Database support for pedestrian region-based navigation

Jean Pierre Gatera
March, 2009
Database support for pedestrian region-based navigation

by

Jean Pierre Gatera

Thesis submitted to the International Institute for Geo-information Science and Earth Observation in partial fulfilment of the requirements for the degree in Master of Science in Geoinformatics.

Degree Assessment Board

Thesis advisor  Dr. Javier M. Morales
Dr. Otto Huisman

Thesis examiners  Dr. Ir. Rolf A. de By
Dr. A. Wombacher

INTERNATIONAL INSTITUTE FOR GEO-INFORMATION SCIENCE AND EARTH OBSERVATION
ENSCHÉDE, THE NETHERLANDS
Disclaimer

This document describes work undertaken as part of a programme of study at the International Institute for Geo-information Science and Earth Observation (ITC). All views and opinions expressed therein remain the sole responsibility of the author, and do not necessarily represent those of the institute.
Abstract

With progress in miniaturization of computing devices and with availability of positioning systems enabling location-based services such as car navigation systems, pedestrians are also gaining interest in getting reliable guiding instructions to reach a desired location, and in getting location-based information about their environment. However, pedestrians can not be assisted optimally by the same systems as cars, because cars definition of routes is based only on road network whereas pedestrians can use road network as well as environments that lack predefined paths such as parks or city squares. Therefore, supporting pedestrians' navigation requires new approaches for definition of navigation routes such as the region-based definition of routes which takes into account the ability of pedestrians to move inside areas that lack predefined paths.

Supporting pedestrian region-based navigation requires a set of services capable of continuously tracking pedestrians' changing directional positions, and linking these positions to suitable routes and to relevant information about their environment. For this, a database support is indispensable, for the management of this time changing data about pedestrian's motion (i.e. trajectory data), for the storage of information about the environment and for supporting the querying of this data. However, database support of this continuously changing position is a challenge because of the continuous updates of this data; this is due to the deficient of current data models in representing properly trajectories associated with moving objects.

The aim of this research is to model concepts related to pedestrian region-based navigation, to define a data model suitable for database representation of these concepts and finally to design a database based on the defined data model in order to effectively support pedestrian region-based navigation. Based on an application scenario representing a pedestrian assisted by a SAMS (Spatially-aware map service) in an urban environment, a requirement analysis was done and an information model capturing requirements for supporting pedestrian region-based navigation was built. Based on the information model, a data model was defined for database representation of pedestrians' trajectories and other required data. Finally, a database for supporting pedestrian region-based navigation was designed based on the defined data model and using PostgreSQL\ PostGIS as the implementation DBMS.

Keywords

Pedestrians Region-Based Navigation, Trajectory Data, Moving Object Database, Spatially-Aware Map Services
Contents

Abstract i

List of Tables vii

List of Figures ix

Acknowledgements xi

1 Introduction 1

1.1 Motivation and Problem statement 1

1.1.1 Motivation 1

1.1.2 Problem statement 2

1.2 Research identification 2

1.2.1 Research objectives 2

1.2.2 Research questions 3

1.2.3 Innovation aimed at 3

1.3 Method adopted 3

1.4 Thesis outline 6

2 Related research 7

2.1 Pedestrian navigation systems 7

2.1.1 Pedestrian behaviour 7

2.1.2 Pedestrian information requirements 10

2.1.3 Spatially-aware map services 12

2.1.4 Region-based navigation 13

2.2 Moving objects databases 14

2.2.1 Modeling moving objects 14
## List of Tables

2.1 Empirical methods in pedestrian monitoring [MG07a] .... 9

4.1 relevant spatial objects with their corresponding data types .. 46

4.2 Datatypes for representing pedestrian trajectories ........ 49
# List of Figures

1.1 Method adopted for the research ........................................... 5

2.1 Factors influencing route choice behaviour. Source [MG07a] ........ 8

2.2 Generation of a dataset suitable for pedestrian navigation sys-

tems. Source [Bir07] .......................................................... 12

2.3 Related research summary ................................................... 18

3.1 Analysis of the application scenario ......................................... 25

3.2 Overview of the pedestrian navigation system .......................... 32

3.3 The first part of the information model .................................... 36

3.4 The second part of the information model ............................... 37

3.5 The complete information model .......................................... 39

4.1 OpenGIS simple features model [Ope99] .............................. 45

4.2 Example of UML class notation ............................................ 52

4.3 The conceptual schema ..................................................... 53

4.4 The logical schema ......................................................... 57

5.1 DDL generation dialog window ............................................. 61

5.2 The DDL file generated with SQL statements ........................ 61

5.3 Example of a geometry column generation command ............... 63

5.4 Method for creating pedestrian segments and nodes inside an un-

constrained navigable area .................................................. 64

5.5 Testing dataset generation process ........................................ 69

5.6 Overview of the testing dataset .......................................... 70

5.7 Example of a loader command converting a shapefile into SQL

statements ................................................................. 70
List of Figures

5.8 Example of SQL statements generated by the conversion of a shapefile .................................. 71
5.9 Example of SQL statements inserting data into tables ......................................................... 72
5.10 Query retrieving the last update location and direction of a pedestrian ................................. 74
5.11 Verification of the query retrieving the last update location and direction of a pedestrian ............ 75
5.12 Query retrieving the location of the Hello mall ................................................................. 75
5.13 Verification of the query retrieving the location of the Hello mall ........................................ 76
5.14 Query retrieving building registered as shops ..................................................................... 77
5.15 Query retrieving the location of the nearest shop .................................................................. 78
5.16 Query retrieving the location of the UTC building ............................................................. 79
5.17 Query retrieving the location of a restaurant in less than five meters ................................. 80
5.18 Query retrieving the location of a restaurant in less than sixty meters ................................ 80
5.19 Query retrieving the location of a shop within 200 meters ............................................... 81

A.1 SQL statements creating tables ............................................ 94
A.2 SQL statements adding geometry columns to tables ......................................................... 94
A.3 SQL creating SPATIAL_REF_SYS and GEOMETRY_COLUMNS tables .................................. 95
Acknowledgements

First of all, I would like to thank God for his love and for protecting me till now.

I would like to express my deep and sincere thanks to my supervisor Dr. Javier Morales for his continuous guidance, support and rich discussions we had along the thesis period. My thanks go also to Dr. Otto Huisman my second supervisor for his valuable comments.

I would like to take this opportunity to thank the Netherlands Government through NUFFIC and ITC for providing me the opportunity to pursue MSc studies.

Many thanks go to all my classmates and friends in ITC for their kindness and fun we had together. It was a great experience to work and to study with all of you. I wish you all the best.

Finally, my warm thanks go to my family for their love, encouragement and support during my studies at ITC. May God bless you all.
Chapter 1

Introduction

1.1 Motivation and Problem statement

1.1.1 Motivation

With rapid development of communication technology and the success of car navigation systems, pedestrians are also gaining interest not only in reliable guiding instructions to reach a desired location but also in a variety of information about their environment. Nevertheless, pedestrians cannot be assisted by the same systems as cars, because their movement environment is not similar and information about the environment provided to drivers by car navigation systems is not suitable for pedestrians. While cars move only along roads network, pedestrians can use network environments such as walkways or streets but they can also use network independent environment like parks, city squares and train station where their movements are not constrained by predefined paths. Hence, car generation of routes which is based only on road network is not appropriate for pedestrian navigation and information about the environment provided by car navigation is also not optimal for pedestrians because it is related to place accessible only by cars. Therefore, pedestrian navigation systems must be based on new approach for routes definition like the region-based definition of routes which takes into account the ability of pedestrians to move inside areas that lack paths network [WC06]

Supporting pedestrian region-based navigation requires a set of services capable of tracking their continuously changing positions and providing route guidance as well as other information that may be relevant to pedestrians. According to [ZPM03], such services are referred to us as spatially-aware map services (SAMS) since they will be aware of pedestrian directional position and its related environment information. Due to the need of management of time changing data about pedestrian’s motion (i.e. trajectory data), and due to the need of storage of information about the environment as well as the need of querying of this data, a database support of the above specified services is indispensable [ZPM03]. A database for supporting the SAMS will be responsible
of a well structured management of trajectory data associated with pedestrians and of storage of geographical information of the environment in which these services are meant to be provided.

### 1.1.2 Problem statement

Current database do not support the management of trajectory data associated with moving objects like pedestrians. As trajectory data are associated with continuously changing data such as time, position, direction, velocity and as they may also be associated with time varying standard deviation and errors, current data models do not support them because of the continuous updates of this data [MdB05]. In addition to this, most of existing query languages are non temporal and limited to accessing a single database state. Therefore, challenges for current databases in order to support trajectory data associated with pedestrians reside in the deficient of current data models in representing trajectory data. Therefore, database support of SAMS for pedestrian region-based navigation requires new database models.

New conceptual, logical and physical models have to be designed in order to effectively represent, manage and operate on trajectory data associated with pedestrians. Once these data models suitable for database representation of trajectory data are available, databases can be built and be used as a data repository on top which a spatially-aware map service consisted of separate autonomous services can be integrated and operate in requesting the location of a pedestrian and deliver to him route instructions as well as other relevant information about its environment.

### 1.2 Research identification

#### 1.2.1 Research objectives

The aim of this research is to model concepts related to pedestrian region-based navigation and to define data models that represent these concepts and finally to design a database based on these data models in order to effectively support SAMS that assist pedestrians. The database to be designed in this research will be responsible for the management of trajectory data and other relevant data associated with pedestrian region-based navigation application domain.

The specific objectives are:

1. To model the spatial and spatio-temporal aspects of the world associated with pedestrian region-based navigation;

2. To design conceptual, logical and physical data models that represent trajectory datasets and other relevant data required by SAMS to support pedestrian region-based navigation;
Apart from the above objectives, we are also interested in designing a database which can be used not only for managing data sets specific to pedestrian region-based navigation context, but also which can be used for managing data associated with other location management applications domains. Therefore, we want our database to be generic for location management applications for which the user trajectory management and the storage of environment data in a database are fundamentals. The result of this research is of interest for any application dealing with management of trajectory data as it provides suitable data storage structures.

1.2.2 Research questions

Two main research questions and their sub questions, require answers in order to meet our objectives:

1. What are the data about pedestrians' physical world required by SAMS in order to support pedestrian region-based navigation in urban areas? And how to model this data?

2. How to design and implement a database that can manage trajectory data and other relevant data associated with pedestrian region-based navigation and answer relevant types of queries?
   - How to design a conceptual data model and a logical model?
   - How to design the physical model?
   - How to effectively answer queries associated with services needed by pedestrians?

1.2.3 Innovation aimed at

The innovation aimed by this research is “to define data models suitable for database representation of pedestrian trajectory data and other relevant data associated with pedestrian region-based navigation”. The defined data models will then be used for the design and implementation of a database that can support spatially-aware map services that assist pedestrians.

1.3 Method adopted

The method adopted for this research are based on method proposed by [EC02] for the design of object-relational database and on method proposed by [Van07] for the design of information systems. The two methods combined provide four main phases in which we will follow during this research; figure 1.1 shows the four main phases which are analysis, design, implementation and evaluation:
1.3. Method adopted

**Analysis phase:**

In this phase the following issues will be addressed:

1. Spatial and spatio-temporal information required by SAMS in order to support pedestrian region-based navigation application are identified;
2. An information model describing relevant information is built using UML class diagrams as it is the standard modelling language for object oriented systems design.

**Design phase:**

The design phase is done in three steps; the first step is the definition of a data model, the second step is the design of a conceptual model and the third step is the design of a logical model.

1. **A data model definition:**
   (a) Data types required for database representation of spatial and spatio-temporal aspects is defined;
   (b) Operations required for computation on data are also defined
2. **Conceptual model design**
   (a) Spatial and spatio-temporal properties of the objects are modelled as objects classes;
   (b) Associations between objects classes are defined;
   (c) The conceptual schema is designed by using UML class diagrams.
3. **Logical design**
   (a) A logical schema is specified using SQL of PostGreSQL as the DBMS chosen for our research.

**Implementation phase**

The implementation phase concerns the physical design and the following points are addressed:

1. Data storage structures are created;
2. A testing dataset is prepared and loaded into the database;

**Evaluation phase**

In the evaluation phase, relevant queries are formulated and executed in the database.
Chapter 1. Introduction

Figure 1.1: Method adopted for the research
1.4 Thesis outline

Based on the method adopted, this research is organized in six chapters as below:

Chapter 1 discusses the motivation behind this research, states the problem, identifies research objectives and questions and presents the method adopted for this research.

Chapter 2 gives related research in the field of pedestrian navigation systems as well as in the field of moving objects databases.

Chapter 3 makes an analysis of information required to support pedestrian region-based navigation. Based on the results of the analysis, an information model is also built in this chapter.

Chapter 4 defines a data model suitable for a database representation of the information model designed in chapter three. Based on this data model, a conceptual schema and a logical schema are designed in this chapter.

Chapter 5 considers the implementation part of the research. The designed logical schema is transformed into a physical schema that defines storage structure and a testing dataset is loaded into the database.

Chapter 6 discusses the result of the research, gives a conclusion and some recommendations for future works.
Chapter 2

Related research

This chapter introduces related research in the fields of pedestrian navigation systems and moving objects databases. For pedestrian navigation systems, research done on pedestrian behaviour and requirements are presented in section 2.1 and for moving objects databases, research done on modelling and querying moving objects are presented in section 2.2.

2.1 Pedestrian navigation systems

With the increase of communication portable devices and the rapid development of information and communication technologies, it is possible to assist pedestrians by providing them with reliable guiding instructions and additional location-based information. Although car navigation systems already exist and are routinely used, they cannot provide optimal route instructions to pedestrians because route instructions provided by car navigation systems are linked to road networks whereas pedestrians do not necessarily move on road networks. The increasing interests of pedestrians in getting route instructions and other information about their surroundings especially in unfamiliar areas or complex environments (e.g. urban areas), gave result to a number of research. Research that has been carried out focused on development of pedestrian navigation systems and they can be categorized in three main research areas: 1) defining pedestrian behaviour, 2) pedestrian requirements and 3) spatially-aware map service

2.1.1 Pedestrian behaviour

Providing optimal route instructions to pedestrians requires the knowledge of pedestrian spatio-temporal behaviour. Within the last years, a number of studies have been done to understand pedestrian behaviour in choosing a route to a desired destination. Their findings reveal that human route choice relies on a huge variety of influence factors [MG07b]; among these factors are personal
2.1. Pedestrian navigation systems

characteristics, characteristics of the trip, properties of the infrastructures and environments characteristics.

Furthermore, it has been pointed out that pedestrians often have a preference to route based on arbitrary factors rather than just shortness or fastest as it is the case for cars. e.g. [MS05b] revealed that pedestrians prefer “most beautiful”, “most convenient” or “safest routes” and [Gol95] found that “fewest turns”, “least time”, “most aesthetic” or “novelty” are other factors that influence pedestrian route decisions. Additionally, [MG07a] specifies that apart from physical route factors like “shortness” or “accessibility for handicapped people” and emotional route factors like “attractiveness”, mental route factors also play a significant role in choosing a path in an unfamiliar environment. All these results show that pedestrian walking behaviour and the choice of a specific route depend on a combination factors that [MG07a] resumes on four main factors: physical route factors, emotional route factors, lifestyle and cognitive route factors. (see figure 2.1)

![Diagram of Factors influencing route choice behaviour. Source (MG07a)](image)

Even though the above research identifies the factors influencing pedestrian route choices, it does not identify specific requirements and interests of certain groups of pedestrians (e.g. tourists) [GCP07]. Therefore, it is clear that pedestrian navigation systems should be based on a deep understanding of pedestrian spatio-temporal behaviour and requirements in order to provide optimal routes to different kinds of persons moving in different contexts.

