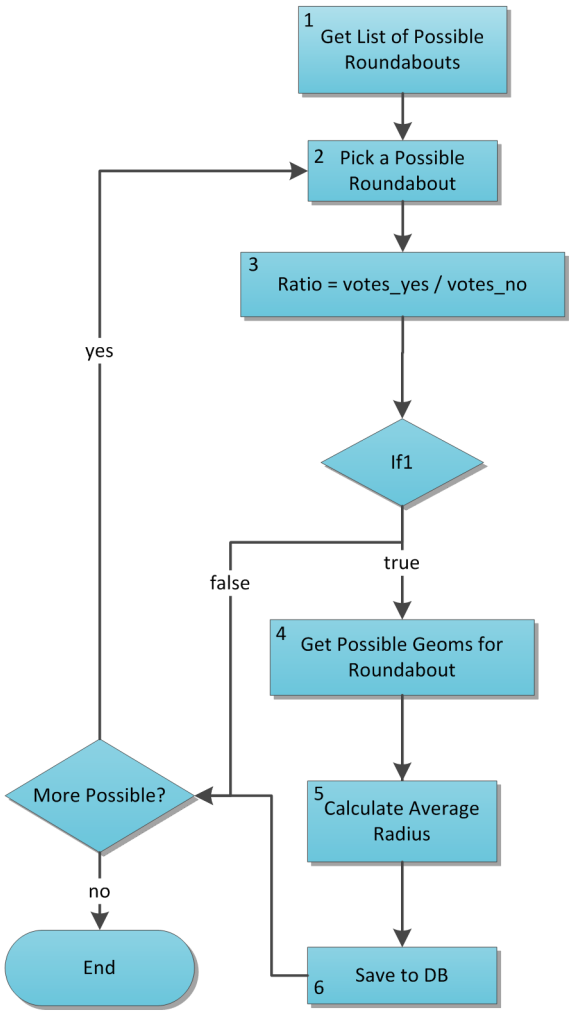
Results can be seen on section 4.2.3.

3.6 Detection of Wrong Road Geometry

Geometry of a street usually changes with the construction of new streets and alteration of junction position.

Another reason for a wrong geometry in the digital map is an incorrect digitisation due to inaccurate data or for performance reasons since a segment with less points occupies fewer resources.

Incorrect geometry may lead to errors during map-matching which might cause inaccurate information to the user or even wrong positioning on the PNA.



If1: Ratio > Min_ratio && (votes_yes + votes_no) > Min_votes

Figure 3.13: Detection of Roundabouts - Phase Two diagram

Figure 3.14: Detection of Wrong Road Geometry - Problem

Figure 3.14 showing a digital map drawn with blue and GPS points, in red, indicates possible wrong geometry for some segments, since GPS positions deviates from street directions.

3.6.1 Algorithm

This algorithm is composed of two phases:

- Phase One: processing of user points and detection of possible geometry changes;
- Phase Two: selection of the most probable street changes by comparison of user travels for a segment and existing geometry.

3.6.1.1 Phase One

Figure 3.15 is the diagram of the algorithm for the detection of wrong road geometry.

Distance from the GPS points to the center of the segment (snap position) is calculated to verify if there is reason to process the segment during the second phase (2).

When the maximum allowed distance is reached then the segment is saved to be processed during phase two (3).

3.6.1.2 Phase Two

Figure 3.16 explains that all possibly changed segments are selected from phase one to be processed (1). For every segment point being analysed it is constructed a circular buffer

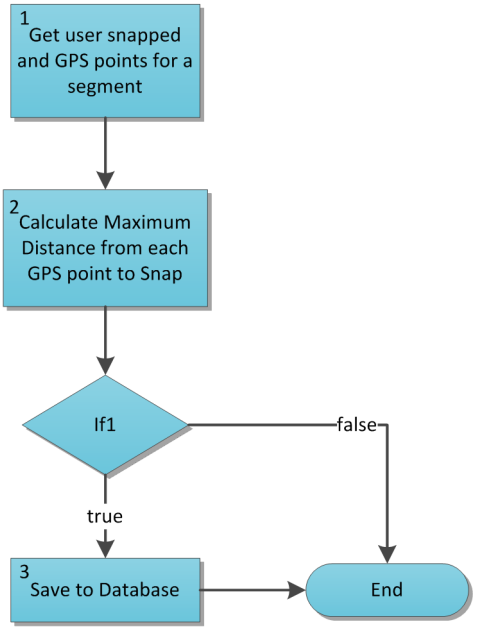


Figure 3.15: Detection of Wrong Road Geometry - Phase One Diagram

around it and all user points are averaged (3). Users' Points snapped to the segment are averaged (check section 3.10) along the segment with a defined radius and a new geometry is computed (4). The new geometry is then simplified by the algorithm Douglas Pecker to reduce number of points (5).

This detection will only return good results when there is a large quantity of user travels.

Results can be seen on section 4.2.4.

3.7 Detection of Wrong Direction

Alteration in allowed street direction is a common change that causes problems with routing since it impossibilitates the user from traveling to a required junction.

To overcome this problem routing should be able to try different routes that do not pass on the problematic street. Some PNAs allow the user to mark the road as blocked and will recalculate a route without the problematic segment.

There are four possibilities for the direction of a street as specified in [tel08] for the field *ONEWAY* of table *NW*:

- One Way TF (TF) - Street is only one way from *junction To* (*T_JNCTID*, table NW) to *junction From* (field *T_JNCTID*, table NW);

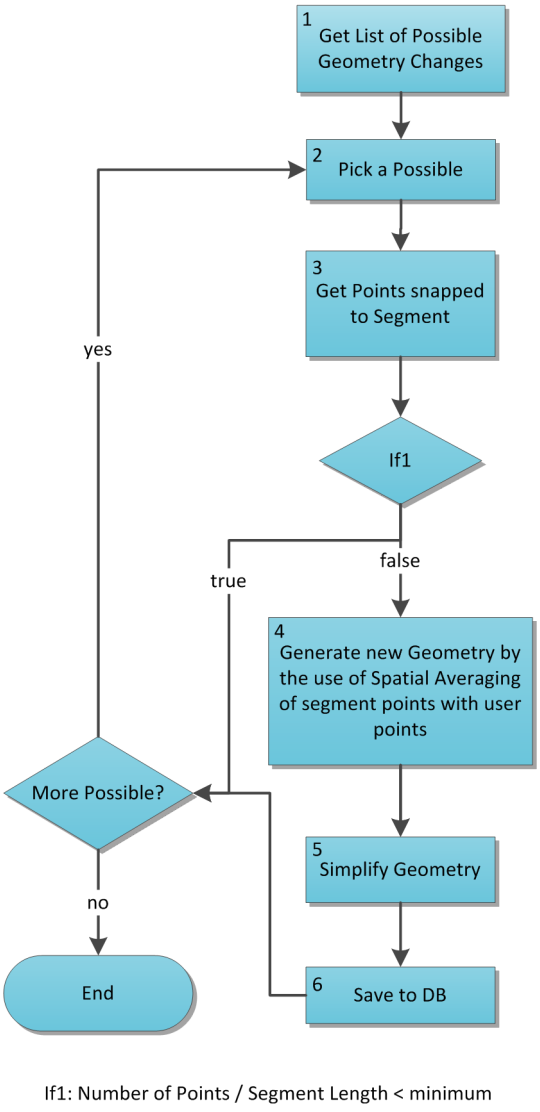


Figure 3.16: Detection of Wrong Road Geometry - Phase Two Diagram

- One Way FT (FT) - Street is only one way from *junction From* (F_JNCTID, table NW) to *junction To* (field T_JNCTID, table NW);
- Both (blank/NULL) - Street may be travelled in both ways, this is the most common;
- None (N) - Street has no traffic.

The detection of wrong direction should be made by comparison of traffic in each way.

This detection is completely based on the correct snapping of user points. Errors in the map-matching module will alter results.

3.7.1 Algorithm

This detection is made in two phases:

- Phase one: travel direction is saved to a counter (FT or TF) that will be compared later with the authorised direction; every possible wrong way is saved as a *linestring*;
- Phase two: comparison of counter values for each direction and the result is compared with the authorised direction.

3.7.1.1 Phase One

By selecting three segments (previous, current and next) each time (figure 3.17) (1), snapping errors are reduced. The "current" segment is the one that will be processed (2). The three selected segments must be connected, if they are not then "current" segment is skipped.

Selection of the current direction is made by comparison of junctions' id of current segments with fields F_JNCTID and T_JNCTID of table *NW* (2).

