TOWARDS DATABASE SUPPORT FOR MOVING OBJECT DATA

Nirvana Meratnia, February 2005

Samenstelling van de promotiecommissie:

Promotoren:

Prof. dr. Peter M. G. Apers (Promotor)

Assistent-promotor:

Dr. ir. Rolf A. de By (Assistant-promotor)

Leden:

Prof. dr. Bart van Arem, Twente University, The Netherlands
Dr. Henk M. Blanken, Twente University, The Netherlands
Prof. dr. Christian S. Jensen, Aalborg University, Denmark
Prof. dr. Peter van Oosterom, Delft University, The Netherlands

CTIT Ph.D. Thesis Series No. 05-69

Center for Telematics and Information Technology (CTIT)

P.O. Box 217 - 7500 AE Enschede - The Netherlands

ITC Dissertation Series No. 120

International Institute for Geo-information Science and Earth Observation (ITC)

P.O. Box 6 - 7500 AA Enschede - The Netherlands

SIKS Dissertation Series No. 2005-04

The research reported in this thesis has been carried out under the auspices of SIKS, the Dutch Graduate School for Information and Knowledge Systems.

ISBN: 90-365-2152-1
ISSN: 1381-3617
Cover designed by: Fariba Bandi
Printed by: PrintPartners Ipskamp

TOWARDS DATABASE SUPPORT FOR MOVING OBJECT DATA

PROEFSCHRIFT

ter verkrijging van
de graad van doctor aan de Universiteit Twente,
op gezag van de rector magnificus,
prof. dr. W.H.M. Zijm,
volgens besluit van het College voor Promoties
in het openbaar te verdedigen
op woensdag 23 februari 2005 om 13.15 uur

door

Nirvana Meratnia

geboren op 1 September 1975
te Tehran - Iran

Dit proefschrift is goedgekeurd door:

Prof. dr. Peter M. G. Apers (promotor)
Dr. ir. Rolf A. de By (assistent-promotor)

Preface

My interest in moving objects dates back in 1999, while doing an elective course during my MSc at ITC (International Institute for Geo-information Science and Earth Observation). The course was about time and GIS and moving object was one of the related topics. My first impression was that the term is funny. I was not so wrong about that, because later on whoever asked me what my research is about, had a funny face after hearing the term. Perhaps, people who know me a bit will say, oh, right, that is why you chose it. I am afraid to disappoint you. Soon, it became clear that it is very challenging, though interesting field of research. What is really nice about it is its relation to human beings, the concept is somehow indispensable part of our daily lives, it seems to be very tangible; we are all a moving object after all. However, when it comes to define it and solve its related problems, it is not as easy as it sounds.

At the time, not many people were working on this field. Not many references were available, while lots of open research issues existed. Reading technical reports from Chorochonos project was great inspiration for me. They addressed various issues, showing the diversity of research topics related to the moving object. Besides, they were really good pieces of work. After doing my master thesis on 'implementation of data structures and operations for moving objects modeling', I knew for sure that is the direction I want to follow for my Ph.D.

I was lucky to be surrounded with people who shared the same interests and enthusiasms and found the topic worthwhile study. Their professional and scientific view on research revealed to me the importance and enormous potential of this field of research. Due to the diversity involved, the main direction of the research changed many times. It was difficult to stayed focus and to decide what to do first. I was feeling like a kid who is left alone with a room full of colorful and amazing toys, having no idea what each of them is, while they were all appealing and inviting to try and enjoy. Since, it was impossible to solve everything in four years, finally it was decided to focus on particular problems, preferably starting

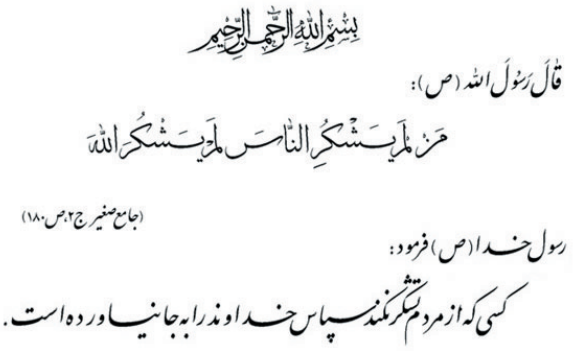
from simple cases and build a strong foundation, which can be used as a basis for further research and deal with more complicated cases later on. Despite all ups and downs and changes in topics to be addressed, here it is; a small window to moving objects' life.