For a deep understanding of pedestrian behaviour, appropriate methods for the acquisition and assessment of pedestrian spatio-temporal behaviour are required. A number of different methods have been used before, like questionnaire surveys, trip diaries, direct observations, video-based analysis and localisation technologies; however, they presented some advantages and drawbacks; e.g. methods which were based on study and explanation of visible behaviour did not succeed to reveal pedestrian behaviour as they were just based on their visible activities. Table 2.1 summarizes the advantages and the drawbacks of the methods used.
Table 2.1: Empirical methods in pedestrian monitoring (MG07a)

<table>
<thead>
<tr>
<th>Method</th>
<th>Data</th>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Questionnaire</td>
<td>Decision processes, individual habits, motives, intentions, lifestyle</td>
<td>• Low costs</td>
<td>• Inaccuracy</td>
</tr>
<tr>
<td>surveys</td>
<td>attributes</td>
<td>• Large samples</td>
<td></td>
</tr>
<tr>
<td>Trip diaries</td>
<td>Decision processes, individual habits, motives, intentions, lifestyle</td>
<td>• Detailed information</td>
<td>• Dependant on participant’s memory</td>
</tr>
<tr>
<td></td>
<td>attributes</td>
<td></td>
<td>• Varying quality</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Small samples</td>
</tr>
<tr>
<td>Direct observation</td>
<td>Visible activities, routes</td>
<td>• Detailed information</td>
<td>• Time-consuming</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• natural behaviour</td>
<td>• Labour-intensive</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Observer effects</td>
</tr>
<tr>
<td>Video-based analysis</td>
<td>Visible activities, routes</td>
<td>• Large samples</td>
<td>• Small observation field</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Detailed information</td>
<td>• Cost-intensive</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Localisation</td>
<td>Location data, routes</td>
<td>• Large observation field</td>
<td>• Observer effects</td>
</tr>
<tr>
<td>technologies</td>
<td></td>
<td></td>
<td>• Cost-intensive</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Inaccuracy</td>
</tr>
</tbody>
</table>
2.1. Pedestrian navigation systems

Due to the drawbacks of the mentioned methods and to their inability to provide a deep understanding of pedestrian behaviour, [MG07a] proposes the use of a combination of various empirical methods to study human behaviour. [MG07a] states that a comprehensive insight into route decision processes and fundamental influence factors can be identified by determining and extracting discriminative features for different classes of spatio-temporal behaviour. Those features can then be provided by a user to an implemented wayfinding system which will consequently deliver customized information. [MG07a] is a research which is still ongoing that aims at providing a more insightful understanding of pedestrian walking patterns.

Currently, preliminary results allow extracting some features that include velocities, frequency, duration, location of stops and significant changes in direction and they indicate also that a number of homogeneous behaviour patterns are being observed, especially in consistent context situations. It is expected that the result of [MG07a] will be a final model of pedestrian mobility styles which can be used in future mobile navigation services to provide customized route suggestions and location based information.

2.1.2 Pedestrian information requirements

The development of pedestrian navigation systems requires more than understanding walking behaviour of pedestrians. It necessitates also the identification of pedestrian information requirements while navigating from an origin point to a destination point. [ATSM03] argues that a key prerequisite to designing successful pedestrian services and delivering these over mobile devices is to understand the nature of the navigation task and the information requirements of pedestrians. Having specified pedestrian requirements, designers can then design an information product that satisfies these needs.

As pedestrians need guidance support especially in complex environments such as urban areas, several studies have been conducted for urban navigation context. Their results show that landmarks are by far the most predominant navigation indicator followed by the distance information and street names which were infrequently used [ATSM03, MS05a, TD03, LHM99]. Another result of [ATSM03] is that pedestrians are not simply directed from one navigation decision point to another, but they also require information between those decision points in order to maintain their trust in the information source and their confidence and orientation throughout the route. However, questions can arise from what are the features that these studies qualify as being landmarks. [MS05a] defines landmarks as “stationary, distinct and salient objects or places, which serve as cues for structuring and building a mental representation of the surrounding area; any object can be perceived as a landmark if it is unique enough in comparison to the adjacent items”. As landmarks are primary means for the communication of route direction to pedestrians, their use is highly recommended at decision points where there is a need of a re-orientation or they can be used as route marks as confirmation for being on the right way.
However, designing successful pedestrian navigation systems based on the theory of using landmarks for route instruction rather than street names or distance, is a lot of work because these objects qualified as landmarks must be contained in a database in order to be used for route instructions. This is a problem because currently geographical datasets do not contain these landmarks that are relevant to pedestrians.

Therefore, for route selection and route communication appropriate for pedestrian requirements, a dataset with the required spatial granularity is needed. The content of this dataset must be quite different from available spatial datasets as it must contain features that are required to compute a path from a start point to a destination point and to portray it on a map-like representation [Bir07].

By developing a dataset and enrich it with relevant information for pedestrians like footpaths, pedestrians bridges, access paths to buildings, path within multi-story buildings, pedestrians zones or generally accessible areas (customized to pedestrians needs), the dataset can be used for automatic generation of appropriate routes, directions and maps for pedestrian navigation systems [Bir07].

According to [Bir07], the development of a dataset for pedestrian navigation systems requires:

- Necessary data sources (e.g. topographic map data, real estate cadastral map data and indoor map data);
- The exploration of the given data and extraction rules are set up to choose all pedestrian relevant data (data selection);
- GIS analysis techniques to derive new information from the new data
- The different data parts have to be merged into a single, geometric consistent dataset using conflation algorithms to establish the necessary connectivity between all information parts (geometric integration)

The basic idea is to use different existing spatial databases and extract the relevant information parts for pedestrians. For example, in order to make an indoor and outdoor direction-finding representation, a map that represents paths inside buildings (like train station or shopping centers) can also be integrated.

Figure 2.2 shows an example of how different datasets can be combined to provide one dataset suitable for pedestrian navigation systems [Bir07].

As our research concerns database support for pedestrians, the identification of their requirements is crucial because all their requirements must be represented in the database in order to be accessed by pedestrian queries while searching for information about their environment.
2.1.3 Spatially-aware map services

As many research has focused on defining pedestrian behaviour and requirements, other researchers have been working on how to design products which track specific types of pedestrians and provide customized spatial information to them. It is in this context that a number of spatially-aware map services (SAMS) applications which locate mobile users and allow them to access spatial services, anytime and anywhere, have emerged. These services are called spatially-aware map services because they are aware of the types of user or application task and the preferences of the user or constraints to the application [LPM08].

An example of a SAMS developed for tourist, is the GUIDE which is an electronic intelligent tourist guide which provide to the Lancaster city visitors, information tailored to both their personal context (e.g. personal interest: history or architecture) and environment context (e.g. opening time of the attractions) [CDM+00]. The GUIDE system provide to its users, an interface from which
they can personalize the service they want to get from it; among the task users can perform are information retrieval, navigation of the city using a map, creating and then following a tour of the city and communicating with other visitors. However, even though, GUIDE is a helpful tool for tourists visiting the city of Lancaster, it is limited only for pedestrians in tourism navigation context. Therefore, it can not provide optimal services to other pedestrians navigating in different context than tourism.

However, even though SAMS can be customized for different navigation context, they have four common typical components [LPM08]:

1. A wireless networked access mobile devices,
2. A geographic information system (GIS),
3. A location determination system such as a satellite based global positioning system.
4. A data repository (i.e. database)

Therefore, as the fourth component is a data repository, it is obvious that SAMS can deliver customized information once it has all the necessary data stored in the database and with appropriate access methods.

2.1.4 Region-based navigation

As revealed by many research done on pedestrian behaviour, pedestrian navigation is different from car navigation. This difference in navigation is mainly due to the fact that pedestrians are not tied to road network as it is the case for cars. As the cars are always constrained on roads network, pedestrians are free to use network systems such as streets, walkways but they may also use unconstrained areas that lack predefined paths such as parks, city squares, train stations etc. Therefore, in order to provide optimal navigation assistance to pedestrians, pedestrian navigation systems must be based on new approaches of generating navigation routes.

It is in this context that [WC06] proposed a theoretical foundation for the generation of route instructions which is based on a region-based definition of routes, that defines routes as a sequence of regions. In order to define a relation between a navigator and a defined path, this theoretical framework defines topological stages of closeness between a circular spatially extended point (CSEP) representing the navigator and other CSEPs representing waypoints that compose a route. It is therefore based on how the CSEP representing the navigator is close to the CSEPs representing waypoints of a route, that the system will generate suitable instructions.

The CSEP around the navigator are semantically different from the ones describing the waypoints of a path. The CSEP around the navigator consists of
a position derived from a GPS and a circular extent given by the positional inaccuracy, whereas the CSEP around waypoints describe a decision area around waypoints within which pedestrians can move without requiring new instructions. Even though this research provides an interesting approach for providing instructions to pedestrians based on stages of closeness between an area around a pedestrian and a predefined route, it does not consider how this route is generated and how it is suitable for pedestrians.

2.2 Moving objects databases

Moving objects database is a research area that focuses on how to represent moving entities in a database and allow asking queries about their movements. Moving objects can be classified in two categories [GS05]; the first category includes moving objects for which only their time-dependent position is relevant; among these objects are pedestrians, cars, animals, aircrafts and they are typified as moving points. The second category concerns objects for which not only their time-dependent position matters, but also their spatial extents; among them are forest fires, hurricanes, epidemic diseases and they are typified as moving regions. Moving point and moving region objects have in common their time-dependent geometries and the fact that they are termed as moving objects specify that their geometries change continuously [GS05].

2.2.1 Modeling moving objects

Recently, the availability of wireless network and position-aware devices such as personal digital assistants (PDA), on-board units in vehicles or mobile phones, lead to the increase of applications dealing with moving objects. This increase gave result to a huge volume of movement data (i.e. trajectories) which needed to be managed and analysed in database systems. However, database support for this type of data required a set of critical functionalities to be integrated and built on top of traditional database management systems [WSCY99]. It is for this reason that researchers have been working on how to represent this movement data in databases.

Research done on representing moving objects in databases can be looked at in two perspectives: a spatio-temporal perspective and a location management perspective [GS05]. For the spatio-temporal perspective, the research interest was on how to store in a database histories of movement and be able to ask queries for any time in the past. For the location management perspective, the research interest was on how to manage in a database the continuously changing information about the current positions and be able to predict the near future positions of moving objects.

Spatio-temporal perspective:
In this perspective, research interest was in the storage and querying of histories of moving objects in a database. The main research contribution in this perspective was done during a European union funded project called CHOROCHRONOS where European researchers working on spatial and temporal database got together to achieve an understanding of each other's research and put together their results and methodologies. It is during this project that Guting et al. provided a rich algebra with data types and operations that can be embedded into a database management system (DBMS) query language to extend it and obtain a query language for spatio-temporal data [GME+00].

In order to represent moving points and moving regions in database, Guting et al. propose to consider these objects as three dimensional (2 dimensional space + time) objects whose structure and behaviours can be captured by modeling them as abstract data types. These abstract data types can then be integrated as attribute types into any DBMS data model like relational model, object-oriented model or any other. Their embedment in a DBMS query language allows obtaining a query language for spatio-temporal data and particularly for moving objects. These abstract data types are defined together with appropriate operations that indicate all possible state of moving objects. These abstract types together with associated operations provide an accurate and conceptually powerful foundation for the representation and querying of spatio-temporal data [GME+00].

Data structures for the types and algorithms implementing the operations are needed in order to implement them in an extensible DBMS architecture which has interfaces for registering components such as data structures for the types and algorithms for the data types and operations. Such algebra can be added as datablade, cartridge or as an extender [GS05]. The above data model developed by Guting is limited for the history of movement.

However, in the CHOROCHRONOS project, the approach based on data types for modeling moving objects proposed by Guting was not the only one. Grumbach et al. [GKR+03] proposed a different approach based on constraints for modeling moving objects. The main idea of this approach is to represent temporal and spatial objects as infinite sets of points satisfying first-order formulae [GKR+03]; e.g. a polygon on the plane is seen as infinite sets of points of Q2 inside its boundary. The problem with infinite sets is that it is not possible to store and manipulate them in computer; for this, Grumbach et al. propose to describe and manipulate them through finite representations satisfying first-order by using linear constraints.

According to Grumbach et al. [GKR+03], many spatial, temporal and spatio-temporal database works have been based on the approach of extending conventional data models with abstract data types and operations, which is also the case for the work done by Guting et al. [GME+00]. For Grumbach, this approach leads to lacks of uniformity and he proposes a different approach based on constraints which represent spatio-temporal data in a unified framework and supports declarative query languages. According to Grumbach, the
benefit of this approach is that it allows to uniformly representing infinite and indefinite information.

**Location management perspective**

In this perspective, the interest is on how to manage in a database the locations of a set of mobile objects that are currently moving and be able to retrieve their current positions and to predict their near future positions. The main contribution of research treating moving objects for the location management perspective was done by [WXCJ98] in a project called DOMINO (Database fOr MovINg Objects). To solve the problem of continuously changing location of moving objects, they propose to represent the location of a moving object as a function of time that changes without an explicit update as it is the case for traditional model. This function of time is described by an object’s motion vector which is represented as an attribute of the object. Even though the motion vector of a moving object changes with time, its update is done less frequently comparing to update of a simple location of a moving object.

It is in this context that [WXCJ98] introduced a data model called MOST (Moving Objects Spatio-Temporal) suitable for describing and querying current movement as well as expected near future movement. The MOST data model initiated for the first time the concept of a dynamic attributes; these dynamic attributes are attributes that changes continuously their values with time without being explicitly updated. They are therefore in contrast with traditional static attributes that require an explicit update in order to change. Even though static attributes are different from dynamic attributes for the point of view of their updates, they are both represented by the same data types (e.g. point) and queries accessing dynamic attributes are formulated as if they refer to static attributes. However, due to the continuously change of the value of the dynamic attribute with time, a similar query posed at different times will give different results.

Normally, a dynamic attribute A as described by [WXCJ98] is represented by three sub-attributes as follows:

1. $A.updatevalue$, $A.updatetime$ and $A.function$ where $A.function$ is a function of a single variable $t$ that has value 0 at $t = 0$.

The value of a dynamic attribute depends on the time is defined by [18] as follows:

1. At time $A.updatetime$ the value of A at time $A.updatevalue$, and until the next update value of A the value of A at time $A.updatetime + t0$ (where $t0$ is a positive number) is given by $A.updatetime + A.function(t0)$.

For objects moving in two dimensional space, [WXCJ98] proposes to model its location attribute $L$ by two dynamics attributes $L.x$ and $L.y$, each with its
own update value, update time and function representing the x and y coordinates of the object respectively. This concept can also be extended to motion in three dimensional space by adding a third dynamic attribute with its corresponding update value, update time and function. For objects moving along a predefined route, the MOST data model proposes represent the dynamic attribute with five sub-attributes namely: $L.route$, $L.x.updatevalue$, $L.y.updatevalue$, $L.updatetime$ and $L.speed$ where $L.route$ is a line spatial object specifying the route on which the objects is moving; $L.x.updatevalue$ and $L.y.updatevalue$ are the x and y of the position of the object on the $L.route$ at time $L.updatetime$ and the $L.speed$ is a linear function of the form $f(t) = b*t$ speed of the moving object. This function gives the current distance from the starting location as a function of the time since $L.updatetime$.

In connection with our research that aims at designing a database for supporting pedestrian region-based navigation, our research will consider pedestrians movement at the location management perspective rather than at the spatio-temporal perspective. This is due to the fact that supporting pedestrian region-based navigation requires the management of information about their current positions rather than the storage of histories of their movement. Therefore, our research will be based on research carried out for the management and querying current locations of moving objects.

### 2.2.2 Querying and indexing moving objects

Once the DBMS has been extended to support the defined data types and operations, access methods and query processing techniques have also to be defined as they are two fundamental issues that the performance of the DBMS depends on [PFJ+03]. If the DBMS has to support the spatio-temporal applications, new access methods and query processing techniques are needed rather than the ones that traditionally relational DBMS have relied on.

While the access methods and query processing techniques for spatial data exist already, the ones that have been proposed for spatio-temporal data are mainly their extensions. It is again in the CHOROCHRONOS project that Di Pasquale et al. [TVS96] proposed access methods and query processing techniques which can be divided in two categories depending on the spatial method they extend; these are R-tree-based methods and quadtree-based methods. The latter extends the R-trees which are tree data structures used for spatial data that split the space by using minimum bounding rectangles (MBR) and the former extends quadtrees which are tree data structures that recursively divide the space into four quadrants.

For R-tree based methods, the basic idea for indexing spatio-temporal data is to consider these data as three dimensional data composed by two spatial dimensions on which an additional spatial dimension representing time have been added. Therefore a two dimensional rectangle $(x_1, y_1, x_2, y_2)$ with an associated time interval $[t_1, t_2)$ is viewed as a three dimensional box $(x_1, y_1, x_2, y_2, t_1, t_2)$ [TVS96]. However, this approach of considering time as another
2.3 Conclusion

dimension has a disadvantage of excessive dead space [TVS96] which can be overcome by the overlapping techniques proposed by [BHK85]. Therefore, most of the proposed access methods in the CHOROCHRONOS project were based on two approaches: the first approach which considers time as an extra dimension and an approach based on overlapping trees.