After calculation of direction, the respective counter is increased (3).

If the direction of the user counters the specified on the digital map then a line composed of snapped coordinates and all the GPS points are saved to database for visual inspection if necessary (4).

3.7.1.2 Phase Two

Phase two consists in the comparison of direction counters and authorised directions.

The phase two diagram is shown in Figure 3.18. Segments that do not have a minimum usage should not be processed (1). To accept a direction as a possible way, a percentage limit for the counter should be specified (3).

$$ratio_{ft} = (ft/(ft+tf)) * 100 \tag{3.7}$$

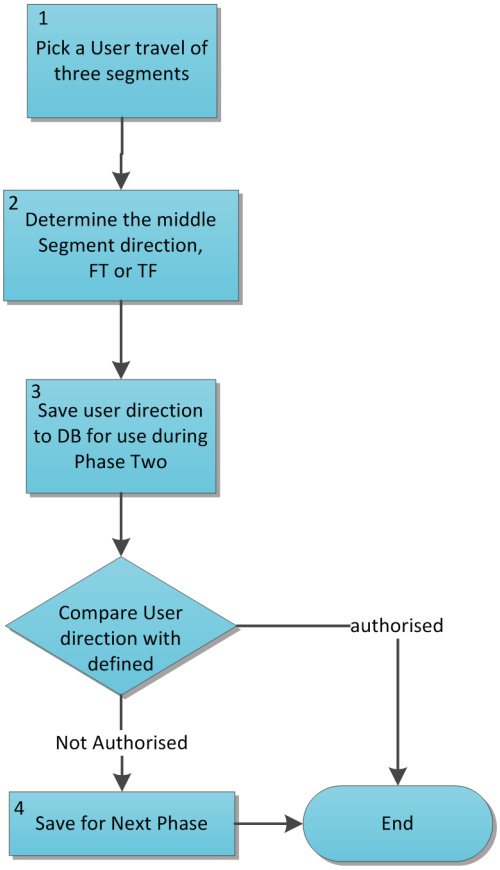


Figure 3.17: Detection of Wrong direction - Phase One diagram

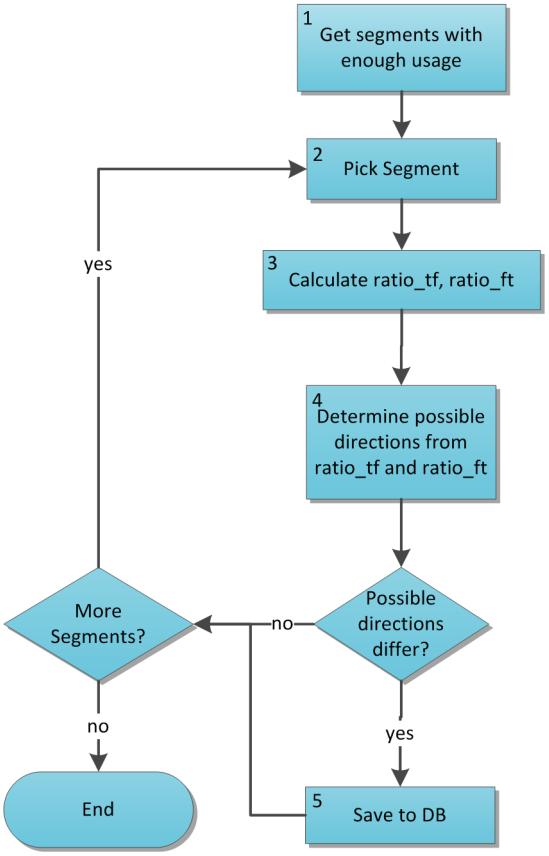


Figure 3.18: Detection of Wrong direction - Phase Two diagram

$$ratio_{tf} = (tf/(ft+tf)) * 100 \tag{3.8}$$

The previous equations (3.7 and 3.8) defines the calculation of ratios for each direction (used in (4)). If the calculated ratio for a direction is lower than a specified percentage then it is assumed that there is no traffic in that direction. The result is then compared to the existing value and, if different, it is saved to database (5).

Results can be seen on section 4.2.5.

3.8 Detection of Wrong Manoeuvre

Traffic signs that limit manoeuvres are often changed to re-route traffic. Wrong manoeuvres, like wrong directions (section 3.7), change the behaviour of a routing application and may lead to failure of a route.

Detection of wrong manoeuvres is done with a similar approach to wrong directions.

TeleAtlas database [tel08] has a table *MN* that defines manoeuvres and relates a junction and a path (order of traveled segments).

To check for a possible manoeuvre restriction it is necessary to query *MN* with a junction id and check fields *FEATTYP* for restricted manoeuvre. If there are results then query table *MP* with *MN_id* and check for the sequence of segments.

3.8.1 Algorithm

This algorithm is composed of two phases:

- Phase One: Detection of possible wrong manoeuvres by comparison of user travel with manoeuver paths;
- Phase Two: Selection of wrong manoeuvres from list of possible manoeuvres by comparison of usage counter for each segment and detected counter.

3.8.1.1 Phase One

Phase one (figure 3.19) processes three segments of a user travel at a time (s1, s2, s3) (1). It is necessary to get id of junction (s1,s2) to query table MN by junction id (2).

For every result with *FEATTYP* defined for a restricted manoeuvre, check table MP for a sequence equal to the user path (s1,s2) or (s1,s2,s3) (3). If there is a match, a line with the snapped points is saved to the database for phase two (4).

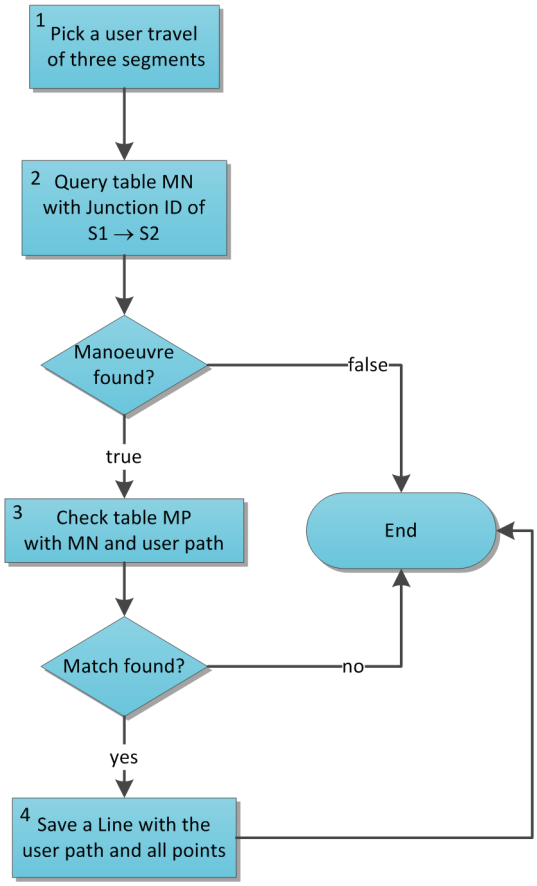
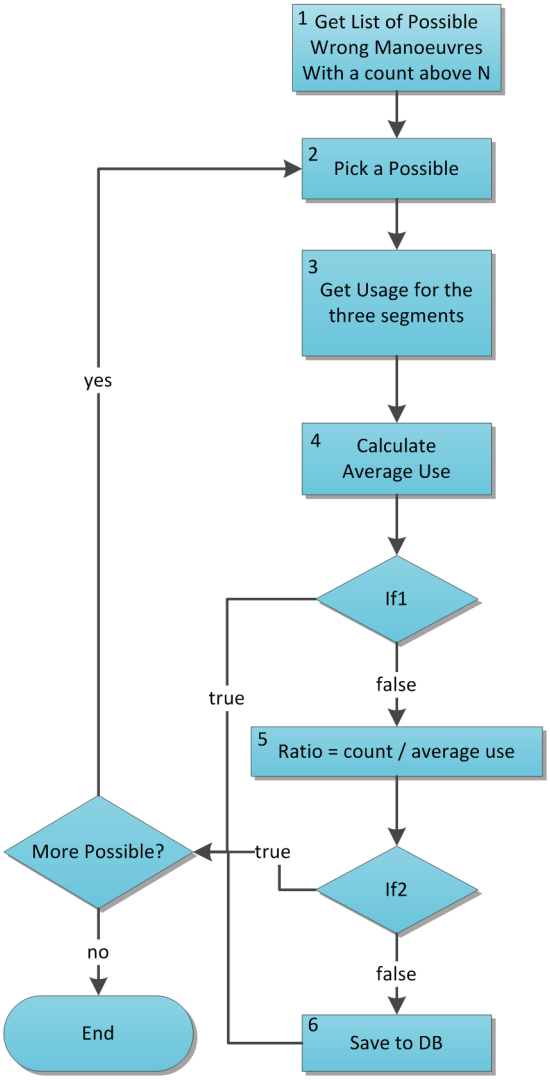


Figure 3.19: Detection of Wrong Manoeuvre - Phase One



If1: Average use Below minimum

If2: Ratio Below minimum

Figure 3.20: Detection of Wrong Manoeuvre - Phase Two

3.8.1.2 Phase Two

Phase two (figure 3.20) is based on the usage of adjacent segments and number of detected possible wrong manoeuvres. The algorithm starts by getting the possible detected manoeuvres with the count above N to be processed (1). Usage for the three segments that are being processed is obtained (3) and average use is calculated (4).

$$ratio = count/average \tag{3.9}$$

Average uses of segments is calculated and ratio (equation 3.9) is compared to a given value (If2).

If ratio is below a given value and average use is above a minimum count then a wrong manoeuvre is assumed as detected (6).

Results can be seen on section 4.2.6.

3.9 Junction Position Correction

Due to error during the digitisation process, it is possible that junctions are not positioned accurately.

Adapting the location of a junction improves PNA/user interaction due to a corrected positioning on screen and during routing.

Figure 3.21: Junction Position Correction - Problem

Figure 3.21 represents a possible misplaced junction. GPS points (red) do not pass on the junction so it is slightly misplaced according to the GPS records.

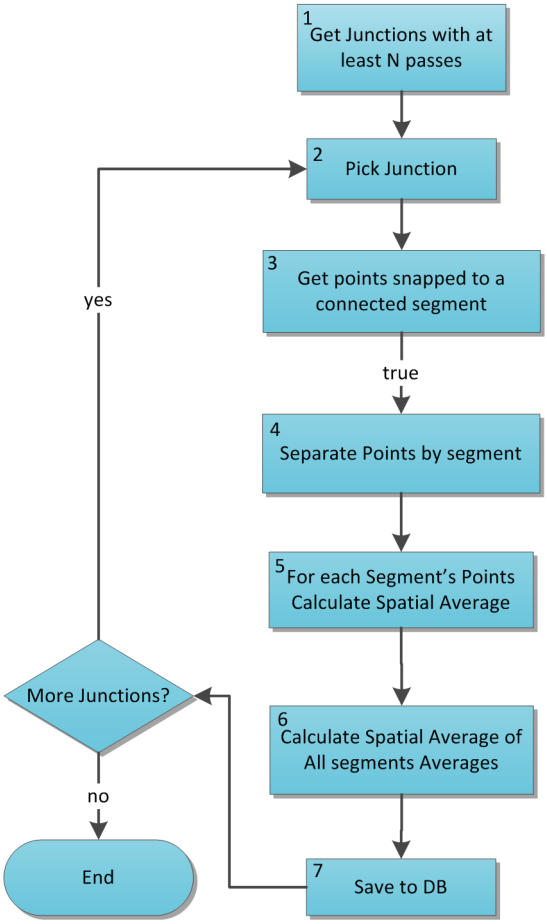


Figure 3.22: Junction Position Correction - Algorithm

3.9.1 Algorithm

Junction detection is made by average positioning near the existing one. By averaging GPS coordinates from users from all possible directions it is possible to estimate the new correct place for a junction.

Figure 3.22 is a representation of the algorithm. After having all points snapped, process only junctions with a high number of passes (1). For each junction get all points that were snapped to one of the connected segments and are within a selected radius from current junction position (3).

In order to balance the final average, all user points are separated according to the snapped segment (4). So, for each segment calculate the spatial average (section 3.10)(5) and then calculate the spatial average of all averages (6). The new junction is then saved to the database (7).

Results can be seen on section 4.2.7.

3.10 Spatial Average

Some of the previously presented algorithms depended on the calculation of a spatial average to create new geometries.

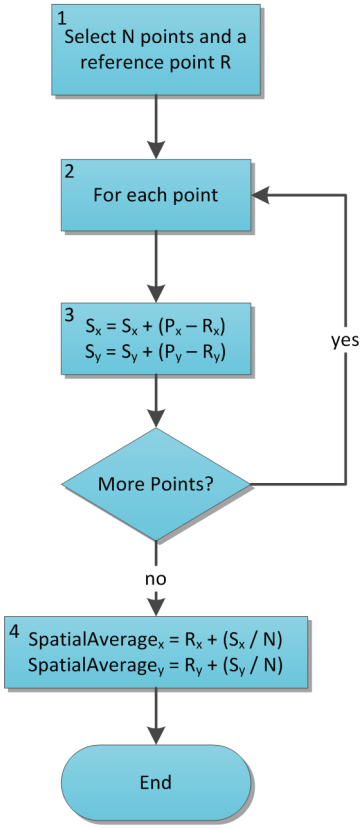


Figure 3.23: Spatial Average Algorithm

Figure 3.23 presents the algorithm that calculates the spatial average of N points.

The algorithm calculates the distance from each point to a reference point (3) and computes the average. The reference point translated by the average vector is the spatial average (4).

3.11 Summary

This chapter introduced the work-flow developed and the processes that permit detection of map updates. A complete method from the collection of data to the visualisation phase is proposed. The presented database model is a possible design that may be adapted to better suit other detections.

Although the methodology used as a base a TeleAtlas designed database, it is possible to use any other model as long as geometric segments and junctions are defined.

The next chapter will present an implementation of the system and applications.

Chapter 4

Implementation and Results

Based on the created process, previously introduced on chapter 3, an implementation of the system was developed.

This chapter presents the system architecture as well as the results obtained by applying test data sets to it.

4.1 Architecture

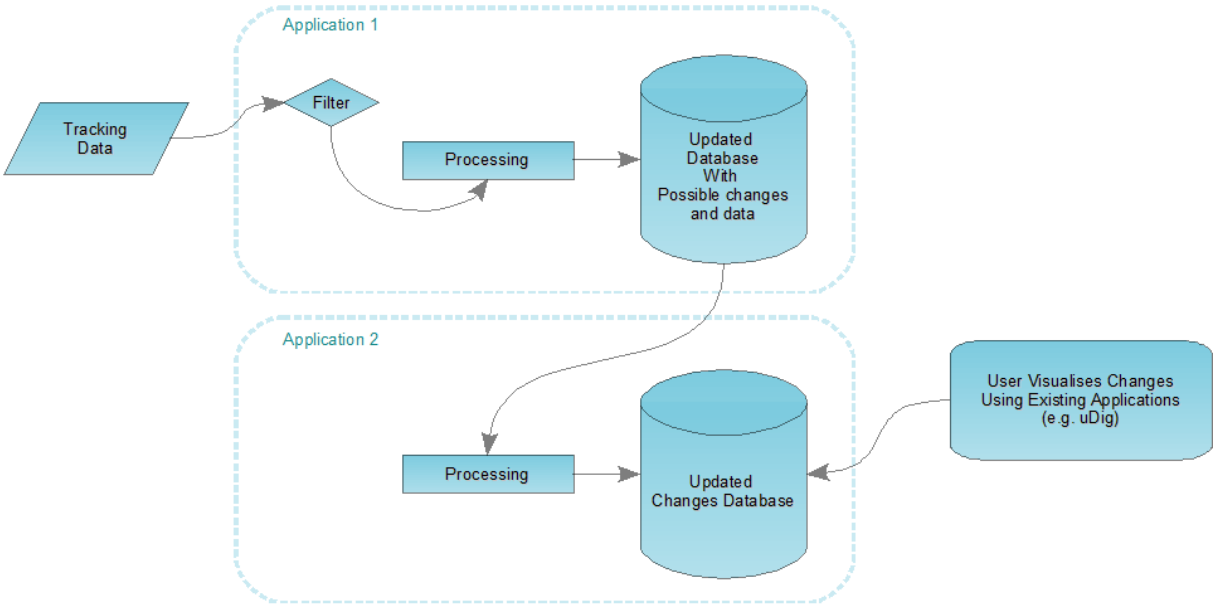


Figure 4.1: System Architecture

The implemented system is composed of a form of data collecting, two applications that process acquired data and a visualisation method as seen in 4.1.

4.1.1 Data Collecting

A PNA NDrive Touch was used with an altered software to collect points. The information that is recorded by the device is the normally available GPS information.