Among the proposed methods are 3D R-Tree, 2+3 R-Tree which are both based on the addition of time as an extra dimension approach and the HR-Tree which is based on overlapping technique. The 3D R-Tree is an index method that considers both ends of the interval \([t_1, t_2)\) of each rectangle to be known and fixed [PFJ +03]; therefore, for open or expanding objects, this 3D R-Tree can not handle such open objects. The 2+3 R-Tree is an approach that overcome the problem of open objects by combining 2D R-Tree and 3D R-Tree [PFJ +03]; hence its name. The HR-Tree stands for Historical R-Tree which is an approach based on overlapping technique that creates a new R-tree each time an update occurs; this approach is efficient whenever the number of objects changing position in space is relatively small, however, if this number is becoming large this approach is not efficient as it produces independent tree structures without common paths [PFJ +03].

2.3 Conclusion

Related research presented in this chapter can be summarized by the figure 2.3

Related research presented in this chapter has root in two main research areas: pedestrian navigation systems and moving objects database. For pedestrian navigation systems, research carried out on pedestrians' behaviors, their information requirements and on spatially aware map services that support pedestrian navigation have been introduced. For moving objects databases re-
search, modeling of these objects has been presented in two perspectives: location management and spatio-temporal perspectives. Finally, research done on database querying of moving objects have also been presented.
2.3. Conclusion
Chapter 3

Information model

The objective of this chapter is to assess information required by a spatially-aware map service (SAMS) for supporting pedestrian region-based navigation in urban areas and to build a model that describes this information. The assessment of this information is important for this research as it will reveal information to be represented in the database that our research aims to build. To obtain this information, a scenario that represents a pedestrian assisted by a SAMS application running on a mobile phone is sketched in section 3.2 and analysed in section 3.3 to extract relevant information to be represented in the database. Finally, a model describing the application relevant information is built in section 3.4.

3.1 Introduction

While navigating in urban areas especially in unfamiliar areas, pedestrians are in need of not only navigating assistance from an origin point $A$ to a destination point $B$, but also they are in need of a variety of information about their environment. Information required varies according to individual’s requirements and specific context in which the individual is moving; e.g. a tourist interested in history of a place will require different information compared to someone interested in shopping. For this reason, a SAMS will satisfy pedestrians’ needs if it is able to track its users and provide them contextualised information related to their environment; this imply the SAMS to be aware of the user’s spatial context and to have all necessary data related to their needs.

Due to the variety of services required by pedestrians particularly in urban areas, there is a need to assess what kind of data to provide to a SAMS in order to provide these services to pedestrians. Data required by SAMS has to be stored in a well structured database which will ensure well organisation of this data, their maintenance and their security. However, as the SAMS deals with continuously changing position of pedestrians, the database will also be responsible of the management of the time-dependent positions of pedestrians in order to link their directional positions with respective information about
3.2. Application scenario

Robert is coming for the first time to the city of Enschede which is a well known city for its mixed shops, green areas, monuments and beautiful buildings. He has a number of things to do in this city, shopping and visit the well known their environment. The database that will manage pedestrians’ positions as well as geographical data about pedestrians’ environment has to be based on data models that allow modelling those time-dependent positions of pedestrians and their view on the physical world (urban area). Therefore, there is a need to assess relevant data accustomed to needs of pedestrians for region-based navigation systems in urban areas and build suitable data models.

Although the data required by SAMS depends on the types of pedestrians and on the context in which they are travelling, it is possible to find common data that pedestrians need. Based on objects that are used in human wayfinding in the general context of pedestrian navigation and based on information asked by pedestrians in urban areas, relevant spatial features together with their associated attributes, as well as the relationships between these data, can be determined. The determination of these data will facilitate the design of data models that will allow the design of a database suitable for supporting pedestrian region-based navigation systems.

To get a better insight of what kind of data pedestrians require about their surroundings, an application scenario is sketched below. The scenario represents a pedestrian moving in an unfamiliar city assisted with a SAMS application running on his mobile phone; we assume that the design of this SAMS was driven by pedestrian needs in an urban environment. Consequently, the application is able to track pedestrians’ directional positions and satisfy their needs. The main idea behind the scenario is that, if a SAMS can track pedestrians and satisfies all their needs in a certain urban area, it is possible to assess what kind of services pedestrians asked for and what kind of data SAMS require at their disposal (from a database) in order support pedestrians.

For that reason, it is critical to analyse from both points of view, users’ view and providers’ view, what types of information pedestrians ask for and what types of data the database should contain respectively. Therefore, from a critical analysis of the scenario, it is possible to build an information model that represents spatial, spatio-temporal and non-spatial data relevant for SAMS that can assist pedestrians in urban navigation. Based on this information model, data models can be designed that will guarantee efficient storage structure of the relevant data and management of pedestrian trajectories.

As our research concerns database support for SAMS, our task go beyond the efficient storage of data and management of pedestrian trajectories and consider also the access of these data; therefore, we will also analyse type of queries that pedestrians asked to the SAMS in order to identify queries that the database is supposed to support.
UTC building before he leaves back to his home place in Oldenzaal 10 kilometers away from the Enschede city. As he walked out of the train station, he launches a pedestrian navigation application on his mobile phone and the application asked him to put a destination, he chooses first to go to the “Hello mall” which is business center that a friend recommended him; the result shows a building and he clicks on it to see associated options to it, he then select “go there” option; immediately his phone proposed him to choose among three routes: “shortest”, “fewest turn” and “most scenic”, he chooses the most scenic then he clicks on the start link and his phone shows him a digital map on top of which there are, a series of connected segments together forming the route to follow, a point showing his current position on the map and an arrow pointing to the direction to take. The phone continuously tracks his position and the arrow continues to show the direction to take; after two minutes, the phone notifies him with a vibration and a voice that in about 4m, his going to turn on the right after the “red house” to take the “Helmastraat”.

As he continues walking on helmastraat, his phone notifies him again that on his right there is a museum and that the tall building that he is seeing about 100meters in front of him is called “Hope building”. Ten minutes later, his phone notifies him again that he is reaching the “Hello mall” and the arrow points to the entrance of the mall. Inside the mall, the phone notifies him with a list of shops that are inside the mall; he chooses to be directed to the nearest shoes shop, the phone directs him again using voice and the arrow to turn on left or right inside the shop center till he arrives to the shoes shop. From the shoes shop, he launches again the navigation application to take him to the nearest clothes shop from the shoes shop, and then the phone shows him again a direction to follow. After buying the clothes, he decides to go out of the mall and the phone directs him to the exit of the mall. At the exit of the hello mall, he chooses to be directed to the UTC building, but as he did not take lunch, he asked the phone to notify him if there is a fast-food restaurant in less than 5 meters on his way going to the UTC building.

The phone showed him the direction to follow, after three minutes his phone notifies him that he will reach a Mc Donald’s in three minutes. After exactly three minutes, the phone notifies him that there is a Mc Donald’s inside the white building which on his right; he then clicks on pause of his phone he then enters inside to buy food; from there he resumes the route to the UTC building. While walking, his phone alerts him that they are going to pass through a park, the phone is showing him to the entrance of the park, and he then received an instruction “walk towards the monument in about 200 meters in front of you”. Reaching the monument, he received another instruction “turn on the right and walk towards the clock tower by keeping the Rutbeek river on your right”; reaching the clock tower, the phone directs him to the exit of the park which was near the clock tower. At the exit the phone notifies him that he is now on the “vrienstraat” at which the UTC building is; the phone notifies him that he has to go straight in just 150 meters on his right he will see a tall blue building with then name UTC on top of it. After enjoying the nice-looking building, he requested to search for any electronic shop that may be within 200 meters
from his location; the navigation application found two electronics within this
distance and he chose to be directed to one of them. As it was late when he left
the electronic shop, he decided to go back home and the phone directs him back
to the train station.

3.3 Information requirement analysis

The information required by a SAMS in order to satisfy pedestrians’ needs
can be found by analysing the above scenario. As the information required
at both the user level and service provider level are complementary, there is
a need to make an analysis at both levels and come up with a complete informa-
tion model. Such information model can be exploited by database designer
while building databases that support SAMS for pedestrian navigation in ur-
ban areas. As SAMS is a location-based service and as it deals with pedestrians
changing their positions continuously, this requirements analysis will be based
on method proposed by [PT98] for the evaluation of requirements for spatio-
temporal applications; figure 3.1 summarizes the method adopted for the anal-
ysis:

As figure 3.1 shows, our analysis will be guided by the following questions:

1. What are the functions that a SAMS system is supposed to perform?

2. What are the spatial, spatio-temporal and non-spatial data required by
SAMS in order to support pedestrian region-based navigation?

3. What are the relationships between these data?

4. What are the types of queries asked by pedestrians?

3.3.1 Analysis of required function

As supporting pedestrian region-based navigation requires the SAMS to per-
form a certain number of functions, it is reasonable to first analyse the func-
tions the SAMS is supposed to perform. The idea behind this analysis is that,
one the functions are known then a methodical analysis of data requirement
can be carried out for each function and combined to make a complete data
requirement for the whole system.

Since we are dealing with pedestrian navigation, the first function that
comes directly in our mind is the function that calculates a route from a cer-
tain origin point to a pre-specified destination. However, there are of course
other functions performed, as for example it does not make any sense that the
system can generate a route and then provide to users without giving them
instructions of how to follow it; therefore a route guidance function may be re-
quired also to give structured and precise instructions to follow the calculated
route. Based on [CW05] for guidance of cars and pedestrians by navigation
Figure 3.1: Analysis of the application scenario
3.3. Information requirement analysis

systems and based also on the above application scenario, four main function to be performed are the following:

- **Directional Positioning**: supporting pedestrians requires the SAMS to know their location or other information about their motion such as direction. Positioning functions are responsible for tracking pedestrians and measure the position, the speed and direction or other users’ motion related information from the signals provided by positioning systems like GPS or other sensing devices.

- **Route generation**: the route generation function is responsible for the calculation of a route from the location of the user provided by the positioning system, to a destination pre-specified by the user.

- **Route guidance**: once the route to follow is generated, the route guidance function gives detailed instructions along the calculated route to reach the desired destination.

- **Map matching and display**: this function is responsible to link the given position of the user to the right navigable road element and to display them on a digital map through a mobile device such as a mobile phone or a PDA (Personal Digital Assistance).

The above four functions are not the only possible functions that can be performed by a SAMS, other functions can be included in a SAMS depending on the services that SAMS is designed for; however, in any system that supports navigation the above functions are basics and indispensables. In harmony with the description of the above functions, each function has its specific requirements in terms of data; required data range from spatial data (with fixed position in space) to spatio-temporal data (with time-changing position) as well as to non-spatial data deemed relevant to pedestrian region-based navigation. For this reason, the analysis of data requirement is carried out below systematically for each function.

3.3.2 Analysis of relevant data

To ensure a complete and consistent analysis of required data, this analysis is based on the aforementioned functions that the SAMS is supposed to perform and it determines data required or produced for each function, that need to be represented in the database. This database will operate as a data repository that the SAMS can access anytime and retrieve data required by its users.

1. **Data generated by the directional positioning functions**: Like other location-based services, the SAMS require a reliable, accurate and continuous directional position determination of its mobile users in order to link their location with surrounding data. The positioning function tracks the continuously changing position of pedestrians and provides
data related to their motion that has to be used by the remaining functions. As route generation, route guidance as well as the map matching functions require updated data about the location of the user in order to ensure reliable and accurate services, it is crucial to manage the time dependent directional positions and other relevant information related to pedestrian motion in a database. For this reason, the assessment of information about pedestrian motion required to manage in the database is important; this assessment is done by analysing the application scenario and by finding this relevant time-dependent information to be managed in the database.

To get a better insight of information to represent about pedestrian motion, the following questions related to the above scenario are put forward:

- Which information about pedestrian motion does the system needs in order to calculate a route?
- Which information about pedestrian motion does the system needs to give accurate instructions such as “walk towards the monument in about 200 meters in front of you” or “you will reach a McDonald’s in three minutes”?

From the first question, it is obvious that the system needs the actual position of users to be able to calculate the route to a desired destination. The actual position of users may be of geographic coordinates or any identification of the position as a starting location. By knowing the actual position of a pedestrian, the system is also able to link it with other surrounding information required or that may be of help to the user.

From the second question, it is clear that user’s position is not the only information about pedestrian motion that the system requires. By analysing the instruction “walk towards the monument in about 200 meters in front of you” it is wise to state that this instruction requires that the system is aware of the pedestrian position in order to know the exact distance between him and the monument. However, how the system knows that the monument is in front of him? That requires the system to be aware of additional information which is the direction of the pedestrian at that moment.

Another question rises also from the instruction “you will reach a McDonald’s in three minutes”; the question concerns the information about pedestrian motion that the system is aware of so that it can calculate exactly the time that Robert will use to reach the McDonald’s restaurant. As the system is already aware of the direction of the pedestrian as well as of its exact position, it is able to determine the distance between the position of a pedestrian and the position of a certain place in space. However, this distance is not sufficient to allow the system to predict the time the pedestrian will use; therefore there is an additional information required by the system. According to the physics relation between distance, velocity and time which states that the time is equal to the distance travelled...
divided by the speed, it is clear that the other information that the system requires about pedestrian motion in order to predict the time it will take to a user to reach a certain place is his speed.

In summary, the SAMS will provide services to pedestrian such as route calculation from his position to a desired location as well as route guidance in terms of distance or time, once the system is aware of the following information about pedestrian motion:

- pedestrian position in the space
- pedestrian direction
- pedestrian speed

Due to the continuously change of the above information, managing them in a database will ensure their correct updates and allow the system to access this information whenever it needs.

2. **Data required for route calculation:**

Among navigating functions that the SAMS running on the Robert’s phone is able to provide him, is route calculation between locations in a geographic region. By specifying his current location or optionally from equipment that can determine his physical location (e.g. GPS), the route calculation function examines possible routes between two locations and determines the optimal one to follow from the starting point to a desired destination within a certain geographic area. The route calculated is presented to Robert as a digital map layered with a series of connected route segments over which he can travel from his actual location to the destination location.

Apart from the starting location and the destination location that the route calculation function requires in order to calculate a route, the function requires also other additional data. Data required by the route calculation function can be found by analysing the above scenario according to the following two questions:

- What are the spatial features required by the route calculation function?
- Which information is required about these spatial features?

To answer the above questions, an analysis of the scenario in terms of data requirements for route calculation is conducted. Starting from the first time that Robert requested a route to the “hello mall”, the system asked him to choose between three types of route: the shortest, the fewest turn or the most scenic routes; these three choices show clearly that the route calculation function has many solution routes between the starting location to the destination location. This means, the function has access to data that qualify a route as the shortest or with fewest turn. Therefore, the route calculation does not require only routes network within
the geographic region that pedestrians are moving inside, but also it requires attributes data associated with those routes in order to account for walking preferences.

According to the scenario, some portions of the route calculated by the system correspond to the normal streets network that cars usually follow such as “Helmastraat” or “Vrienstraat”, and other portions do not correspond with the streets network; this can be noticed when Robert was directed through a park which lacks underlying networked paths. Therefore, routes generated for pedestrians are composed of a series of connected pedestrians’ segments; some pedestrians’ segments correspond to the normal street segments such as “Helmastraat” or “Vrienstraat” whereas others are typical for pedestrians. With these pedestrians’ segments, the route calculation function is able to find optimal routes for them as it does not constrain them on streets network only. As it is the case for car navigation systems which require roads network in order to calculate routes for cars, we argue that the route calculation function for pedestrians requires also pedestrians’ navigable segments network.

In summary, data required for routes calculation that suit pedestrians are:

- Pedestrian navigable segments network of the geographical region surrounding his location and the destination location.
- Attributes data associated with these segments that enable the route calculation function to take into account information about walking preferences like shortest, fewest turn and scenic, paved, etc.

Through the database that store the above data, the route calculation function is able to generate one or more solution routes and provide them to the user who will chose one according to his needs.

3. Data required for route guidance:

Once a route to a desired destination location has been calculated, pedestrians are also in need of direction instructions to reach their destination; this is done by the route guidance function. The route guidance function generates pedestrian guidance messages containing adequate information to guide pedestrian along the route calculated by the route calculation function in order to avoid pedestrian confusion about orientation. For supporting Robert navigation in the above scenario, the SAMS running on his phone uses a set of features represented graphically on a digital map to give him guidance instructions; we want to assess data that the route guidance function requires. Their assessment is important because this data has to be stored in a database in a way that allows the function to access them.

The assessment will be based on the feature classes that are contained in that map together with their associated information and on additional information that the application used to guide Robert. According to the
3.3. Information requirement analysis

scenario, the following are the features used in instructions for guiding Robert:

- Streets network (e.g. Helmastraat, vrienstraat)
- Buildings (e.g. Hello mall, UTC Building, White building)
- Monuments (e.g. the monument in the park, human statue)
- Parks (e.g. Enschede park)
- Water features (e.g. Rutbeek river)

For an easy access of the above features by the route guidance function, their storage in a database is indispensable. As the route calculated is provided to the route guidance function as a series of pedestrians’ navigable segments, it is clear that the guidance function uses information associated to these segments and additional information from the database to provide guidance instructions to the user. For segments that corresponds to the normal street network, the route guidance function uses associated information such as their names (e.g. helmasstraat), whereas for pedestrians’ segments that do not correspond with streets network, the function uses information about surrounding features in guiding instruction (e.g. walk towards the monument).

For guiding pedestrian to the destination, some objects in the above feature classes are used in instructions given by the SAMS (e.g. the Hope building, the white house) whereas other are not used. From this, a number of questions can arise: why the hope building is used? Why not its neighbor building? On which criteria the SAMS is based to choose an object to be used in instruction? The answers to these questions can be found by looking in what context the above objects are used. First of all, the pedestrian specifies his destination point and the SAMS calculates an optimal route and a number of instructions using those spatial objects are generated to guide the pedestrian along the calculated route. The following are the cases where those objects have been used:

- When he is going to take a different direction on the streets (e.g. turn right after the red house);
- As a reference point inside the building (e.g. the black human statue);
- As a reference point in the park (e.g. the monument in the park);
- As a reference point to increase his confidence that he is in the right way (e.g. there is a museum at your right, the hope building in front of him)

These reference points are the ones that Millonig [MS05a] calls landmarks; she defines them as “stationary, distinct and salient objects which serve as cues for guiding pedestrians in their surrounding areas”. She specifies that those objects are especially used at a decision point where a re-orientation is needed or as route marks as a confirmation of being on
the right way. She also denotes that an object is perceived as a landmark if it is unique enough compared to its neighbours objects. Therefore, any object that belongs to the above feature classes can be used as a landmark by the route guiding function if it satisfies the above condition.

4. \textbf{Data required for map matching and display function:}

As the map matching and display function is responsible for connecting pedestrian’s position determined by the positioning function to pedestrians’ navigable segments determined by the route calculation function. This connection allows the route guidance function to provide appropriate guidance instructions that are related to positions along the route calculated. Therefore, the map matching function requires only the updated position of pedestrians and the pedestrians’ segments along which they are travelling.

3.3.3 \textbf{Type of queries asked by pedestrians}

The spatial feature classes and their associated data that are relevant to pedestrians can be used to guide pedestrians once the system is able to access them. It is therefore critical not only to design a database that can store the relevant features, but also to define the appropriate query processing strategies. For this reason, it is wise to assess the types of query that pedestrians pose to the system. Looking back again to the application scenario, we can evaluate what types of query that Robert posed to the system. The following are the queries posed by Robert and their respective types:

- Take me to “hello mall”; This is a get location query
- Take me to the nearest shoes shop : This is a nearest neighbor query
- Take me to “UTC” building and alert me if there is restaurant in less than 5 meters: This is a range query
- Find electronics shops that are within 200 meters: This is a range query

Therefore, the database to be developed must be able to answer the above three types of queries.

3.3.4 \textbf{Overview of the system}

From the above analysis of requirement for a SAMS to support pedestrians’ region-based navigation, an overview of the whole system is of high importance as it will highlight the main parts of the system as well as the part of the system that our research will emphasize on. Figure 3.2 gives an overview of the SAMS system in terms of functions to be executed and of database support.
3.3. Information requirement analysis

Figure 3.2: Overview of the pedestrian navigation system
3.4 Information model

To provide reliable information to their users, SAMS applications require accurate information about the physical world in which they are meant to operate. This information describing the place must be contained in a model that represents all information granularities required by the application. In this section, we model the information required by the SAMS that we found during the analysis of the application scenario in which a pedestrian was assisted by a SAMS in urban navigation.

3.4.1 Location modelling

As people use maps which are models of a particular environment for visualisation, for reference and for navigation from one location to another, location-aware applications require also models of the environment in which they are supposed to operate in order to support their users [Dom01]. Models required by location-aware applications such as SAMS are different from map models; they go beyond the modelling of environment for visualization of place and objects, and take into consideration complex relationships between spatial features. This consideration allows the application to relate these spatial features and to perform some computations with them (e.g. computation of a distance between objects). As users of maps depend on information contained on those maps, location-aware applications depend also on information represented on these location models that they use. Therefore, the success of the location-aware applications depends on how these models represent accurately the information required by the applications.

Based on the analysis of the application scenario, relevant information required by the SAMS in order to support pedestrians in region-based navigation should be modelled in one model. Modelling relevant information is a necessity as it will give result to a model that describes in abstract way, relevant objects and relationships between them as well as other important information about the environment in which the system is supposed to operate. The model will be exploited as a basis to build a data model that allows information contained in the model to be represented in a database that can be queried by the SMAS for supporting pedestrians’ region-based navigation.

According to [FM01] and [DR03], the construction of such models should be driven by the following factors:

- The type of operations the model is expected to support
- The objects and relationship between them to be represented
- The type of model to construct

For the type of operations that the model is supposed to perform, [FM01] and [DR03] emphasize that the extent of the information to be represented
in the model, is defined by the operations that the application is designed for; therefore, the model must represent necessary information that a system functions may need. Based on the analysis of the application done before, the information to be represented in the model must incorporate data required by each of the four functions to be performed by a SAMS.

For modelling objects and their relationships, [FM01] proposes to choose among geometric modelling, symbolic modelling and hybrid model. In a symbolic model, objects are represented as symbols associated with their names whereas in a geometric model, objects are modelled as points, line or regions represented in a coordinate system. Each of the two methods has its own advantages and disadvantages: the symbolic model is easy to read and understand for humans, and supports better scalability; however, it is not suited for computation of information like distance between objects because it lacks position accuracy. The geometric model represents objects in a coordinate system; this allows it to be appropriate for identifying accurate positions and to be suitable for calculating distance between objects. However, for a system that requires to address their data to their users in a meaningful way for humans, extra computing are required to map the coordinates to human understandable information. Since each of the two types of modelling has its advantages and drawbacks, they can be combined to overcome their respective drawbacks and make a hybrid model which will combine the advantage of performing precise calculations and submit the results to the user in an understandable manner.

Due to the fact that the SAMS performs some computations (e.g. route calculation or distance between objects) and deliver the results to pedestrians we choose to use the hybrid model because of its combined advantages.

According to [FM01], the next step after choosing the type of modelling for objects, concerns the construction of an implicit model or explicit model. In an implicit model, the application contains the model information whereas in an explicit model, the information model is stored separately and can be queried by the application. Due to the fact that the explicit model is separated from the services that are provided, it will achieve a better scalability. Hence, we choose the explicit model as the type of model that will be separate to SAMS application. In summary, the information model which will be used to build a database that supports SAMS for pedestrians’ region-based navigation will represent all information found during the analysis and will be a hybrid model as well as an explicit model.

Modelling the information required by the SAMS requires a critical understanding of the underlying semantics and a modelling language that allows expressing these semantics correctly. To build this model, we choose to use UML as our modelling language because it will not only permit us to model the whole system but also it has a rich and powerful syntax that will allow us to model properly the semantics of the application. In UML, classes of objects are modelled as squares and relations between objects are modelled as inheritance, composition or as aggregation.

For a better clarity of the information model built, we divided the model
into two parts that we will combine to make one complete information model. Figure 3.3 shows the first part of the model:

Figure 3.3 shows the first part of the information model which represents features found relevant for the application. In order to model these relevant features, we considered physical world or space associated with pedestrians' navigation to be composed by two feature classes: a pedestrian navigable class and a guidance feature class. Pedestrian navigable space class represent all physical world features over which pedestrians can move and the guidance feature class represent all features that can be used to guide pedestrians.

As pedestrians can move on network space (e.g. street network) as well as inside unconstrained areas that lack underlying paths (e.g. parks or other open spaces), the pedestrian navigable space class is modelled as a generalisation of the network space class and unconstrained space class of objects. The network space class represents features that form network paths over which pedestrians can travel whereas the unconstrained space class represents remaining features which do not have predefined paths over which they can travel.

Since the analysis of relevant data for route guidance function revealed that the system requires and uses guidance features, we considered a guidance feature class that will represent features that can be used to guide pedestrians. Based on result of the analysis done before, guidance features required by the system can be classified into a building class, a landmark class, a water feature class and a monument class. These four classes are modelled with a generalization relationship between them and the guidance features class.

To maintain a better clarity of our model, the second part of the information model is presented in figure 3.4. Since the second part of our model is connected to the first part already presented, some classes of the first part are again presented in the second part order to show their relationships with the remaining classes; these classes are unconstrained space, network space and guidance feature classes.

The analysis of relevant data required by the system revealed that the route calculation function requires pedestrians' navigable routes in order to calculate a route from an origin to a destination. As unconstrained space and network space are modelled as inheritance of the pedestrian navigable space, it is in this space that pedestrian routes have to be established. As in car navigation systems routes are modelled as a succession of road segments and road nodes, our model represent also pedestrians' routes as a succession of pedestrians' segments and pedestrians' nodes. As long as a pedestrian is moving along a street network, pedestrians' segments and pedestrians' nodes will correspond to street segments and street nodes respectively; once the pedestrian will be inside an unconstrained area, pedestrians segments and pedestrians' nodes must be defined inside the area in order to be taken into account by the system. By modelling pedestrian routes in that way, object classes to represent those segments and nodes have to be created.

For each of the unconstrained space class and network space class, two
Figure 3.3: The first part of the information model
Chapter 3. Information model

Figure 3.4: The second part of the information model
classes with an aggregation relationship are defined to represent segments and nodes features in the corresponding space. For unconstrained space, a navigable segment class is defined to represent route segments that a pedestrian can follow inside the area and a navigable node class is also defined to represent the nodes inside these types of area. The unconstrained space class is also a generalisation of two feature classes: park class and open space class. The former represents park features whereas the latter represents other open spaces which do not have an underlying paths network.

For network space, a street segment class is defined to represent segments that correspond to normal streets of the area and a street node class is also defined to represent nodes that are associated with streets inside a network space. In summary, four object classes are defined from the unconstrained space and network space classes: navigable node class, navigable segment class, street segment class and street node class. These four classes represent together features over which pedestrians can move; therefore, it is wise to define a pedestrian orientation node class which is a generalisation of navigable node class and street node class and to define also a pedestrian segment class which is a generalisation of navigable segment class and street segment class. The pedestrian orientation node class and the pedestrian segments class make aggregation relationships with a pedestrian route class which represents a combination of those segments with their corresponding nodes.

As in real physical world especially in urban areas, objects such as buildings, landmarks or parks are linked to street network by addresses number, we argue that guidance features found during the analysis need also to be linked to pedestrian routes by an address. This will enable the SAMS to identify geographic location or features and knows on which routes they can be accessed. Due to this need, the guidance feature class is linked to the pedestrian segment class and this latter is linked to an address range class which represent address numbers that link guidance features with a segment.

As the analysis of relevant data revealed that the system requires also the position, the direction and the speed of pedestrian continuously, we propose to model this time-varying information by considering that the evolution of these three variables of a pedestrian moving inside pedestrian routes during a given time interval corresponds to a trajectory record. For this reason, we defined a pedestrian trajectory class and associated it to pedestrian route class because pedestrians are always within the navigable space. Figure 3.5 shows the complete information model as a combination of the first part and the second part of the model that we above presented.

### 3.4.2 Validation of the model

As the information model described in figure 3.5 will be exploited as a basis to define a data model that allows the representation of this information in a database, it is logical to first validate it according to required data found during the analysis and according to factors of minimal modelling effort presented.
Figure 3.5: The complete information model
According to [BD05], the quality of a location-based application depends on three properties of the model that represents the location information; these properties are:

- **Accuracy**: the information represented in the model should be consistent with the real world to the extent required by the application. Therefore, an accurate model has to represent required objects and their relationships in a way that allows the application to perform requisite functions.

- **Flexibility**: a model is flexible if it can support additional information to be added to the model without significant changes in the model. Flexibility of models that represent physical world reality is crucial because of its dynamic behaviour. For that reason, the model must be able to accommodate any change that may occur in the physical world without affecting significantly the model.

- **Granularity**: the granularity of a model describes the level of details that the model describes. According to [Dom01], highly detailed model generally describe a small part of the physical world whereas less detailed model may describe a large part. Therefore, the model must be detailed to an extent suitable for the application.

The validation of the information model will be based on the above quality elements but also on the following questions that we will prove the reliability of the model:

1. Is the model able to fully support the positioning function?
2. Is the model able to fully support the route calculation function?
3. Is the model able to fully support the route guidance function?
4. Is the model able to fully support the map matching function?

- **Validation based on reliability questions**:

  For question 1, the information model is considered as suitable for the directional positioning function as it represents properly data produced by this function. By means of a pedestrian trajectory class, the model describes data related to pedestrian motion (i.e. position, direction, speed) tracked continuously by the directional positioning function. The information model represents properly this information as it associates this pedestrian trajectory class to pedestrian route class; this is a correct association as pedestrians are always tracked on their corresponding navigable routes.

  For question 2, the model fully supports the route calculation function as it accommodates all data required by this function. The model takes into account the capability of pedestrians to navigate along network space.
such as walkways or streets as well as inside areas that have no obvious network structure; it describes pedestrian routes from network space and unconstrained space. Therefore, the model allows the generation of routes suitable for pedestrians.

For question 3, the model supports properly the route guidance function as it separates objects used for route guidance instructions from the ones used for route calculation; however, it maintains meaningful associations between them. These associations described in the model allow the route guidance function to link routes generated with environment features to give appropriate guidance instructions to pedestrians. The model associates guidance features to two object classes: firstly, to pedestrian routes class because the guidance features are accessible through these routes and this association allows the application to identify and describe guidance objects and other features with reference to pedestrians’ routes; secondly, the guidance object class is associated to unconstrained areas because some guidance features can be found inside such kind of areas. The pedestrian routes class is also linked to an address range class because each pedestrian route will have a range of address number to be given to features that are accessible through that route; this will allow the identification of features that are on the same route.

For question 4, the model supports the map matching function as it associates pedestrians’ trajectory class to pedestrians’ route class. This representation allows the map matching function to map a pedestrian position and direction to the exact corresponding route. Therefore, pedestrian position and direction are described by the model relative to pedestrians’ routes and link them through the address class to the guidance features.

- **Validation based on quality elements:**

  **Accuracy:** as an accurate model must represent relevant data in a way that allows the application to perform its functions, the accuracy will be validated by verifying that the model is appropriate for the application. Since the model supports the four basic functions of the application which consist of tracking pedestrians in urban areas and provide them with navigation assistance, we can conclude that our model is accurate enough for the application.

  **Flexibility:** due to dynamic behaviour of the physical world, flexibility of a model is a good quality as it allows the model to accommodate any additional information by keeping its high level of structure. The above information model is flexible because additional features of the physical world can be accommodated by the model; any additional feature may be added to the model because any new feature can be considered either as a specialisation of guidance feature class or of pedestrian navigable space class because these two classes are at the high level of the model and are modelled as generalisation of the following classes.

  **Granularity:** as the granularity represents the level of details represented by the model, the details of the physical world represented in the
model must be sufficient for the application. Since the application to be supported by the model cover an area of a dimension of a city, we argue that the details presented in the model which stop at the building level and not at the room level is enough for this application.

3.5 Conclusion

The purpose of this chapter was to assess and to model information required by a SAMS that supports pedestrian region-based navigation in urban area. This information was assessed by sketching an application scenario that represents a pedestrian assisted by a SAMS running on his mobile phone in an urban area and a model describing this information in a well structured approach was built.

As the first research question of our research is "What are the data about pedestrians' physical world required by a SAMS in order to support pedestrian region-based navigation in urban areas? And how to model this data", this chapter answered this research question by identifying required data and by building a model that represents this data and their relationships. By answering this first research question, the first objective of this research which was "To model the spatial and spatio-temporal aspects of the world associated with pedestrians' region based navigation" is also met. Therefore, this chapter answers the first research question and meet the first research objective.
Chapter 4

Data modelling

This chapter aims at defining data models suitable for database representation of concepts of the SAMS application presented in chapter three that supports pedestrian region-based navigation. A data model describing required data types and operations is defined in section 4.2; a conceptual data modelling is described in section 4.3; finally, a logical data modelling is presented in section 4.4.