The device was programmed to record every available GPS data, while connected, to a file in CSV format that is downloaded by the user in order to be processed.

Since this is just a prototype there were no performance requirements and the collected data were not compressed nor selected. In a final version the collecting process should be streamlined to reduce possible performance penalties for PNA.

4.1.2 Applications

Algorithms previously presented depend on two phases. To ease programming, two applications were developed that make the system complete.

To archive data and results a database was also implemented.

With the objective of easing future programming, documentation for the whole application has been created using Doxygen.

4.1.2.1 Database

Based on the proposed model presented in 3.2 a database was built.

Since PostGres does not allow access to multiple databases on the same query, tables were added to the existing database PRTIP (TeleAtlas format of Portugal digital Map created by InfoPortugal) so processing could be easier.

The database contains the archive of collected data and can be exported to other formats through use of existing software.

4.1.2.2 Application One

The first application is composed of all the phase one algorithms plus some filters and preprocessing.

Before processing the collected data it is necessary to filter it by quality, excluding results with low accuracy and points with no relevant information (like when a car is stopped).

Repeated points form a cluster that may confuse some algorithms and occupies space on the database. To trim this data down, all points that are closer to the previous and have a null speed are converted to a point with a sum of seconds.

Points were filtered using HDOP, VDOP, PDOP as a quality measure to reduce outliers.

To reduce the application's run time, a location filter was also developed that only processes points within a defined region.

Another preprocessing that was done was the calculation of azimuth for slow speed points. Positions with low speed have an inaccurate azimuth due to accuracy errors from GPS and it is necessary to calculate direction from previous points.

Configuration of application one is done by changing a file. A configuration example is included in Appendix A that includes parameters used for the algorithms.

Application one starts by snapping the preprocessed points and then executes each algorithm in the chain. Adding algorithms is easy as it is only needed to add a new link in the chain. Every algorithm processes points and then passes data to the next link.

4.1.2.3 Application Two

Application two implements all the second phase of the algorithms. This application is much smaller when compared to application one. Most of the job is done by the PostGIS extension by selected queries to the database.

This application is responsible for the final output of results to the database. Implementation of the algorithms was done with classes, each class representing an algorithm. The application when run will execute all the final detections at once.

The options file is included in Appendix B and includes all the parameters of the algorithms and application.

4.1.3 Visualisation

Visualisation and comparison of the results to the road network can be done using any GIS editor. Figure 4.2 shows the application uDig with some map features loaded.

The user that is correcting the digital map has visual access to the proposed alerts and, in the same editor, can make corrections to the geometry features.

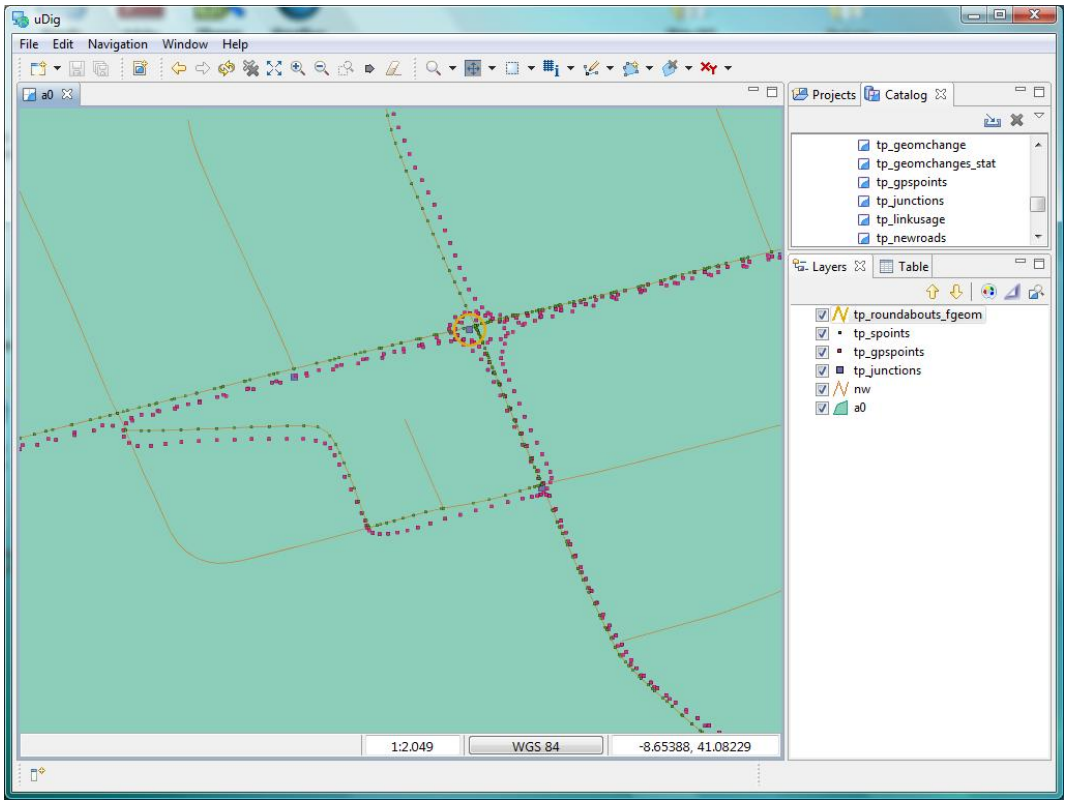


Figure 4.2: uDig Visualisation

4.2 Results

In order to verify correctness of the created algorithms, points were collected, with a sample rate of 1 second, by a PNA, across the district of Oporto, Portugal from November 2008 to January 2009.

The system processed 170 797 points and created results that includes 97 047 records and excludes points with an horizontal, vertical and positional dilution of precision above 2.9 as well as stationary points.

Results were visualised using uDig during development and the presented ones used Google Earth to better perceive the accuracy of the system since it provides a visualisation of the created data as well as an image of the region.

4.2.1 Map-Matching

Figure 4.3: Map-Matching - Results

The snapping process is one of the algorithms that were more tested. A small example of the results of snapping is shown in figure 4.3. It is shown multiple accesses to a highway and it is possible to see that the algorithm snapped points by the use of route information, instead of only direction and distance.

Another view of the image above is seen on figure 4.4. Red markers are the original GPS points and the green pins are the snapped positions.

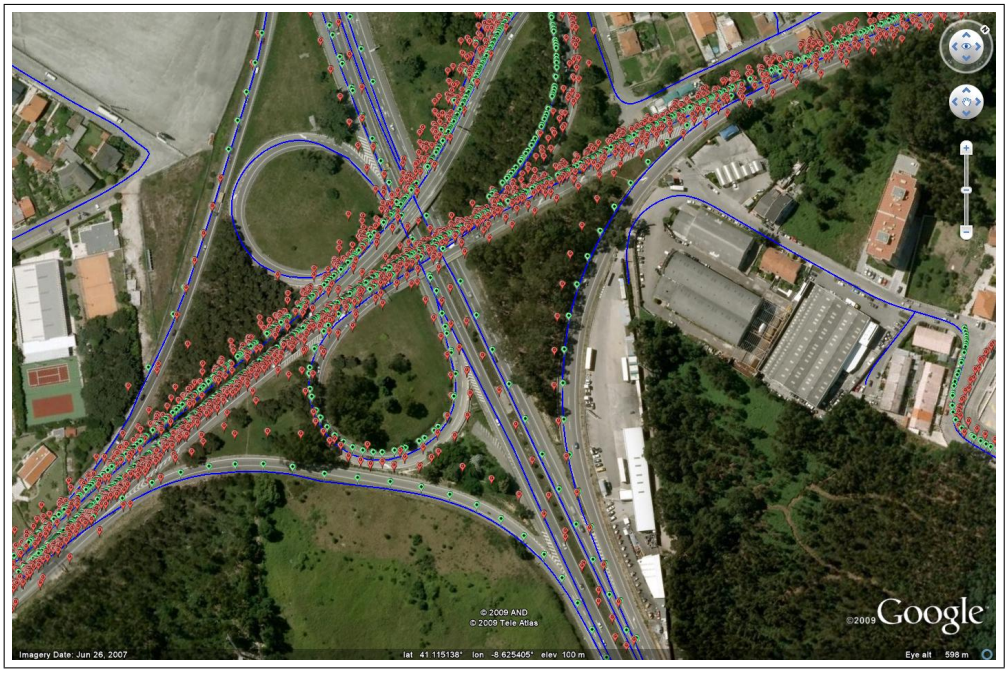


Figure 4.4: Map-Matching - Results 2

4.2.2 Detection of New Roads

Figure 4.5 and 4.6 are an example of detected new roads.