4.1 Introduction

The SAMS application presented in chapter three requires a database support to manage the time changing location of pedestrians and to store relevant data about their environment; this will allow the SAMS to ask queries about pedestrians' locations as well as about their environment. Due to the continuously changing pedestrians' locations, databases that can support SAMS that assist pedestrians fall into the category of moving objects databases. Moving objects databases have their root in spatio-temporal databases which deal with only discretely changing geometries whereas continuously changing geometries to their turn are a special concern of moving objects databases; hence, the term moving objects [GS05].

Moving objects databases research can be seen in two perspectives [GS05]: a spatio-temporal perspective and a location management perspective. For the spatio-temporal perspective, the interest is in database storage of a whole history of a moving object and be able to ask queries about any time in the past of the movement, whereas for the location management perspective, the interest is in managing in the database the continuously changing information about the current positions of moving objects and be able to ask queries about their current positions and also to predict their near future positions. Since the goal of the SAMS application is to continuously track pedestrians' positions and provide to them real time services corresponding to their current positions, database support for the SAMS application dealing with pedestrians' region-based navigation will be considered at the location management perspective of moving objects.
In order to support SAMS applications, databases must manage information required by the applications and allow the applications to pose queries. Managing in a database this information involves also managing the continuously changing positions of pedestrians in order to link their locations to the surrounding environment; this is a challenge because normally, data in a database is assumed to be constant unless it is explicitly modified [GS05]. Consequently, in order to represent a pedestrian position in a database and answer queries about his location, pedestrian’s location has to be continuously updated. However, this continuous update of data is also a problem in case a large number of pedestrians may send very frequent updates as it will give result to a high update load.

Tracking pedestrians and manage their locations in a database require new capabilities that are lacking in existing DBMS [WXCJ98]. Adding these capabilities to the DBMS imply its extension to support applications dealing with management of moving objects; the extension must be made by providing suitable data types and operations, by extending query and data manipulation language as well as by defining appropriate indices [GS05]. Appropriate data types and operations will reflect objects of the real world and their properties that are relevant to the application and express all of the questions that may rise about these objects.

### 4.2 Data model

The goal of this section is to provide a comprehensive data model that allow to represent application data, their relationships in a database as well as their querying. The data model to be defined in this section comprises of data types for data representation, and operations for computations on this data. As relevant data found during the analysis range from spatial data to spatio-temporal data, it makes sense to define corresponding data types for each of the two aspects.

#### 4.2.1 Spatial aspect data modelling

As the SAMS that support pedestrian region-based navigation is associated with spatial data, database modelling of this data is indispensable for their storage. Many researches have been conducted on database modelling of spatial features; therefore, it is logical to base ourselves on these researches and determine how we can model these spatial features to allow their storage and querying inside a database. The main contribution in modelling spatial data was the definition of the OpenGIS simple features model for SQL [Ope99] proposed by the Open Geospatial Consortium (OGC) which is a non profit, international voluntary consensus standards organization that manages the development and implementation of standards in the geospatial and location based
services. The OpenGIS simple features model defined in [Ope99] describes spatial objects as geometry features having the following properties:

- Every geometry has a spatial reference system which describes the coordinate space in which the geometric object is defined;
- Every geometry belongs to a geometry class in the defined model.

Figure 4.1 shows the OpenGIS simple feature model defined by [Ope99]

Figure 4.1 shows the objects geometry model which consists of objects classes. Some objects classes in the model are instantiable and others are non-instantiable. The Geometry class is a non-instantiable class which is the root class of the model as well as a generalisation of all other object classes in the model. Since Geometry class is associated with a spatial reference system and as it is a generalisation of the other objects classes, therefore, all classes in the model are associated with a spatial reference system.

The Geometry class is a generalisation of four objects classes Point class, Curve class, Surface class and GeometryCollection class. The Point class is an instantiable class that represents 0-dimensional geometry objects with a single location in a coordinate system. The curve class is a non-instantiable class that represents 1-dimensional geometric objects represented as a sequence of points with a sub-class Linestring which is an instantiable class that uses a linear
4.2. Data model

<table>
<thead>
<tr>
<th>Application objects</th>
<th>Spatial types</th>
</tr>
</thead>
<tbody>
<tr>
<td>Building</td>
<td>Point</td>
</tr>
<tr>
<td>Landmark</td>
<td>Point</td>
</tr>
<tr>
<td>Water feature</td>
<td>Polygon</td>
</tr>
<tr>
<td>Monument</td>
<td>Point</td>
</tr>
<tr>
<td>Unconstrained space</td>
<td>Polygon</td>
</tr>
<tr>
<td>Street node</td>
<td>Polygon</td>
</tr>
<tr>
<td>Street segment</td>
<td>Polygon</td>
</tr>
<tr>
<td>Pedestrian segment</td>
<td>Polygon</td>
</tr>
<tr>
<td>Pedestrian orientation node</td>
<td>Polygon</td>
</tr>
</tbody>
</table>

interpolation between points; the Linestring class is to its turn a generalisation of a Line class which is a Linestring with exactly two points and of a LinearRing which is a Linestring that is both closed and simple.

The Surface class is a non-instantiable class that represents two-dimensional geometric objects and is a generalisation of Polygon class which is an instantiable class that represents simple surfaces that are planar. The GeometryCollection is also an instantiable class that describes collections of geometry objects such as points, curves or surfaces; it is a generalisation of MultiSurface, MultiCurve and MultiPoint object classes.

Based on the above model of spatial objects and also on the information model presented in chapter three, table 4.1 shows spatial objects relevant for our application and their corresponding object classes:

The implementation of the above model in a database can be done in two ways: by using a geospatial type library as extension or by using a DBMS with native data types that can represent geometries. Based on the OpenGIS simple features specification for SQL presented in [Ope99], extensions of some DBMS to support geographic objects have been made; this is the case of PostGIS for PostgreSQL, ESRI’s SDE and Oracle’s spatial data cartridge (extensions).

Among the above mentioned extensions, the PostGIS extension for PostgreSQL object-relational DBMS is of great interest for us not only because it is an open source, free software, but also because of its mature, its good functionality and its functional extensibility [Res05]. The PostGIS is an open source library of functions and data structures that extend the PostgreSQL DBMS for supporting spatial data [Res05]; it implements the OGC simple feature model for SQL databases which allows the storage of spatial objects. Relying on PostgreSQL/PostGIS provides not only suitable database storage for our spatial data but also it provides appropriate functions for the analysis and processing of this data; It also gives to our application possibilities generally given by DBMS for concurrent updates, backup/recovery, real transactions, triggers and validation [Gro].
4.2.2 Spatio-temporal aspect data modelling

Relevant data to be managed in the database having a spatio-temporal connotation are pedestrians’ trajectories. As we wish to manage in a database these trajectories and be able to retrieve pedestrian’s current position, direction and speed, the data model has to describe data structures suitable for the representation of these three variables. As stated before, there are researches that were carried out on database representation of moving objects for a location management perspective; the main contribution in this perspective was done by [WXCJ98], which proposed a model called MOST that we can rely on and define data structures suitable for the representation of pedestrian trajectories.

The MOST data model introduces a concept of dynamic attributes that change continuously as function of time without being explicitly updated. According to MOST, a dynamic attribute \( A \) is represented by three sub-attributes, \( A.value \), \( A.updatetime \) and \( A.function \) where \( A.value \) is the value of the attribute at time \( A.updatetime \) until the next update, and where \( A.function \) is a function of a single variable \( t \) that has value 0 at \( t = 0 \). The semantics of this representation is that the value of the attribute can be obtained at any time \( t \geq A.updatetime \) by the following formula:

\[
Value(A,t) = A.value + A.function(t - A.updatetime)
\]

For moving objects on routes, [WXCJ98] proposes to extend the dynamic attribute concept and include also routes by modelling a dynamic attribute \( L \) as represented by five sub-attributes as follows: \( L.route \), \( L.x.value \), \( L.y.value \), \( L.updatetime \), \( L.speed \) where \( L.route \) is a pointer to a spatial object specifying the route on which the objects is moving; \( L.x.updatevalue \) and \( L.y.updatevalue \) are the \( x \) and \( y \) of the position of the object on the \( L.route \) at time \( L.updatetime \) and the \( L.speed \) is a linear function of the form \( f(t) = b*t \) where \( b \) is the speed of the moving object. The function \( f(t) \) gives the current distance from the starting location as a function of the time since \( L.updatetime \). Even though this type of modelling can be a good basis for modelling the spatio-temporal aspect of our application, it is appropriate for objects moving freely in \( x \) and \( y \) plane.

Based on the dynamic attribute concept, it is reasonable to extend it and relate pedestrian’s position, direction and speed to pedestrians’ navigable routes. We consider position, direction and speed because they are variables that are relevant for our application and which have a spatio-temporal connotation. Modelling pedestrians’ location relative to their navigable routes is realistic as it allows the system to track pedestrians along routes they are able to navigate.

In this perspective, we propose to represent the dynamic pedestrian’s location attribute with six sub-attributes namely: \( route \), \( startlocation \), \( starttime \), \( direction \), \( speed \) and \( uncertainty \). Therefore, for a location dynamic attribute \( Loc \), its six sub-attributes are:

1. **Loc.route**: is a pointer to a spatial object which describes the route along which a pedestrian is moving;

2. **Loc.startlocation**: is a point on Loc.route that specifies the location of a
4.2. Data model

pedestrian at a time Loc.starttime;

3. **Loc.starttime**: is the time at which a pedestrian was at a position Loc.startlocation on the route Loc.route;

4. **Loc.direction**: is the direction of a pedestrian along a route;

5. **Loc.speed**: is a function that represents the predicted future locations of the object. It gives the distance from the Loc.startlocation as a function of time elapsed since the last location update Loc.starttime (i.e. the distance is $v \cdot t$). In its simplest form, Loc.speed represents a constant speed $v$.

6. **Loc.uncertainty**: represents a threshold on the deviation of pedestrian’s location. It may be as a constant or a function of time elapsed since the last location update. Every time this threshold is reached, an update of the location must be made.

The above modelling of the spatio-temporal aspects associated with pedestrian region-based navigation application is appropriate because it allows to represent pedestrians’ position, direction as well as speed with respect to their navigable routes as it is required by the SAMS. With the above modelling, we have to define data types that allow their representation in a database:

- **Data type for Loc.route**: As Loc.route is a pointer to a spatial object that represents the route along which a pedestrian is moving. As pedestrians’ navigable routes have been modelled early as polygon features represented by pedestrians’ navigable segments and nodes, loc.route will be represented by a basic data type such as integer that will identify the segment on which a pedestrian pedestrian is moving.

- **Data type for Loc.startlocation**: As Loc.startlocation is a point that specifies the location of a pedestrian on a navigable route, a suitable data type for its database representation is a point data type. The point data type is used to represent the geometrical aspect of an object in space where only its location is important but not its extent.

- **Data type for Loc.starttime, Loc.speed, Loc.uncertainty and Loc.direction**: These four sub-attributes can be represented using basic data types such as integer data type for the first three and character type for the Loc.direction sub-attribute.

Table 4.2 summarizes the six sub-attributes to be represented and their respective data types.

The data types presented in table 4.2 that are required for database management of pedestrian trajectories, are provided with PostGIS which is a library
Table 4.2: Datatypes for representing pedestrian trajectories

<table>
<thead>
<tr>
<th>Sub-attributes</th>
<th>Data types</th>
</tr>
</thead>
<tbody>
<tr>
<td>Loc.route</td>
<td>integer</td>
</tr>
<tr>
<td>Loc.startlocation</td>
<td>Point</td>
</tr>
<tr>
<td>Loc.starttime</td>
<td>Integer</td>
</tr>
<tr>
<td>Loc.direction</td>
<td>Character</td>
</tr>
<tr>
<td>Loc.speed</td>
<td>Integer</td>
</tr>
<tr>
<td>Loc.uncertainty</td>
<td>Integer</td>
</tr>
</tbody>
</table>

that adds spatial capabilities to PostgreSQL. Therefore, as PostgreSQL with its add-on PostGIS library were found able to properly represent relevant data that has a spatial connotation presented in sub-section 4.2.1 as well as data that has spatio-temporal connotation presented in sub-section 4.2.2, we chose it as the DBMS product that we will use for the implementation of the database to be designed in this research.

4.2.3 Operations

Operations to be defined in this sub-section are operations required for computations on application relevant data. As data that are relevant for our application are categorised in spatial data and spatio-temporal data, it makes sense to define operations that are specific for each category:

- **Spatial operations**:

  Spatial operations are necessary for computations and analysis on spatial data; spatial operations to be defined are those we deem relevant for SAMS that support pedestrian region-based navigation. Based on spatial operations defined in OpenGIS simple features specification for SQL presented in [Ope99], relevant spatial operations are:

  - WITHIN (Geometry g1: Geometry g2): indicates if g1 is spatially within g2;
  - DWITHIN (Geometry g1: Geometry g2: distance): returns TRUE if geometries g1 and g2 are within the specified distance one another;
  - INTERSECTS (Geometry g1: Geometry g2): indicates if g1 spatially intersects g2;
  - DISTANCE(Geometry g1: Geometry g2): returns the Euclidean distance between g1 and g2.

  The above operations are suitable for computations on spatial data but not for spatio-temporal data; therefore, operations that can be applied for data with a spatio-temporal aspect have also to be defined.

- **Spatial temporal operations**:
Based on types of queries mentioned in section 3.3.3 that are associated with services requested by the pedestrian, the following are spatio-temporal operations that are relevant for our research:

- **POSITION(Location p1: Time t):** calculates from the current location, the position of the pedestrian p1 on pedestrians’ navigable routes at time \( t \), where \( t > \) the last update time and \( t < \) the next update time;

- **DIRECTION (Location p1: Time t):** calculates the direction of the pedestrian p1 on pedestrians’ navigable routes at time \( t \), where \( t > \) the last update time and \( t < \) the next update time;

- **DISTANCEMS (Location p1: Geometry g2: Time t):** returns the distance between the location of a pedestrian p1 and a static geometry g2 at time \( t \), where \( t > \) the last update time and \( t < \) the next update time;

- **DISTANCEMM (Location p1: Location p2: Time t):** returns the distance between the location of a pedestrian p1 and another moving pedestrian p2 at time \( t \), where \( t > \) the last update time;

- **NEARESTMS (Location p1: Geometry g2: Time t):** returns the nearest static object g2 from the location of a pedestrian p1 at time \( t \), where \( t > \) the last update time and \( t < \) the next update time;

- **NEARESTMM (Location p1: Location p2: Time t):** returns the nearest pedestrian p2 from the location of a pedestrian p1 at time \( t \), where \( t > \) the last update time and \( t < \) the next update time;

- **TRAVELLEDDISTANCE (Location p1: Time t):** returns the travelled distance of a pedestrian p1 since the last position update until time \( t \), where \( t > \) the last update time and \( t < \) the next update time.

Apart from the spatial operations defined in this sub-section which are provided by PostGIS, the defined spatio-temporal operations are theoretical and not available as implemented functions. However, they can be used to show theoretically how the types of queries mentioned in sub-section 3.3.3 that are associated with services requested by pedestrians can be answered. In order to be used in practice, the defined operations must firstly be implemented based on the data model defined in section 4.2.2.

### 4.3 Conceptual data modelling

Conceptual design phase focuses on expressing application requirements without the use of computer metaphors [TPJ03]; the conceptual data schema to be designed in this section will express data requirement of SAMS for pedestrian region-based navigation. Objects required for the application and their associations have been shown in the information model, however the information
model does not capture the attributes and properties of these objects. Therefore, the conceptual schema will show objects, their attributes as well as properties of associations between these objects. The most common way of designing databases is by means of conceptual models; the most popular conceptual models are the Entity relationship (E/R) model and UML [GBE+03]. Due to the very few modelling constructs of E/R and to the stronger capability of UML to model object oriented systems [GBE+03], we chose UML as the modelling language to use for the design of the conceptual data schema; an other reason for choosing UML is that it is the standard language for modelling object-oriented applications and it allows the design of not only the conceptual schema, but also of the whole system [EC02].

4.3.1 Unified Modelling Language

The UML is the open, industry standard visual modelling language approved by the OMG (Object Management Group) [AN02]; its structure consists of three major parts: 1) building blocks which are the basic UML modelling elements, relationships and diagrams, 2) common mechanisms which are common UML ways of achieving specific goals and 3) architecture which the UML view of the system architecture. In the building blocks, UML consists of: things which are modelling elements themselves, of relationships which are relationships that specify how two or more things are semantically related and of diagrams which are views that show what the system will do and how it will do it [AN02]. There exist nine different diagram types in UML, and among them there is class diagram which describes the structure of object classes.