Blue lines indicates the current digital map and lines in green are the new detected roads. Red markers indicate collected GPS points.

For each distinct segment pair (start,end) there is a new detection. Different green lines for the same road occurs because the snapping algorithm chose a different start or end segment.

These figures indicate roads that were driven for the purpose of testing the method. From these results it will be possible to extend the existing digital map.

4.2.3 Detection of Roundabouts

Figure 4.7 and 4.8 are examples of roundabouts' detection. These roundbouts were also driven on purpose with different ways to imitate possible driving habbits.

Blue paths are the existing digital map, yellow lines are user traces and red marks are GPS points. The orange ellipse is the detected roundabout. Due to differences in software Google Earth represented the circumference as an ellipse although uDig drew a circumference. Circunference radius depends on the detected roundabout and tries to match the real roundabout size.

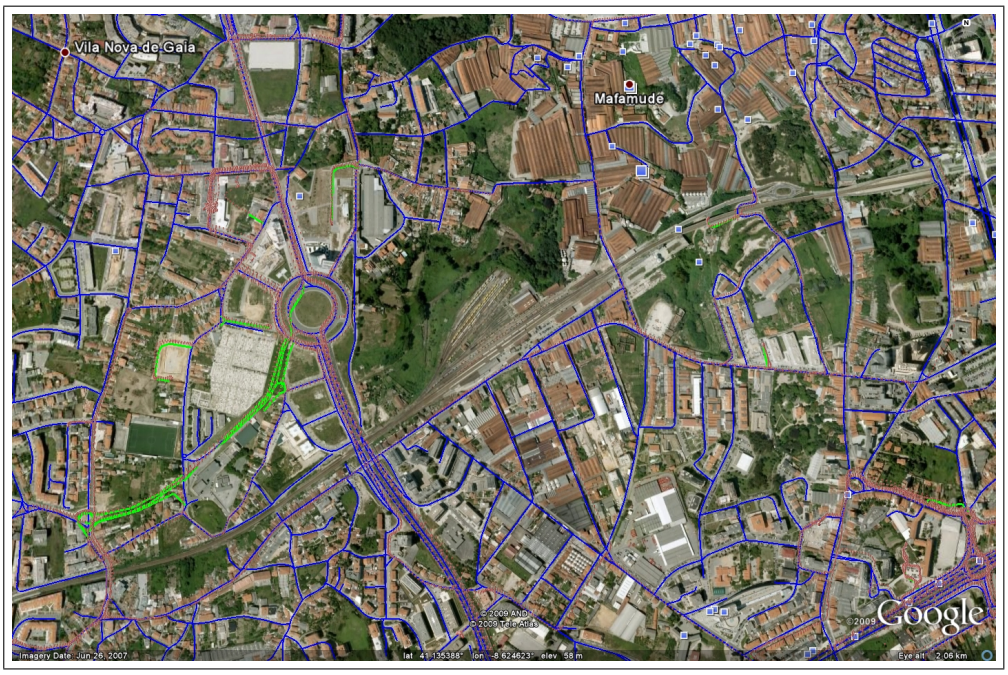


Figure 4.5: Detection of New Roads - Results 1

Figure 4.6: Detection of New Roads - Results 2

Figure 4.7: Detection of Roundabouts - Results 1

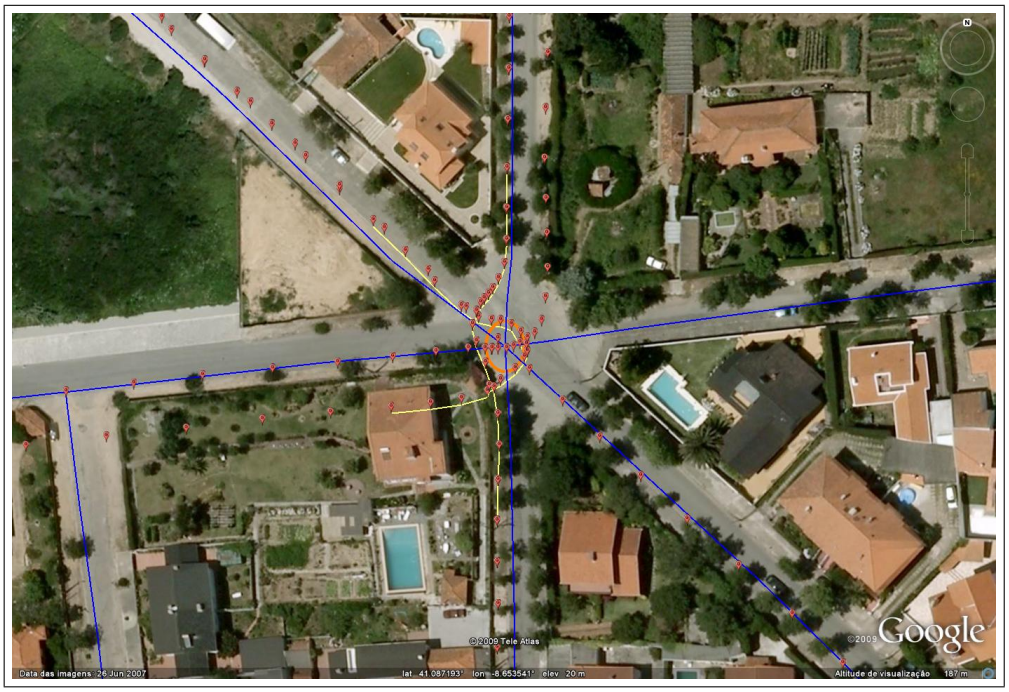


Figure 4.8: Detection of Roundabouts - Results 2

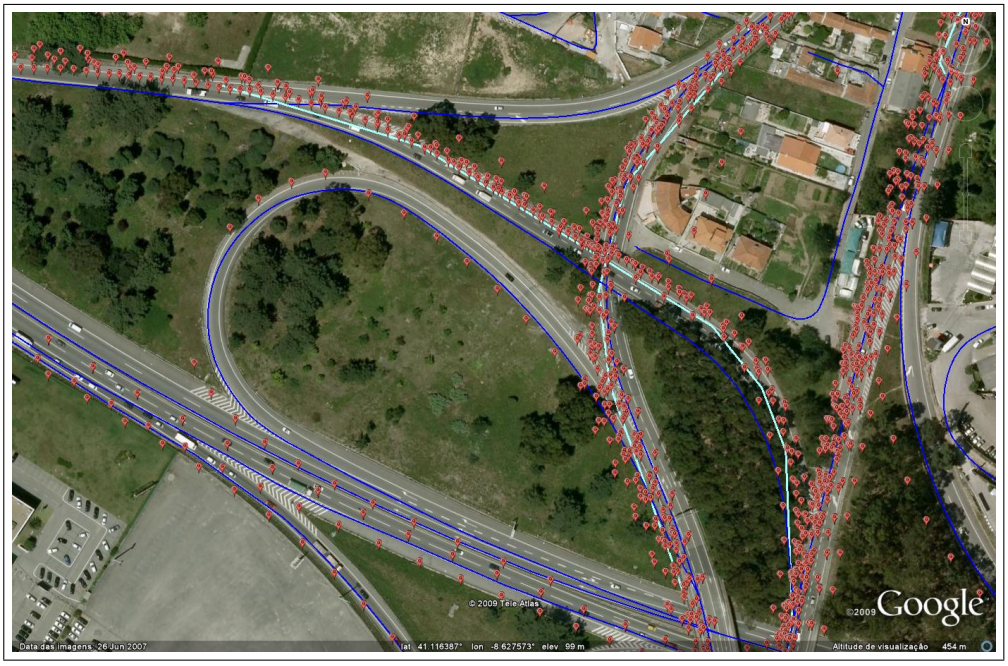


Figure 4.9: Detection of Wrong Road Geometry - Results

4.2.4 Detection of Wrong Road Geometry

Figure 4.9 presents an example of a detection of wrong road geometry. Digital road network is drawn in blue and the line in light blue represents the possible new geometry for the adjacent segment.

The detected geometry differences are for a part of a highway entrance. To have a more correct geometry location, it would be necessary to have a lot more passes.

4.2.5 Detection of Wrong Direction

Detection of wrong direction was applied to collected data and one of the results is presented on figure 4.10.

GPS points are represented by red pints with a 'G' and the detected wrong direction line is pink. The yellow arrows represent the specified authorised direction and blue arrows are user direction. This test was not driven on purpose. When visualising results it appeared and was verified.

4.2.6 Detection of Wrong Manoeuvre

By processing the acquired points with the system it was possible to detect a wrong manoeuvre.

In Figure 4.11 the red line indicates the detected wrong manoeuvre. GPS points are represented by red 'G' pins. The defined prohibited manoeuvre in the database is the path

Figure 4.10: Detection of Wrong Direction - Results

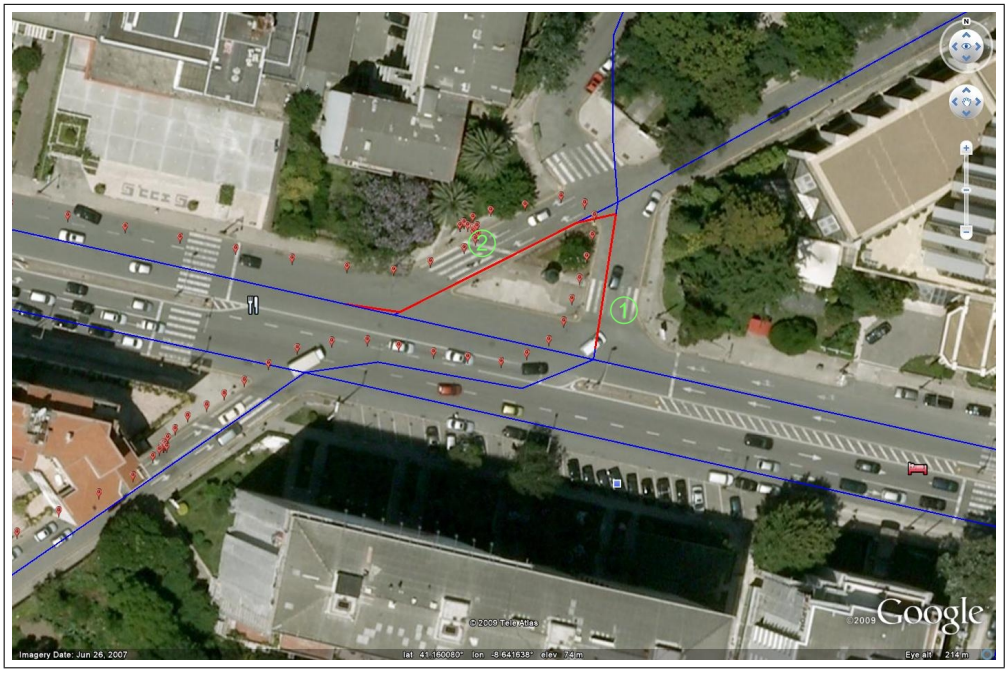


Figure 4.11: Detection of Wrong Maneuver - Results

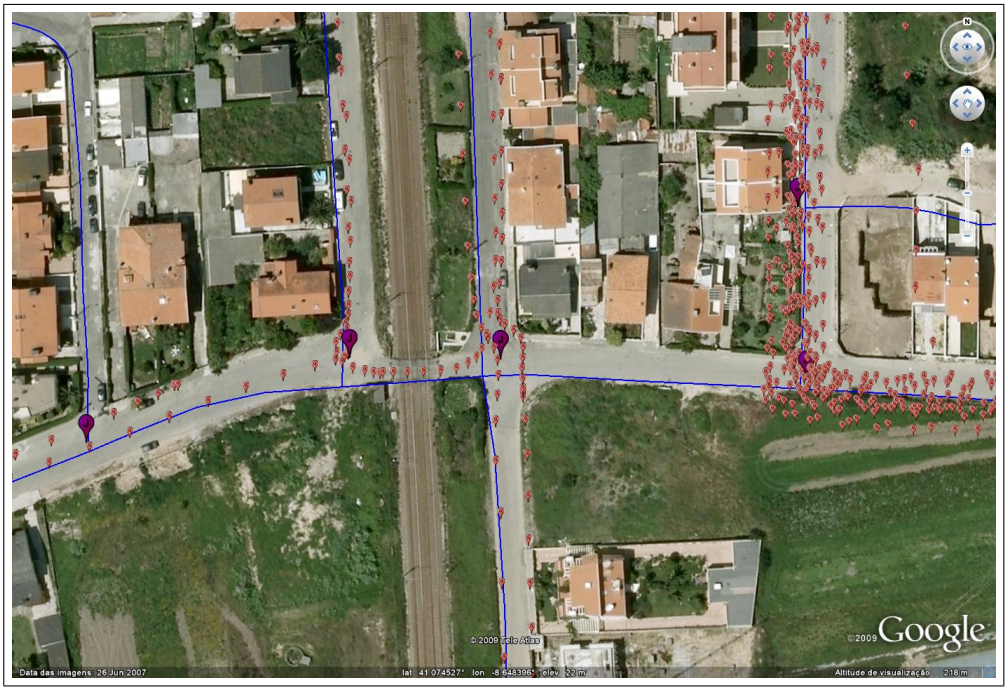


Figure 4.12: Junction Position Correction - Results

from ’1’ to ’2’. GPS points indicates a travel from ’1’ to ’2’, so it is possible that there is an error on the digital map.

This detection also occurred without the need for a search of possible cases. It was presented as an error and verified.

4.2.7 Junction Position Correction

After applying the algorithm to the snapped points there were few results for this detection type which indicates a good positioning of junctions tested.

Figure 4.12 represents a possible junction correction. Pins with a ’J’ represent the new junction location and blue intersections are the present locations of junctions. GPS points are marked with a red pin ’G’. Only junctions with many passes and travels from all directions will yield correct results.

4.3 Summary

The developed system, although a prototype for a future application, correctly processes and implements the previously presented algorithms. This system can be used as a basis for future development since it is easily extendable.

The system was tested by the use of collected points and results were analysed. This chapter presented some of the results of detected changes.

Even with a small data set interesting results were found. By using a more complete data set, with more passes on the same places, by different users who employ different driving habits it will be possible to obtain results with an higher degree of precision.

Results for an even bigger data set could have provided better results and more complete and accurate information.

On the next chapter, achieved goals and future work guidelines are presented.

Chapter 5

Conclusions and Future Work

5.1 Achieved Goals

Personal Navigation Assistants depend on the accuracy of GPS devices and the digital map. Current digital maps still have a high degree of imprecision. To aid map companies to provide more accurate maps at lower costs, some algorithms to detect changes and differences to the road network were developed. This method uses GPS positions, gathered from existing PNAs, for the comparison of the real road network with the digital representation. Methods for the detection of new roads, roundabouts, road geometry, traffic direction, prohibited manoeuvres and junction position were developed successfuly. A process to match locations of GPS positions to the digital map was also developed.

Previous results demonstrated that the proposed algorithms have a possible application in the detection of road network changes. Although an extensive test has to be made in order to define the full capacities, and possible corrections to the method, it is clear that it is possible to detect map differences from user's positions.

Although all the objectives proposed for this project were completed successfully with satisfactory results it was impossible to completely verify the correction rate of the algorithms. To achieve this information it is necessary to further test the proposed method with a large set of roads and a very large number of users. Only then, with statistical information, it will be possible to correctly assure the quality of the method.

Integration and improvement of the implemented system might be a positive step, as it will reduce costs aggregated with network updating.

5.2 Future Work

An important improvement that should be implemented is the possibility to ignore current generated detections so the system does not create them again until solicited. This would be important because the algorithm will always produce some wrong detections that will have to be ignored by the user. The applications should also add the possibility to re-run algorithms for a selected area to be able to test how corrections to the digital map resolve a detection.

Due to the limited time-frame of twenty weeks this project had, it was impossible to develop other types of detections and other possible features. Some preliminary tests were made for the calculation of the average segment speed as this is an important step in the improvement of routing applications and this should be one of future objectives. Another interesting feature related with average speed is the relation with meteorology, by accessing a database with meteorologic data and comparing GPS points speed, it might be possible to generate more accurate speed information.The addition of more possible detections, like number of lanes or the road size, might be important to add more complete information to the existing digital maps.

Since current digital maps are 2D only, by the use of the altitude parameter from GPS points, it is possible to generate a 3D map. Although this is not yet needed for the current applications, one day, with the addition of new services, might be required.

Location based services are being an increasing need in the navigation environment. By having a more complete digital map, with more information besides the road network, it is possible to improve possible services.

Despite the fact this project was focused on the road network, it is possible that a similar type of detections could be used to expand the existing set of features in the digital maps. A possibility is the addition of city features and inside routes like a street walk or garden.