The primary use of UML class diagram is to describe objects class; a class is described by its name, attributes, operations, relationships and behaviour [AN02]. The class diagram describes a class using a box in which attributes names and their types are expressed; operations signatures are described graphically in a separate compartment of the attributes compartment and relationships between classes are described by different types of object-oriented links including generalization, association, aggregation and composition. Figure 4.2 shows a class notation in UML and an example of a person class showing class attributes and operations:

Based on UML class diagram as a conceptual modelling notation, a conceptual schema describing relevant data of SAMS application that supports pedestrian region-based navigation can be constructed.

4.3.2 Conceptual schema

The conceptual schema to be designed in this subsection defines content of a database that can support SAMS application to assist pedestrian region-based navigation. This database ensures the management of pedestrian trajectory data and the storage of data required by SAMS functions such as route calculation function and route guidance function. Based on the information re-
4.3. Conceptual data modelling

requirement model defined in chapter three as well as on the data model defined in section 4.2 of this chapter for database representation of required data, a conceptual schema is defined in figure 4.3; the defined conceptual schema identifies classes of objects together with their attributes that are relevant for the application as well as associations between these classes and data constraints among them:

The conceptual schema designed in figure 4.3, describes objects classes that are relevant for pedestrian region based navigation application. From the top of the schema, a navigable area abstract class is defined to represent physical world areas that are navigable by pedestrians. Because of the ability of pedestrians to walk over roads network and inside areas that lack underlying paths network, the navigable space is considered as a generalisation of two instan-
tiable classes: unconstrained space class and network space class. The unconstrained space area is a class that represent area features inside the physical world that lack underlying paths but inside which pedestrians can walk; among unconstrained spaces we can find parks, city squares or other open spaces that are navigable by pedestrians. Hence, the unconstrained space class is a generalisation of park class and open space class which represent park features and other areas without predefined pedestrians’ paths respectively. The network space class represents features organised in a network over which pedestrians can walk; e.g. streets network, roads’ side walkways and roads’ crossing areas; these features have in common the fact that they are predefined networked paths over which pedestrians can walk.

As navigation functions require pedestrians’ navigable routes segments and nodes for route calculation and guidance, we modelled pedestrians’ navigable routes as a sequence of segments and nodes inside the unconstrained space class as well as inside the network space class. It is in this logic that a Guidance node class and a navigable segment class are defined with an aggregation association with the unconstrained space class because they are defined inside this latter. The multiplicities of this aggregation specify that at least one navigable segment and two guidance nodes are required inside an unconstrained area in order to be taken into account by the route calculation function. The ap-
Figure 4.3: The conceptual schema
proach based on segments and nodes are applied also to network space and two classes namely, street node class and street segment class are also defined with an aggregation relation with the network space class. A pedestrian node class was defined in the schema as a generalisation of a street node and guidance node classes and a pedestrian segment class was also defined in the schema as generalisation of navigable segments and street segment.

As guidance features have been found to be relevant for the application, a guidance feature class was defined in the schema to represent all those features that the system requires to guide pedestrians. This guidance feature class is modelled as a generalisation of a building class, landmark class, water feature class and monument class. A service class is also defined and associated to the building class with a many to many relation because many services can be delivered in one building and one service may be delivered in different building. The service class was defined to represent services information which is associated with buildings; e.g. shops, restaurant, hotel, etc.

Due to the fact that guidance features are accessible through pedestrians’ routes, the guidance feature class has been linked with the pedestrian segment class; this association will allow the system to know a segment that access a specific guidance feature as well as to know all features that are accessible through a pedestrian segment. An address range class is also defined in the schema and linked to pedestrians’ segment class; the multiplicities of this association express that a pedestrian segment is associated with two address ranges on its left and right side. With this association, features that are accessible through a certain segment will be given different address numbers that are within the two ranges depending on which side of the segment they are. The unconstrained space class is also linked to guidance feature class because an area is considered as navigable by pedestrians if it has at least one guidance feature; for that reason, this association specifies that an unconstrained space may have one or many guidance features whereas one guidance feature may belong exactly to one unconstrained space.

Finally, since the application consists of tracking pedestrians’ positions along their navigable routes and manages their trajectories in a database, a pedestrian trajectory class is also defined in the schema. The pedestrian trajectory class defined is associated to the pedestrian segment class because this latter represents features over which pedestrians are tracked. This association specifies that a pedestrian trajectory belongs to one pedestrian segment whereas one pedestrian segment may have zero or many pedestrian trajectories.

4.4 Logical data modelling

This phase of database design is between the conceptual data modelling phase and the physical data modelling phase; it consists of designing a logical data schema that corresponds to data definition rules of a specific DBMS product.
The logical data schema attempts to describe data in more details by specifying entities and relationships between them, attributes for each entity, primary key, foreign keys and other constraints associated with entities [Dom01]. Due to the choice of UML as the modelling language to use in this research, the logical data schema to be designed has to be represented by UML class diagram that describes classes tagged as persistent from classes defined at the conceptual schema level. According to [EC02], transforming a conceptual schema into a logical schema specific to a DBMS product requires a set of guidelines; for that reason, [EC02] proposes four phases of transformation: 1) transformation of classes and attributes, 2) transformation of associations, 3) transformation of generalisations and 4) transformation of aggregations and compositions. Based on these four phases, the conceptual schema presented in figure 4.3 is transformed into a logical schema using data definition of PostgreSQL as it was chosen as it is the chosen DBMS as mentioned in sub-section 4.2.2.

4.4.1 Transformation of classes and attributes

As specified by [EC02], classes to be transformed are only persistent classes; persistent classes are classes whose instances require to be stored in the database. Therefore, before transforming classes described in the conceptual schema described in figure 4.3, we must first identify persistent classes among them. Based on data required by SAMS functions as described in subsection 3.3.2, persistent classes found are: pedestrian segment, pedestrian node, pedestrian trajectory, road, address range, area, service, landmark, water feature and building classes. As the UML persistent classes have to be transformed into PostgreSQL classes, the UML data modelling profile represents these latter as stereotyped classes which are class elements with a stereotype <<table>> applied to them. The attributes of the persistent classes are transformed to their turns into attributes of PostgreSQL data types and a primary key for each class is defined.

4.4.2 Transformation of associations

According to [EC02], UML associations between persistent classes are transformed as foreign keys by using either unidirectional relationships or bidirectional relationships. [EC02] proposes the use of unidirectional relationships if application queries require data in one direction and the use of bidirectional relationships if queries require data in both directions. Associations between persistent classes are transformed by creating foreign keys; figure 4.4 shows how classes associations are transformed by creating foreign keys in child classes pointing to primary keys of parent classes.
4.4.3 Transformation of generalisations

Generalisation relationships between persistent classes at the conceptual level can be transformed in three ways: the first way is to transform all classes participating in a generalisation relationship into one union class; the second way is to transform each persistent class into its corresponding class with only attributes accessible by the persistent class; and the third way is to transform each persistent class into a class that has exactly the same attributes as the persistent class. For generalisation relationships in the conceptual schema presented in figure 4.3, we chose the first way because it locates all attributes in one table which is convenient for updates and retrievals of any class.

4.4.4 Transformation of aggregation and composition

Aggregation and composition relationships between classes can also be transformed by using primary-foreign key pairs. For an aggregation with a many-to-many relationship, the transformation is done by using an intermediate class whereas for a one-to-many relationship a transformation is done with a foreign key in the aggregated class. For a composition relationship a foreign is also used but with also a constraint that if a deletion occurs at the parent class, its part must also be deleted.

4.4.5 Logical schema

The logical schema presented in this sub-section consists of persistent classes described as class elements with a stereotype <<table>>; the relations between them are results of transformations of relations presented in the conceptual schema. Figure 4.4 shows the logical schema obtained after the transformation of the conceptual schema.

4.5 Conclusion

The purpose of this chapter was to define a data model suitable for database representation of requirements of SAMS application that supports pedestrian region-based navigation. Data types suitable for database representation of requirements have been defined in section 4.2, and a conceptual schema has been designed in section 4.3 to define classes of objects, their attributes that are relevant for the application, associations between these classes and data constraints among them. Finally, the conceptual schema was transformed into a logical schema in section 4.4; the logical schema describes the data according to data definition of PostgreSQL as the implementation DBMS.

As the second research question of our research “How to design and implement a database that can manage trajectory data associated with pedestrian’s region-based navigation and answer relevant types of query?” was divided in
three sub-questions, this chapter answers the first sub-question. The first sub-question which was “How to design a conceptual data model and a logical model?” is then answered as the two data models have been designed in this chapter.
Chapter 5
Implementation

The objective of this chapter is to describe the implementation of the database to be used by SAMS for supporting pedestrian region-based navigation. The implementation is done by transforming the logical schema presented in chapter 4 into a physical data schema in section 5.2. For testing purpose, the process of generation of a testing dataset is described in section 5.3.2 and the dataset loaded into the database. Finally, different types of queries that the database is supposed to support are formulated and some executed in the database in section 5.4.

5.1 Introduction

The implementation phase of a database corresponds to its physical design; the physical design is the third level of three different abstraction levels for database design process which are “the conceptual design level”, “the logical design level” and “the physical design level” [RSV]. At the conceptual design level, data requirements for database design are expressed by a conceptual schema. At the logical design level, the defined conceptual schema is transformed into a logical schema that describes application data in conformity with data definition rules of a specific DBMS. Finally, at the physical design level, the defined logical schema is transformed into a physical schema that specifies data storage structures.

The physical design consists of transforming a logical schema into a physical schema that describes data structures in the SQL of the DBMS chosen for the implementation [RSV]. The main activity at this design level is to generate from the logical schema, SQL statements that specify how data will be stored in a database. Typical specifications included in the SQL statements generated by the transformation are: specification of tables and their columns, specification of foreign keys, specification of indexes, query optimization and other performance considerations that may be required by the application. Therefore, the physical schema consists of SQL statements that define how data will be stored by the chosen DBMS. According to the specific needs of the applica-
tion, [RSV] specifies that the physical schema given by the transformation can be fine-tuned to improve the response time and storage space.

5.2 Physical data schema

For the implementation of our database, the logical schema presented in chapter 4 has to be transformed into a physical schema; the transformation is done by converting the UML class diagram that represents the logical schema into SQL statements that specify data storage structures. As the DBMS chosen for our research is the PostgreSQL with its add-on PostGIS library of functions, the physical schema to be generated has to be described by SQL statements of PostgreSQL. For this purpose, EA was used again because it supports PostgreSQL DBMS and transformation processes. For the generation of a physical schema from a logical schema, EA generates a DDL (Data Definition Language) file that encapsulates SQL statements specific for PostgreSQL; by executing these SQL statements using PostGreSQL, empty tables corresponding to the classes of the logical model will be created in the database.

5.2.1 DDL generation

As mentioned before, creating a database requires SQL statements that specify how data will be stored inside the database. Generally, SQL statements are divided in two major categories: Data Definition Language (DDL) statements and Data Manipulation Language (DML) statements; the DDL statements are used to build and modify the structure of tables and other objects in the database whereas the DML are used to work with the data in tables. Therefore, in order to specify data structures of the database we intend to build, DDL statements are required; as the EA supports the generation of DDL from logical schemas, we will rely on it and generate DDL statements that define how data will be stored in the database we intend to design.

As the EA generates DDL from models, the DDL generated in this research was from the logical schema presented in chapter four. By selecting the logical schema and by selecting DDL generation, EA shows a DDL dialog in which specifications of the generation are introduced. Figure 5.1 shows the DDL generation dialog window which appeared when we selected to generate the DDL. As it can be seen in the dialog window, the logical schema classes from which the DDL has to be generated are selected in blue. The dialog proposes also options for the DDL generation; the option that we selected was to create primary keys and foreign keys constraints; it is also inside this dialog window that we specified the path of the result file that we called DDL and that EA stored with SQL extension.

The DDL file generated from the logical schema consists of SQL statements that specify tables to be created as well as their columns. Figure 5.2 shows the DDL file generated and the SQL statements that are contained in the file.
Chapter 5. Implementation

Figure 5.1: DDL generation dialog window

Figure 5.2: The DDL file generated with SQL statements
SQL statements contained in the DDL file presented in figure 5.2 are of
two main types: CREATE TABLE statements and ALTER TABLE statements.
The CREATE TABLE statements specify names of the tables to be created,
their columns’ names as well as columns data types; and the ALTER TABLE
statements specify constraints that are associated with tables like primary keys
and foreign keys.

5.2.2 Tables generation

As the DDL file encapsulates SQL statements that define the names of the ta-
bles, their columns names as well as their data types, tables to be generated
from this file will correspond to the contents of these SQL statements. By exe-
cuting the SQL statements of the DDL file in PostGreSQL\PostGIS database,
empty tables and their respective columns were created together with tables’
primary keys and foreign keys.

However, even though the SQL statements of the DDL file specified tables
and their columns, columns that will store geometries of the spatial features
were not created because their structures were not specified in the DDL file.
This is because the SQL statements generated by EA corresponded to Post-
GreSQL data definition and not to its combination with PostGIS which adds
capabilities of handling spatial objects. Therefore, for tables that have to store
spatial information, an additional column for storing geometries is required. As
PostGIS provides a function called \textbf{AddGeometryColumn} that adds a geometry
column to a table, by specifying the type of geometry to be stored by the table,
we can use this function to create such columns in tables that will store spatial
objects. The \textbf{AddGeometryColumn} function was executed with suitable argu-
ment to add a geometry column to every table that will store spatial objects and
Figure 5.3 shows an example of a command that generates a geometry column.
The first window in the figure 5.3 shows the query execution of the \textbf{AddGeom-
etryColumn} function and the second window shows the table with a geometry
column called “shape” added to it.

In order to store geometries of spatial objects in a PostgreSQL\PostGIS, two
more additional tables are required; these required tables are metadata tables
named: SPATIAL\_REF\_SYS and GEOMETRY\_COLUMNS; the SPATIAL\_REF\_SYS table holds the numeric IDs and textual descriptions
of coordinate systems used in the spatial database whereas the GEOMETRY
\_COLUMNS table stores a row for each geometry column occurring in any spa-
tial table in the database.

5.3 Pedestrian dataset

For supporting pedestrian region-based navigation, SAMS require dataset con-
taining data granularities required for supporting pedestrians; however, these
datasets are not available till now. As the database designed present storage
structures suitable for these pedestrian datasets, for testing purpose, it is necessary to find datasets corresponding to the defined storage structures. Among dataset required, are those representing navigable segments and nodes inside unconstrained area; these data are not available due to lack of proper methods for their collection. It is for this reason that a method for collecting these datasets inside unconstrained area is proposed here.

5.3.1 Method for collecting pedestrian data

In this sub-section, we propose a method for collecting pedestrian navigable segments and nodes inside pedestrians’ navigable area which are areas that lack predefined routes. As mentioned by the information model defined in section 3.4, in order to use such kind of areas for guiding pedestrians, SAMS require navigable routes inside these areas. For this reason, a method for collecting navigable segments and nodes that are suitable for pedestrian region-based navigation is presented in figure 5.4.

The method proposed in figure 5.4, starts by identifying an area suitable for pedestrian navigation. After identifying the area, the method proposes to define entrance points and exit points associated with the streets network around that area; this association is required in order to connect the street network with the network of segments and nodes that have to be defined inside the area. After identifying the entrance and the exit points, nodes are identified at the defined entrances and exit points. As pedestrians require guidance instruction at decision points such as nodes that connect two or many segments, a node is defined at a place which can be described for guiding a pedestrian. After identifying the nodes at the entrance and exit nodes, the method proposes to
5.3. Pedestrian dataset

Figure 5.4: Method for creating pedestrian segments and nodes inside an unconstrained navigable area
record attributes data of these nodes. The attributes data to be collected are
data characterizing the location of the orientation node and any surrounding
area such as a statue on the right, a tennis court on left, etc.

From one of the entrance nodes, another node must be identified inside the
area in the direction of one of the exit node; this node must be visible from the
first one. The method then proposes to check if there is any obstacle between
the last identified node and the entrance node defined before; if there is any
obstacle, a new node must be identified, and if there is no obstacle between the
two nodes, a segment must be defined between these two nodes.

After the segment is identified, its attributed must be defined to describe its
location; examples of attributes data that can be associated with segments are:
keeping water fountains on the right, accessible for bicycles, etc.