Appendix A

Application One Options

This appendix contains the configuration options for the implemented algorithms on the phase one application. These values alter the output of the algorithms.

```
# Configuration file for MapMatcher
# defaults are within ()

# Enabled / Disabled Modules
DetectNewRoadsPointConsumer.enabled = true

DetectGeomChangesPointConsumer.enabled  = true
DetectNewRoundaboutPointConsumer.enabled  = true
DetectWrongWay2PointConsumer.enabled  = true
DetectWrongManoeuvrePointConsumer.enabled  = true

DBSaveGpsPointConsumer.enabled       = true
DBSaveSnapPointConsumer.enabled  = true

LinkUsagePointConsumer.enabled  = false

LocationPointFilter.enabled  = false

#-------------------------------------------------------------------------------

# Enabled / Disabled Debug Output
DetectNewRoadsPointConsumer.debug_enabled  = false
CalculateSpecialBearingPointConsumer.debug_enabled  = false
Candidate.debug_enabled  = true
DBAccess.debug_enabled  = false
DBSaveGpsPointConsumer.debug_enabled  = false
DBSaveSnapPointConsumer.debug_enabled  = false
DetectGeomChangesPointConsumer.debug_enabled  = false
Mapper.debug_enabled  = false
Point.debug_enabled  = false
DetectNewRoundaboutPointConsumer.debug_enabled  = false
DetectWrongWay2PointConsumer.debug_enabled  = false
PointAlternative.debug_enabled  = false
CleanerPointConsumer.debug_enabled  = false
DetectWrongManoeuvrePointConsumer.debug_enabled  = false

#-------------------------------------------------------------------------------
```

```
# Create tables
LinkUsagePointConsumer.create_table  = false
DetectNewRoundaboutPointConsumer.create_table   = false

DetectNewRoadsPointConsumer.create_table = false
DetectGeomChangesPointConsumer.create_table = false

DetectWrongManoeuvrePointConsumer.create_table = false
DetectWrongWay2PointConsumer.create_table = false

DBSaveSnapPointConsumer.create_table = false
DBSaveGpsPointConsumer.create_table = false


# Reset tables
LinkUsagePointConsumer.reset_table  = false
DetectNewRoundaboutPointConsumer.reset_table  = true
DetectNewRoadsPointConsumer.reset_table = true
DetectGeomChangesPointConsumer.reset_table = true

DetectWrongManoeuvrePointConsumer.reset_table = true
DetectWrongWay2PointConsumer.reset_table = true

DBSaveSnapPointConsumer.reset_table = true
DBSaveGpsPointConsumer.reset_table = true

#-------------------------------------------------------------------------------

# Filters
#DOPPointFilter
#Max HDOP value acceptable, ignored with a negative value (2.9)
DOPPointFilter.hdop  = 2.9
#Max VDOP value acceptable, ignored with a negative value (2.9)
DOPPointFilter.vdop  = 2.9
#Max PDOP value acceptable, ignored with a negative value (2.9)
DOPPointFilter.pdop  = 2.9

#MinIDPointFilter
#Minimum acceptable ID (0)
MinIDPointFilter.minid  = 0

#main
#Break on max id (800000)
main.maxid  = 140000

#LocationPointFilter
#Center Point X
LocationPointFilter.center.x  = -8.647592
#Center Point Y
LocationPointFilter.center.y  = 41.0744
#Generated box size
LocationPointFilter.boxsize  = 0.003

#-------------------------------------------------------------------------------


# Producers
```

```
#CSVPointProducer
#File to process
CSVPointProducer.source_file  = dataImport
#Min Distance to accept as clustering (0.000004)
CSVPointProducer.mindistance  = 0.000004
#Option Min Speed to accept as clustering (0)
CSVPointProducer.minspeed  = 0
#Id to base numbering
CSVPointProducer.start_id  = 0


#DBPointProducer
#Start offset to read from DB (0)
DBPointProducer.startoffset  = 0
#Number of points to retrieve at a time from DB (1000)
DBPointProducer.npoints  = 1000
#Base point id to retrieve from table (0)
DBPointProducer.basepoint  = 0
#Top point id to retrieve from table (100000)
DBPointProducer.toppoint  = 100000

#-------------------------------------------------------------------------------

# Others

#CalculateSpecialBearingPointConsumer
#Minimum speed to process bearing (3)
CalculateSpecialBearingPointConsumer.minspeed  = 3
#Min distance from otherpoint to process bearing (0.0002)
CalculateSpecialBearingPointConsumer.mindistance  = 0.0002
#Distance to otherpoint (otherpoint and current point determine bearing) (0.00007)
CalculateSpecialBearingPointConsumer.distance_to_other_point = 0.00007


#Candidate
#Op_val_weight for comparison between candidates (0.2)
Candidate.op_val_weight  = 0.2



#-------------------------------------------------------------------------------

# Databases

#DBAccess
#Connection String (dbname=PRT_IP)
DBAccess.connection_string  = dbname=PRT_IP1


#DBSaveGpsPointConsumer
#Maximum Number of transaction points (2000)
DBSaveGpsPointConsumer.max_transaction_points  = 2000


#DBSaveSnapPointConsumer
#Maximum Number of transaction points (2000)
DBSaveSnapPointConsumer.max_transaction_points  = 2000
```

```
#-------------------------------------------------------------------------------


# Detect Geom Changes
#Minimum Distance value to start saving geom (0.00020)
DetectGeomChangesPointConsumer.mindistance  = 0.00020
#Max distance to junction to accept as a complete segment (0.00040)
DetectGeomChangesPointConsumer.max_distance_to_junction = 0.00040

#-------------------------------------------------------------------------------


# Mapping
#Search Radius for links (0.00045)
Mapper.links_search_radius  = 0.00045
#Maximum Acceptable distance to correct points near Snapped to gps (1.3)
Mapper.correct_near_max_acceptable_distance_ratio  = 1.3
#Maximum Number of continuous bacout << *p << endl;cktrack passes (7)
Mapper.max_backtrack_passes  = 8
#Maximum link passes before candidate removal (3)
Mapper.max_link_passes_before_erase  = 3
#Backtrack multiplier weight (0.7)
Mapper.backtrack_multiplier_weight  = 0.7
#Backtrack Other Multiplier weight (0.8)
Mapper.backtrack_multiplier_others_weight  = 0.8
#Backtrack Maximum candidate distance from snap coord to gps coord (0.00030)
Mapper.backtrack_max_candidate_distance  = 0.00030
#Backtrack weight proportion (0.333333)
Mapper.backtrack_weight_proportion  = 0.333333
#Use last snapped id for next initial snapping (true)
Mapper.initial_snapping_use_last_id  = true


#Point
#Maximum distance from gps coord to snap coord (0.00045)
Point.max_snap_distance  = 0.00045
#Point weight modifier distance constant (100)
Point.weight_distance  = 100
#Point weight modifier azimuth constant (300)
Point.weight_azimuth  = 300
#Point weight modifier power azimuth value (7)
Point.weight_pow_azimuth  = 7
#Point weight max distance divider (0.0003)
Point.weight_max_distance  = 0.0003
#Point weight modifier distance when nobearing (100)
Point.nobearing_weight_distance  = 100
#Point weight modifier azimuth when nobearing (75)
Point.nobearing_weight_azimuth  = 75
#Point weight modifier power azimuth value when nobearing (7)
Point.nobearing_weight_pow_azimuth  = 7
#Point weight max distance divider when nobearing (0.0003)
Point.nobearing_weight_max_distance  = 0.0003
#Point reorder add to reachable_distance (0.0001)
Point.reorder_reachable_distance_plus  = 0.0001
#Point reorder apply if reachable distance is below (0.0002)
Point.reorder_min_reachable_distance  = 0.0002
```

```
#Point reorder when samesegment max acceptable distance ratio (1.2)
Point.reorder_samesegment_max_acceptable_distance_ratio = 1.2
#Point reorder when samesegment unacceptable distance apply reach_dist (1.5)
Point.reorder_samesegment_unacceptable_distance_reach_dist = 0.5
#Point reorder point continuity max acceptable distance (0.0004)
Point.reorder_point_continuity_max_distance  = 0.0004
#Point reorder compare direction cos power (6)
Point.reorder_compare_direction_cos_pow  = 6
#Point discard candidates with cos value below (0.5)
Point.discard_candidates_cos_value  = 0.5
#Point continuity max time acceptable (7)
Point.continuity_max_time_between  = 7
#Point continuity base distance (0.0004-34.40968018 0.42803272985)
Point.continuity_base_distance  = 0.00045
#Try to select road segment with the correct direction (true)
Point.try_correct_way_segment  = true

#-------------------------------------------------------------------------------

# Detect New Roundabout
#Where to stop picking points for id2 (0.0003)
DetectNewRoundaboutPointConsumer.stop_distance_to_junction_id2 = 0.0003
#Where to stop picking points for id1 (0.0003)
DetectNewRoundaboutPointConsumer.stop_distance_to_junction_id1 = 0.0003
#Minimum distance between points with a speed above OPT_min_speed_to_consider_bearing to include bearings (0.
DetectNewRoundaboutPointConsumer.min_distance_between_points_to_consider_bearing = 0.00001
#Minimum speed to consider a valid bearing (3)
DetectNewRoundaboutPointConsumer.min_speed_to_consider_bearing = 3
#Ignore turns to the right above this angle in radians (0.261799) {= 15 degrees}
DetectNewRoundaboutPointConsumer.ignore_turn_to_the_right_radians = 0.261799
#Minimum difference between maxdistance and mindistance relative to street (0.00005)
DetectNewRoundaboutPointConsumer.min_difference_distance = 0.00004
#Minimum maxdistance (0.000035)
DetectNewRoundaboutPointConsumer.min_max_distance  = 0.000035
#Minimum Difference between maxsin and minsin (0.1)
DetectNewRoundaboutPointConsumer.min_diff_sins  = 0.1
#Minimum maxdifang (0.27925268) {= 16 degrees}
DetectNewRoundaboutPointConsumer.min_max_difference_between_angles = 0.27925268
#Maximum acceptable proportion between angle to the left and to the right (3)
DetectNewRoundaboutPointConsumer.max_proportion_turn_left_turn_right = 3

#-------------------------------------------------------------------------------


# Detect New Roads
#Search radius for segments near point (0.00015)
DetectNewRoadsPointConsumer.links_search_radius  = 0.00015
#Distance from present point to point from where to calculate projection error (0.00005)
DetectNewRoadsPointConsumer.take_error_from_distance =  0.00005
#Maximum acceptable distance error after calculation (0.00020)
DetectNewRoadsPointConsumer.acceptable_error  =  0.00020
#Maximum return to snap difference when a snapped point appears during a snapped to gps track (0.00020)
DetectNewRoadsPointConsumer.return_to_snap_acceptable_error =  0.00020
#Maximum ratio point/links to accept new road (0.8)
DetectNewRoadsPointConsumer.max_ratio_point_haslinks = 0.8
#Maximum ratio links per point to accept new road (0.8)
DetectNewRoadsPointConsumer.max_ratio_links_per_point = 0.8
```

```
#Accept same start id and endid for a street (true)
DetectNewRoadsPointConsumer.accept_sameid  = true


#-------------------------------------------------------------------------------------
```