From the last identified, a new visible node must be identified in the direc-
tion of any of the exit nodes identified before; if one of the exit node is found, an
obstacle checking must be applied. If there is no obstacle between the last iden-
tified and the exit node, a segment must be created between them, otherwise, if
there is an obstacle between them, a new node must be identified. This process
will continue until the last node to be identified, is an exit node that connects
to the street network.

5.3.2 Testing data preparation

Testing data to be used in this research, is based on Top10 NL dataset which
is an object-oriented digital topographical file covering the entire Netherlands.
Top10 NL is a digital topographical file that was made based on aerial pho-
tographs, field surveys and existing files of the Top10 vector file which is its pre-
decessor [Spa08]. Top 10 NL contains a collection of topographical object classes
such as road, rail road, water feature, building, terrain (i.e. land use), admin-
istrative area and many other object types. Due to its rich contents, Top10
NL is widely used in various GIS applications for analysis, management and
planning activities. The choice of Top10 NL as the basic dataset for building
the testing dataset was driven by the fact that some spatial objects required
by our application are contained in the Top10NL dataset; therefore, relying on
Top10NL minimizes the effort required for building our testing dataset.

The testing data preparation phase consists of three mains parts: 1) extract-
ing required features from the Top10 NL dataset, 2) creating relevant features
that are missing in Top10NL and 3) adding relevant attributes to the extracted
and created features.

Selecting and extracting features from Top10NL dataset

The process of testing dataset generation starts by selecting and extracting a
small urban area from the Top10 NL dataset using softwares’ resources. As
current datasets (including Top 10 NL) do not contain all data required for
5.3. Pedestrian dataset

pedestrian region-based navigation, the selection of this area must be based to the level of conformity between features contained in that area and features required for pedestrian region-based navigation; the higher the conformity, the higher the area will be suitable for testing. For testing purpose of the designed database, we selected from the Top10 NL a small urban area having together at least features required for pedestrian region-based navigation that can be found in Top10 NL.

For the selection and the extraction of this area, ArcMap together ArcCatalog was used because of their rich set of tools for spatial data manipulation, creation and processing. After selecting the area of interest, its extraction was achieved using the ArcMap “clip operation”, and a shapefile was created using ArcCatalog to contain spatial objects of the extracted area. However, the chosen area contains spatial objects that are relevant for pedestrian region-based navigation and others that are not relevant; for this reason, relevant spatial objects have to be extracted again from the area to be stored in shapefiles that contain only relevant objects.

Required spatial objects were extracted using “select operation” provided by ArcMap to select only the relevant ones, and the “extract operation” was used to extract and store them in shapefiles that were created using ArcCatalog; six object classes representing spatial objects that are relevant for pedestrian navigation were extracted from the chosen area and stored separately in six shapefiles. The six object classes extracted are: street, unconstrained navigable area, street segment, street node, building and water feature classes. However, based on the conceptual schema as well as on the logical schema presented in section 4.3.2 and 4.4.5 respectively, some spatial object classes are missing in the extracted features; missing object classes are navigable segment class, the guidance node class and landmark class. These classes are missing because they were not contained in the Top 10 NL dataset from which we extracted the first six classes. Therefore, the creation of missing spatial objects is indispensable for testing the designed database.

Creation of relevant features

Relevant spatial objects that are missing in the six object classes extracted from the Top10 NL dataset are created in this sub-section. As the missing spatial object classes are navigable segment class, guidance node class and landmark class, their creation requires first to re-examine “why they are needed”, “where they have to be created” and “how they have to be created”. Based on the conceptual schema presented in section 4.3.2, navigable segments and guidance nodes objects are polygons features required for the generation of pedestrians’ routes inside areas that lack underlying pedestrians’ paths; and landmark are objects considered as guidance features that can be everywhere a pedestrian needs guidance features. Therefore, the navigable segments and guidance nodes will be created inside unconstrained areas whereas landmarks objects can be created inside unconstrained areas as well as inside other places where guidance
features are required. Therefore, the “why” and “where” questions related to objects classes to be created, are addressed by specifying that navigable segments and nodes are needed for pedestrians' route generation and guidance inside areas that lack underlying pedestrians’ paths whereas landmarks can be created anywhere in the area but especially where guidance features are needed.

As the “why” and “where” questions are addressed, the question of “how to create these features” must also be addressed because their creation must respect the concepts behind pedestrian region-based navigation. Therefore, the creation of the segments, nodes and landmarks requires a method to ensure that their creation conform to pedestrian region-based navigation concepts. Based on the method presented in 5.4 for identifying pedestrian navigable segments and nodes inside unconstrained navigable areas.

In order to apply the method defined in figure 5.4, a software is required with suitable operations that allow the creation of these objects together with their attributes. As the Top10 NL dataset was available as ESRI shapefiles and as the extracted objects in sub-section 5.3.2 are also stored as shapefiles, it is logical to store the created features also as shapefiles. To do so, we chose to use ArcMap software together with ArcCatalog because they provide suitable operations for the creation of required objects and store them as shapefiles. Another advantage of creating these objects using ArcMap is that the chosen PostgreSQL DBMS provides a function called shp2pgsql which converts shapefile data format into SQL statements for data loading into database.

Based on the data types of relevant features defined in section 4.2.1, and based on the method described in 5.4, the “create feature operation” of ArcMap was used to create polygons inside navigable areas to represent navigable nodes and polygons representing pedestrians’ navigable segments were created between two nodes and landmarks were created also according to the method defined in 5.4. In order to store the created features, three shapefiles namely, navigable segments, navigable nodes and landmarks shapefiles were created using ArcCatalog to store the segments, nodes and landmarks created in the unconstrained area respectively.

Based on the conceptual schema as well as on logical and physical schemas defined in this research, pedestrians’ trajectories are also represented in the database; therefore, testing data representing pedestrians’ trajectories is also required. As specified in sub-section 3.3.2, data that are required about pedestrian motion is pedestrian point position, pedestrian direction and pedestrian speed; due to lack of real pedestrians’ motion data, it is logical to create these data manually for only testing purpose. However, as pedestrians’ positions are represented as point features and as pedestrians are assumed to be tracked only on their navigable routes, the only criteria to be based on for the creation of these position points is to create them inside pedestrians’ segments or nodes.

As the database designed does not store history of pedestrians’ movement, and as any new position entered in the database causes the loss of the previous one, therefore, one pedestrian will be represented by exactly one spatial point together with other relevant data about that position such as the direc-
5.3. Pedestrian dataset

tion or the speed he was travelling with at that specific point. Pedestrians’ positions were created as shapefiles to represent point features inside pedestrians’ segments using again the create operation provided by ArcMap; these pedestrians’ positions were created inside pedestrians’ navigable routes because we assume that services will be provided to them whenever they are moving on these routes.

Combination of extracted features with created features and addition relevant attributes

As road segments and road nodes are among features extracted from the Top10 NL in sub-section 5.3.2, their combination with navigable segments and navigable nodes respectively is logical to make to one shapefile that represent all segments and another shapefile that represents all nodes. To combine the above mentioned shapefiles, the ArcMap “union operation” was used to produce one shapefile called pedestrian segments which combine navigable segments and road nodes, and another shapefile called pedestrian nodes was created to combine navigable nodes and road nodes.

As some attributes data associated with features extracted from the Top10 NL as well as features created using ArcMap resources are not relevant for pedestrian region-based navigation, unnecessarily attributes data have to be removed and relevant ones have to be added. These unnecessarily attributes were removed and replaced by relevant ones based on attributes specified by the physical schema. For created features, attributes data that were lacking were added based on attributes defined by the physical schema.

Figure 5.5 illustrates the process of generation of the testing dataset; the process starts from the selection of an area from the Top10 NL dataset and continues with the creation of navigable segments, nodes and landmarks as well as to the creation of pedestrians’ position. In figure 5.5, the rectangle boxes specify actions done on data and software operations used, whereas the white parallelogram boxes specify data produced after each action.

Overview of the testing dataset

The testing dataset built consists of seven shapefiles and three data tables. The seven shapefiles are: pedestrian trajectories, landmark, pedestrian nodes, pedestrian segments, water features, buildings and navigable areas shapefiles; and the three tables are: address range table, road table and service table. Figure 5.6 captures the seven shapefiles of the testing dataset.

5.3.3 Data loading

The above described testing dataset has to be loaded in the designed database in order to evaluate its consistency. According to [Res05], they are two ways
Chapter 5. Implementation

Figure 5.5: Testing dataset generation process
5.3. Pedestrian dataset

to upload GIS data into a PostGreSQL\PostGIS database: the first way is to use formatted SQL statements and the second way is to use the shapefile loader/dumper. The first way consists of converting data into a text representation and to use formatted SQL statements to get data into the database; and the second way consists of using the shp2pgsql data loader that converts ESRI shapefiles into SQL suitable for data insertion into PostGreSQL\PostGIS database. Due to very long strings associated with text representation of spatial features, we chose the second way using the shapefile loader which will convert the seven shapefiles to SQL statements.

As the loader has several operating modes defined by command line flags and as tables have already been created in the database, we used the \texttt{-a} command to append data from the shapefile into the database table. Figure 5.7 shows an example of a loader command used to convert the building shapefile into SQL statements.

The first line of the command start by specifying the directory \texttt{P:\ PostGre SQL-Tools\Tools>shp2pgsql} of the shp2pgsql loader that converts shapefiles into SQL files; the meanings of the other commands are as follow:
Chapter 5. Implementation

The above mentioned commands are the ones we entered in the command window; however, the loader itself will generate the two bottom remaining lines which indicate the shapefile type as well as the PostGIS type into which the shapefile was converted. From the building shapefile example, figure 5.7 shows that the building shapefile was of polygon type and that the PostGIS type is multipolygon of dimension 2. Figure 5.8 shows the SQL statements generated by the conversion of the building shapefile.

After converting the seven shapefiles to their corresponding SQL statements suitable for populating the database, these SQL statements will be introduced in the query execution window of the PostGreSQL DBMS which will ensure data storage in their corresponding tables. Figure 5.9 shows the SQL statements introduced in the query execution window to insert data into tables.
5.4 Query examples

In this section an experimental evaluation of our database is performed by formulating queries associated with services requested by the pedestrian described in the application scenario. Three services have been requested by the pedestrian: 1) to take him to the hello mall, 2) to take him to the nearest shoes shop and 3) to take him to UTC building and to alert him if there is a restaurant in less than five meters. The three services requested by the pedestrian are associated with three types of queries respectively: 1) getlocation queries which retrieve the location of an object at a specified time, 2) Nearest neighbours’ queries that retrieve nearest objects relative to a specified object 3) and range queries that retrieve objects that are within a specified distance from a specified object. As the data models and the database described in this research are supposed to support SAMS, we want to assess if queries associated by these services can be answered by the designed database.

5.4.1 GetLocation queries

In order to formulate in the database the query “take me to the hello mall” requested by the pedestrian, a functional decomposition approach was adopted that consists of splitting the query into levels of granularity appropriate for available computation functions. Therefore, by splitting the query into parts, each part of the query can be addressed independently and be combined. This approach is suitable because it gives an insight of how the query can be ad-
dressed by answering its parts with available functions in such a way that the query can be reconstructed. Based on operations defined in section 4.2.3, the query was split in the following two queries: “find the current location of the pedestrian” and “find the location of the hello mall”:

**Query 1: Find the current location of the pedestrian:**

```
SELECT POSITION(pt.startloc, NOW), DIRECTION(pt.startloc, NOW), ps.segnamee
FROM pedestriantrajectory AS pt, pedestriansegment AS ps
WHERE pt.pedid='1' AND pt.locsegment=ps.segid
```

The meaning of expressions used in the above query is as follows:

- **POSITION** (pt.startlocation, NOW): is an operation defined in section 4.2.3 that takes the last location update (startlocation) of the pedestrian and calculates its location at a time \( t \); \( t \) was replaced by “NOW” to specify the time at which the query was executed;

- **DIRECTION** (pt.startlocation, NOW): as defined in section 4.2.3, this is an operation that returns the direction of a pedestrian at a time \( t \) and the “NOW” argument indicates the time at which the query is executed.

- **ps.segname**: specifies the name of the segment on which the pedestrian is moving;

- “pedestriantrajectory” and “pedestriansegment” are the two tables that store pedestrians’ trajectories and segments respectively;

- **pt.pedid =‘1’** is the identification of the pedestrian in the database;

- **pt.locsegment = ps.segid** is a join condition between the pedestrian trajectory table and the pedestrian segment table.

Because the two operations **POSITION** and **DIRECTION** are not available as implemented functions that we can call from the database, the verification of this query in the database is not feasible. However, to show that once we have these functions the query is feasible, we can query the database to retrieve the last update location and direction in the database because these are values that the above mentioned operations will use to calculate the new values that correspond to a specific time. Therefore, the last update location and direction are retrieved by the following query:

```
SELECT pt.startloc, pt.locdirect, ps.segnamee
FROM pedestriantrajectory AS pt, pedestriansegment AS ps
WHERE pt.pedid='1' AND pt.locsegment=ps.segid
```
5.4. Query examples

The result of this query is shown in figure 5.10; the answer of the query returns the last update location geometry, the direction of the pedestrian which is NW (North-West) and the name of the segment he is travelling on.

To verify that the answer retrieved by the query is correct, we can use the shapefile that contains the points location of pedestrians. Figure 5.11 shows the attributes specifying the location of the pedestrian ped = ‘1’ (is a point whose spatial information is stored in the shape column), his direction is NW and he is travelling on segments with segid = ‘4’ and the segments attributes table shows that the segid = ‘4’ is called ‘holstraat’. Therefore, the database is able to retrieve the last update of the location of the pedestrian together with other related information such direction and segment on which he is travelling.

**Query 2: Find the location of the Hello mall:**

As the hello mall is a static spatial object, none of the spatio-temporal operations is needed in order to formulate the query in the database; its location query formulation is as follows:

```sql
SELECT ps.segname, b.addressnum, b.shape
FROM building AS b, pedestriansegment AS ps
WHERE b.buildname = 'Hello mall' AND b.refsegment = ps.segid
```

Figure 5.12 shows the result of query requesting the location of the Hello mall, and figure 5.13 shows the verification of the result in ArcMap.
Figure 5.11: Verification of the query retrieving the last update location and direction of a pedestrian

Figure 5.12: Query retrieving the location of the Hello mall
5.4. Query examples

5.4.2 Nearest neighbours queries

The nearest neighbour query associated with the service requested by the pedestrian is “take me to the nearest shoes shop”; its formulation in the database is shown in query 3:

**Query 3: Where is the nearest shoes shop?**

```
SELECT b.buildname, b.addressnum, ps.segname
FROM building AS b, service AS s, pedestriansegment AS ps, pedestriantrajectory AS pt
WHERE b.serviceid = s.serviceid AND b.refsegment = ps.segid AND s.servicetype = 'Shoes shop' AND pt.pedid = '1'
ORDER BY DISTANCEMS(pt.startloc, b.geom, NOW)
LIMIT 1
```

As the testing dataset has service information that stops at the shop level and not at the type of shops, the execution of this query in the designed database can be shown by only requesting the nearest shop. As the query request the nearest shop from the current location of a pedestrian, the DISTANCEMS operation defined in section 4.2.3 was used to calculate the distance.
between a moving point geometry (pedestrian location) and a static geometry at a time ‘NOW’ which specifies the time at which the query was executed. However, as this function is not available as an implemented function, we can answer this query using the last update in the database and by applying the DISTANCE operation defined again in section 4.2.3 which calculates the distance between two static geometries.

Figure 5.14 shows how the query first search all the building registered as shops; and figure 5.15 shows how the query orders building registered as shops based on the distance between them and the last update location of the pedestrian, and retrieve the first one which will be the nearest shoes shop. The query returns the name of the building, its address number and the segment on which the shop is accessible.