Appendix B

Application Two Options

This appendix contains the configuration options for the implemented algorithms on the phase two application. These values alter the output of the algorithms.

```
DBAccess.connection_string  = dbname=PRT_IP
DBAccess.debug_enabled  = false

Roundabouts.min_ratio = 1
Roundabouts.min_votes = 4

Roundabouts.num_segments = 20
Roundabouts.zig_zag = false


Roundabouts.create_table = false
Roundabouts.reset_table = true


Roundabouts.min_ratio = 1
Roundabouts.min_votes = 1

WrongWay.create_table = false
WrongWay.reset_table = true
WrongWay.min_usage = 2

#MAX PERCENTAGE to accept as a NO for that way
WrongWay.max_oneway_percentage = 5

WrongManoeuvre.min_count   = 1
WrongManoeuvre.min_ratio = 0.05
WrongManoeuvre.min_avg_usage = 5
WrongManoeuvre.create_table = false
WrongManoeuvre.reset_table = true

GeomChanges.create_table = false
GeomChanges.reset_table = true
GeomChanges.radius = 0.00015
GeomChanges.stepping = 0.00005
GeomChanges.min_ratio_points_length = 0.3
GeomChanges.radius_simplifier = 0.00002
```

```
JunctionChanges.create_table = false
JunctionChanges.reset_table = true
JunctionChanges.min_junction_passes = 2


JunctionChanges.max_distance_junction = 0.00015

NewRoads.create_table = false
NewRoads.reset_table = true
NewRoads.radius = 0.00015
NewRoads.stepping = 0.00005
NewRoads.min_tracks = 1
NewRoads.radius_simplifier = 0.00004
```

References

[EBPB07] Frank Ekpenyong, Allan Brimicombe, and Dominic Palmer-Brown. Updating road network databases: Road segment grouping using snap-drift neural network. In *Geographical Information Science Research Conference 11-13th April (GISRUK'07)*, Maynooth, Ireland, April 2007.

[Fri05] Laura Friese. Updating the spatial alignment attributes of digital maps using gps points. Master's thesis, Princeton University, 2005.

[Gre02] Joshua S. Greenfeld. Matching gps observations to locations on a digital map. In *81st Annual Meeting of the Transportation Research Board*. Department of Civil and Environmental Engineering, New Jersey Institute of Technology, 2002. http://www.njtide.org.

[LLZ05] X. Li, H. Lin, and Y. Zhao. A connectivity-based map matching algorithm. Technical report, The Department of Geography and Resource Management & The Joint Laboratory for GeoInformation Science, The Chinese University of Hong Kong, Shatin, N.T. Hong Kong, S.A.R., 2005.

[MHA04] F. Marchal, J. Hackney, and K.W. Axhausen. Efficient map-matching of large gps data sets - tests on a speed monitoring experiment in zurich. Technical report, Arbeitsbericht Verkehrs- und Raumplanung, Institut für Verkehrsplanung und Transportsysteme, ETH Zürich, Zürich, 2004.

[NG00] Val Noronha and Michael F. Goodchild. Map accuracy and location expression in transportation - reality and prospects. *Transportation Research Part C*, (8):53–69, 2000. http://www.elsevier.com/locate/trc.

[pos09a] Postgis web site, February 2009. http://postgis.refractions.net/, Last Accessed in February 2009.

[pos09b] Postgresql web site, February 2009. http://www.postgresql.org/about/, Last Accessed in February 2009.

[QON06] Mohammed A. Quddus, Washington Y. Ochieng, and Robert B. Noland. Integrity of map-matching algorithms. *Transportation Research Part C*, (14):283–302, 2006. http://www.elsevier.com/locate/trc.

[QON07] Mohammed A. Quddus, Washington Y. Ochieng, and Robert B. Noland. Current map-matching algorithms for transport applications: State-of-the art and future research directions. *Transportation Research Part C*, (15):312–328, May 2007. http://www.elsevier.com/locate/trc.

[RLW99] Seth Rogers, Pat Langley, and Christopher Wilson. Mining gps data to augment road models. pages 104–113. DaimlerChrysler Research and Technology Center, 1999.

[sYpKsC05] Jae seok Yang, Seung pil Kang, and Kyung soo Chon. The map matching algorithm of gps data with relatively long polling time intervals. *Journal of the Eastern Asia Society for Transportation Studies*, 6:2561–2573, 2005.

[tel08] *Tele Atlas Multinet Shapefile 4.4 Format Specifications*, October 2008.

[tom09a] Tomtom iq-routes web site, February 2009. http://www.tomtom.com/page/iq-routes, Last Accessed in February 2009.

[tom09b] Tomtom mapshare web site, February 2009. http://www.tomtom.com/page/mapshare, Last Accessed in February 2009.

[TZ03] Ameur Touir and Rached Zantout. Design and implementation of an automatic road network map processing system using gps technology. In *Book of Abstracts. ACS/IEEE International Conference on Computer Systems and Applications*. Department of Computer Science, King Saud University, Saudi Arabia, July 2003.

[udi09] udig web site, February 2009. http://udig.refractions.net/, Last Accessed in February 2009.

[WBK00] Christopher E. White, David Bernstein, and Alain L. Kornhauser. Some map matching algorithms for personal navigation assistants. *Transportation Research Part C*, (8):91–108, 2000. http://www.elsevier.com/locate/trc.

[YW04] Huabei Yin and Ouri Wolfson. A weight-based map matching method in moving objects databases. In *16th International Conference on Scientific and Statistical Database Management (SSDBM'04)*. Department of Computer Science, 851 South Morgan, 2004.

[Zho04] Jianyu Zhou. A three-step general map matching method in the gis environment: Travel/transportation study perspective. Technical report, Department of Geography, University of California Santa Barbara, 2004.