5.4.3 Range queries

In order to provide the service “take me to the UTC building and alert me if there is a restaurant in less than five meters” we adopt again the approach of functional decomposition based on available operations and firstly, to retrieve the location of the UTC building and secondly, to continuously search a restaurant that are in less than five meters from the location of the pedestrian. The first query for retrieving the location of the UTC building is formulated in query 4.
5.4. Query examples

Figure 5.15: Query retrieving the location of the nearest shop

**Query 4:** Where is the UTC building?

```sql
SELECT ps.segname, b.addressnum, b.geom
FROM building AS b, pedestriansegment AS ps
WHERE b.serviceid = s.serviceid AND b.refsegment = ps.segid
AND s.serviceType = 'shop' AND pt.pedid = '1'
ORDER BY ST_DISTANCE (pt.startloc, b.geom)
LIMIT 1
```

The second query that searches continuously a restaurant in less than five meters from the location of the pedestrian is formulated in query 5. However, as the query has to be re-evaluated continuously and not only at the database update time, it is not possible for the database to execute the query on each clock tick because the database itself does not store pedestrians' directional positions on each clock tick. According to [GS05], these types of continuous queries are executed with the help of an evaluation algorithm that computes the answer to continuous query just once in the form of a set of tuples annotated with time stamps. For each tuple, its timestamp indicates the period of time during which it belongs to the result. When time progresses, tuples whose time period is entered are added to the answer set and tuples whose time period expires are removed from the answer set. [GS05] Specifies that once an explicit update of the location is made in the database, the first calculated set of tuples become invalid and the continuous query is re-evaluated again to make a new result set of tuples. The query that has to be continuously re-evaluated is shown in query 5.
Figure 5.16: Query retrieving the location of the UTC building

**Query 5:** Continuously alert me if there is a restaurant in less than five meters?

SELECT b.buildname, b.addressnum, ps.segname  
FROM building AS b, pedestriansegment AS ps, pedestriantrajectory AS pt,  
service AS s  
WHERE b.serviceid = s.serviceid AND s.servicetype = 'restaurant' AND  
b.refsegment = ps.segid AND b.serviceid = s.serviceid  

The query search for buildings registered as “restaurant” and apply again the operation DISTANCEMS which calculates a distance between a moving object (e.g. pedestrian) with a static object (e.g. building). The condition \(< 5\) was applied to the function to specify that the query want only restaurant that are in less than five meters. However, due to lack of functions implementing the operation DISTANCEMS, the query was executed using the last update in the database and the function ST\_DISTANCE provided by PostGIS that calculates the distance between two spatial static objects. This was done to show that once the function DISTANCEMS is available, the query can be answered. Figure 5.17 shows that the result of the query did not bring any restaurant; this is not because the query is not correct, but it is because there was no restaurant in this 5 meters distance. To prove its correctness, the same query was executed again to search restaurant that are in less than 60 meters and this time a “Mac donalds” restaurant was found on street “holstraat” at the building number 10 as shown in figure 5.18.

The other query associated with services requested by the pedestrian is to find the location of an electronic shop within 200 meters from the location of the pedestrian; this query is formulated in query 6.
5.4. Query examples

Figure 5.17: Query retrieving the location of a restaurant in less than five meters

Figure 5.18: Query retrieving the location of a restaurant in less than sixty meters
Query 6: Find electronics shops that are within 200 meters?

In order to find all electronics within 200 meters from the current location of the pedestrian, the query searches for all buildings that provide electronic shop service and retrieve those only satisfying the condition of being within 200 meters from the current location of the pedestrian; the formulation of the query is as follows:

```
SELECT b.buildname, b.addressnum, ps.segname
FROM building AS b, pedestriansegment AS ps, pedestriantrajectory AS pt, service AS s
WHERE b.serviceid = s.serviceid AND s.servicetype = 'electronic shop' AND b.refsegment = ps.segid AND pt.pedid = '1' AND DWITHIN(POSITION(pt.startlocation, NOW), b.geom, 200)
```

Query 6 uses one spatial operation DWITHIN and one spatio-temporal operation POSITION defined in section 4.2.3; The DWITHIN is a spatial operation that takes three arguments (two geometries and a distance) and returns TRUE if the two geometries are within the specified distance one another. Therefore, to check if the distance between the current position of a pedestrian and an electronic shop is within 200 meters, the POSITION function was applied in order to calculate the position of the pedestrian at the time the query was executed, and its result will be used as one argument of the DWITHIN function. However, as the function POSITION is not available and as the testing dataset has service information that stops at the shop level and not at the type of shops, we used the last update location in the database and the service “electronic shop” was replaced by “shop” in the query as shown in figure 5.19.
5.5 Conclusion

The objective of this chapter was to implement a database that can be used by SAMS for supporting pedestrian region-based navigation systems. Data storage structures were defined by transforming the logical schema presented in sub-section 4.4.5 into the physical schema presented in section 5.2. For testing purpose, a testing dataset was built in section 5.3.2 and loaded in the database. Finally, relevant queries that are supposed to be supported by the database were formulated in section 5.4.

As the second research question “How to design and implement a database that can manage trajectory data associated with pedestrian region-based navigation and answer relevant types of queries?” was divided in three sub-questions: “How to design a conceptual data model and a logical model?” “How to design the physical model?” and “How to effectively answer queries associated with services needed by pedestrians?”; by defining data storage structures and by answering relevant types of queries, chapter 5 answered the second and the third sub-questions respectively. Therefore, as chapter four and chapter five answered the three sub-questions of the second research question, the second objective of this research which is “To design conceptual, logical and physical data models that represent trajectory datasets and other relevant data required by SAMS to support pedestrian region-based navigation” is also achieved.
Chapter 6

Discussion and conclusion

The objective of this chapter is to discuss and make a conclusion about the contribution of this research as well as to give recommendations for future works. A discussion based on research questions addressed by this research is made in section 6.1; a conclusion of the achievement of the research is presented in 6.2 and some recommendations for future works are given in section 6.3.

6.1 Discussion

In this section, a discussion is done to verify if the research questions have been answered and if the objectives of this research have been achieved. The discussion is based on the research questions presented in section 1.2.2, that the research was supposed to answer in order to achieve its objectives presented in section 1.2.1.

6.1.1 Discussion on research questions

The first research question that required an answer was: “What are the data about pedestrians’ physical world required by SAMS in order to support pedestrian region-based navigation in urban areas? How to model this data?”. This question was addressed based on an application scenario presented in section 3.2 and on its analysis presented in section 3.3. The application scenario sketches a pedestrian assisted by a SAMS running on his mobile phone in an unfamiliar urban area, and its analysis was done in order to find services requested by the pedestrian and data required by SAMS in order to satisfy the pedestrian’s requests. The analysis done at the service requester level (the pedestrian) and at the service provider level gave us the data required by SAMS in order to support pedestrian region-based navigation.

The analysis reveals that for supporting pedestrian region-based navigation, SAMS require three types of data which are spatial data, spatio-temporal data and non-spatial data. Based on this analysis, a model representing classes
of objects required and their relationships was built in section 3.4; the model was built using UML class diagrams because it is the standard modelling language for object-oriented applications and also because of its rich and powerful syntax that allow modelling properly the semantics of a whole database system. In summary, the first research question was answered effectively as data required by SAMS for supporting pedestrian region-based navigation was found and described in an objects classes model.

The second research question was “How to design and implement a database that can manage trajectory data and other relevant data associated with pedestrian region-based navigation and process relevant types of queries?” This question was addressed by splitting it in four sub-questions as presented in section 1.2.2.

- The first sub-question was “How to design a conceptual data model and a logical model that represent required data?” The design of the conceptual data model presented in section 4.3 expresses data requirement of SAMS for supporting pedestrian region-based navigation; it was designed using UML class diagram to describe relevant objects classes, their attributes, their associations as well as their behaviours. The conceptual model defines the contents of a database that can be used by SAMS for supporting pedestrian region-based navigation. The conceptual model was transformed into a logical model based on transformation rules and by using again UML class diagram. The logical model is presented in section 4.4 and describes how relevant data will be represented in PostGreSQL \PostGIS as the chosen DBMS; in the logical model, relevant data are described according to data definition rules of PostGreSQL \PostGIS. Therefore, the design of the conceptual model and logical model answered the first sub-question.

- The second sub-question was “How to design the physical model that store required data?” The physical model designed in section 5.2 is a result of the transformation of the logical model. The transformation was done using EA to transform the UML class diagram that represents the logical schema into a DDL file that encapsulates SQL statements that define data storage structures specific to PostGreSQL. These SQL statements were executed into PostGreSQL and give results to tables with their corresponding columns, primary keys and foreign keys. However, the tables created by the SQL statements of the DDL file were lacking columns that store geometries due to lack of spatial data types support by EA that generated the DDL file. Columns that store geometries were finally added to appropriate tables using the AddGeometryColumn function provided by PostGIS. Therefore, by designing suitable data storage structures for data required by SAMS, the second sub-question was then answered.

- The third sub-question was “How to effectively answer queries associated with services needed by pedestrians?” This research question was addressed by formulating relevant types of queries associated with services requested by pedestrians found during the analysis done in section
3.3.3. By formulating these queries and by defining required operations in order to answer them in the designed database, this sub-research question was therefore answered.

In summary, by giving answer to the above four sub-questions, the second research question was also answered since conceptual, logical and physical data models have been designed and relevant types of queries formulated.

6.1.2 Discussion on adopted approaches

To achieve the objectives of this research, adopted approaches present benefits as well as some limitations. For database representation of trajectory data associated with pedestrian region-based navigation, an approach based on storing a motion vector rather than each position of the pedestrian was adopted and present the following benefits:

- Database representation of pedestrians’ trajectories by storing a motion vector rather than storing each position, minimizes significantly the frequent update of positions as it would have been if the position was updated continuously at each clock tick.

- By storing a motion vector which is a positional function that changes its value with time automatically, it is possible to predict the near future position of a pedestrian.

   As the two benefits consider the advantages of storing a motion vector, questions can arise for the updating advantage of the motion vector because it needs also to be updated. We argue that even though it is still necessary to update the motion vector, the update frequency of this vector will be smaller than updating each position; and for a database managing a large number of pedestrians, the storage of each position for each pedestrian becomes not feasible as it would result in a very high database update load, whereas the update of the motion vector is done once the difference of the real location and the location stored in the database reaches a certain threshold.

   However, the above mentioned benefits are associated also with the following limitations:

- Update of the motion vector is overwritten to the existing one, therefore, there is no way to retrieve the history of movement of pedestrians. This makes the database not suitable for location based applications having interest in the storage of history of movement or in post-processing (e.g. data mining) purposes.

- The database designed is limited to support movement associated with 0-dimensional (point) moving objects; therefore, the database cannot support applications dealing with moving region objects;
The above limitations are associated with the approach adopted of using the motion vector for representing pedestrians’ directional positions; however, even though this makes the database unable to record the history of pedestrians’ movement and to support applications where the extent of the moving object is also of interest, we argue that this approach is still the most appropriate for pedestrian region-based navigation because details about pedestrians’ movement that SAMS require such as the direction, the speed, the current position, the expected near future position of a pedestrian are all captured in the motion vector as a function of time.

6.2 Conclusion

In this research project, a data model that describes data requirements of SAMS for supporting pedestrian region-based navigation was defined and a database was designed to manage pedestrians’ trajectories data and other relevant data required for supporting pedestrian region-based navigation. The contribution of this research can be summarized in the following points:

- An information model was designed to capture requirements for supporting pedestrian region-based navigation;
- A data model was defined for database management of pedestrians’ trajectories and other relevant data;
- A conceptual model was designed using UML class diagram to represent object classes that are relevant for SAMS that support pedestrian region-based navigation in urban areas; a logical model was designed using EA to represent relevant objects according to data definition of PostGreSQL PostGIS; and a physical schema was designed using SQL of PostGreSQL to define tables, their columns as well as primary and foreign keys;
- Relevant types of queries were formulated and some executed in the database, and required operations for the execution of other types of queries were also defined.

This research reveals that, by modelling the pedestrian physical world and by representing pedestrians’ trajectories in database using motion vectors (i.e. positional functions of time) with suitable computation functions, spatial and spatio-temporal queries associated with services requested by pedestrians in urban areas navigation, can be answered.

6.3 Recommendation

- Firstly, we recommend the development of datasets suitable for pedestrian navigation which are not available for the moment; these datasets must be
enriched with relevant information for pedestrians like footpaths, access paths to buildings, paths within buildings, path within pedestrians’ zones, and pedestrians’ landmarks or other features appropriate for pedestrians’ guidance. These datasets can then be used for automatic generation of appropriate routes, directions and maps for pedestrian navigation systems.

- Secondly, we recommend the development of functions that implement the spatio-temporal operations defined in this research as well as other relevant functions that can be used to address queries related to pedestrian navigation.

- Thirdly, we recommend also the development of region-based navigation algorithms for the generation of routes and guidance instructions based on pedestrian navigation datasets.

- Finally, as this research was done as a part of a main research for designing Spatially-Aware Map Services for pedestrians, we recommend the development and implementation of services that will satisfy specific contextual requests of pedestrians by making use of the models and database designed in this research.
Bibliography


Bibliography


[MS05b] Alexandra Millonig and K. Schechtner, *Decision loads and route qualities for pedestrians - key requirements for the design of pedestrian navigation services*, 3rd International Conference on Pedestrian and Evacuation Dynamics (Wien, sterreich), 2005.


[RSV] Philippe Rigaux, Michel Scholl, and Agnes Voisard, *Spatial databases with application to gis*, Mogan Kaufmann.


Bibliography


Appendix A

A.1 SQL for tables creation
A.2 SQL for Geometry columns creation
A.3 SPATIAL_REF_SYS and GEOMETRY_COLUMNS tables definition
A.3. SPATIAL_REF_SYS and GEOMETRY_COLUMNS tables definition

Figure A.1: SQL statements creating tables

```
CREATE TABLE addresstext ( rangeid integer NOT NULL, from_integer NOT NULL, to_integer NOT NULL, state char(50));
CREATE TABLE area ( areaid integer NOT NULL, areaname char(50) NOT NULL, type char(50), refsegoid integer NOT NULL, addressid integer NOT NULL, state char(50), postcode char(50));
CREATE TABLE building ( buildid integer NOT NULL, buildingchar(50), buildingchar(50), addressid integer NOT NULL, postcode char(50), addressid integer NOT NULL, postcode char(50));
CREATE TABLE landmark ( landmarkid integer NOT NULL, landmarkname char(50), landmarktype char(50), addressid integer NOT NULL, postcode char(50), addressid integer NOT NULL, postcode char(50));
CREATE TABLE node ( nodeid integer NOT NULL, nodechar(50), nodechar(50), addressid integer NOT NULL, postcode char(50), addressid integer NOT NULL, postcode char(50));
CREATE TABLE road ( roadid integer NOT NULL, roadname char(50), roadtype char(50), addressid integer NOT NULL, postcode char(50), addressid integer NOT NULL, postcode char(50));
CREATE TABLE waterfeature ( waterid integer NOT NULL, watername char(50), addressid integer NOT NULL, postcode char(50), addressid integer NOT NULL, postcode char(50));
CREATE TABLE addresstext ADD CONSTRAINT Pk_ADDRESSTEXT PRIMARY KEY (rangeid);
ALTER TABLE area ADD CONSTRAINT Pk_AREA PRIMARY KEY (areaid);
ALTER TABLE building ADD CONSTRAINT Pk_BUILDING PRIMARY KEY (buildid);
ALTER TABLE landmark ADD CONSTRAINT Pk_LANDMARK PRIMARY KEY (landmarkid);
ALTER TABLE node ADD CONSTRAINT Pk_NODE PRIMARY KEY (nodeid);
ALTER TABLE road ADD CONSTRAINT Pk_ROAD PRIMARY KEY (roadid);
ALTER TABLE waterfeature ADD CONSTRAINT Pk_WATERFEATURE PRIMARY KEY (waterid);
ALTER TABLE building ADD CONSTRAINT Pk_BUILDING_FOREIGN KEY (refsegoid) REFERENCES road (buildid);
ALTER TABLE landmark ADD CONSTRAINT Pk_LANDMARK_FOREIGN KEY (nodechar, nodechar) REFERENCES road (landmarkid);
ALTER TABLE node ADD CONSTRAINT Pk_NODE_FOREIGN KEY (rangeid) REFERENCES road (nodeid);
ALTER TABLE road ADD CONSTRAINT Pk_ROAD_FOREIGN KEY (rangeid) REFERENCES road (roadid);
ALTER TABLE waterfeature ADD CONSTRAINT Pk_WATERFEATURE_FOREIGN KEY (refsegoid) REFERENCES road (waterid);
```

Figure A.2: SQL statements adding geometry columns to tables

```
SELECT adddegeometrycolumn(‘gatera1956’, ‘pedestrans’);  
SELECT adddegeometrycolumn(‘gatera1966’, ‘pedestrans’);  
SELECT adddegeometrycolumn(‘gatera1966’, ‘building’);  
SELECT adddegeometrycolumn(‘gatera1956’, ‘waterfeature’);  
SELECT adddegeometrycolumn(‘gatera1966’, ‘area’);  
SELECT adddegeometrycolumn(‘gatera1956’, ‘landmark’);  
SELECT adddegeometrycolumn(‘gatera1966’, ‘point’);  
```

94
Appendix A.

Figure A.3: SQL creating SPATIAL_REF_SYS and GEOMETRY_COLUMNS tables