Obviously, different views can be seen from different windows, depending on where they are located, how they are built and who is looking through. This small window is no exception. It is designed to show four problematic aspects of moving objects, namely, uncertainty associated with moving object data, trajectory representation, trajectory compression, and similarity measures for moving object trajectories. If time was allowing, many more things could have been done. However, this is also encouraging to know that this is not the end, just a beginning.

Hope what one sees through this piece of work is inspiring enough to encourage the one to look for ways to first create and then open the windows to colorful, interesting and dynamic life of the moving objects.

Nirvana Meratnia

Acknowledgements


بسم الله الرحمن الرحیم

قال رسول الله (ص):

من لم یشکر الناس لم یشکر الله

(جامع صغیر ج۲، ص۱۸۰)

رسول خدا (ص) فرمود:

کسی که از مردم تشکر نکند سپاس خداوند را به جا نیاورده است.

Mohammad (peace be upon him) said:
Who does not thank people, has not acknowledged God.

Thanks be to God, who makes everything possible, who raised me up when I was down, who was with me when no one else was around, who took me through most difficult moments when nobody could, who brought me to this stage and will take my hand and walk me through all days ahead. And then,

There are many people who helped me in one way or the other during and prior to these years that I spent on my research. It would be impossible to name each and everyone of them. Although, I mention some whom I worked closely with, my thoughts and appreciations are with all.

I would like to thank my promotor, Prof. dr. Peter M. G. Apers for giving me the opportunity to conduct my research. His support and kind words have been continuous encouragement and great inspiration. His pleasant attitude in facing

and solving problems and finding alternatives has always been a sign that things get better. Thank you, Peter.

My dream of further study had come to an early end if Dr. ir. Rolf A. de By did not kindly accept to supervise and lead me through. His ever-lasting support and outstanding guidance and advice accompanied me through out the whole period of my Ph.D research as well as my MSc. His professional view on research issues and critical comments opened new windows before my eyes. He thought me great deal about how research should be done, how a researcher should extend its own view and still be focused. I learnt from him how to be patient with one who is far from your standards but is willing to learn and how to help such person to find its own way. Rolf, I would like to express my sincerely gratitude for all you have done for me and the most importantly for being such a great teacher and a good friend.

Thanks are due to Prof. dr. Bart van Arem, Dr. Henk M. Blanken, Prof. dr. Christian S. Jensen, and Prof. dr. Peter van Oosterom who honored me by accepting to be in my graduation committee.

I believe the work environment is as important as the work itself. Members of Database (DB) as well as Information System (IS) helped to make a friendly and pleasant atmosphere. I will remember the nice moments we had in coffee breaks, birthdays, lunches (specially fitting as many as possible persons at a small round table), social events, and 'almost weekend meetings' with joy. We shared so much fun. Thank you for being such good colleagues. My special thanks to Sandra Westhoff and Suse Engbers for taking care of all administrative tasks and being there for each and every colleague.

I would also like to thank Prof. dr. Wolfgang Kainz, whom I started my research with, prior to his departure to University of Vienna. His interest in the topic made it possible to start this work.

My thanks fly to ITC staff, especially members of GIP department and library staff, who always received me with open arms, smiles and kind words. They made me feel welcomed and treated me as I have never left.

During this period, as always, I received enormous supports, words of encouragement and hope from my family and friends. Words can not express my true appreciation. What each of them have given me is so valuable and naming them will make a long long list. I am sure, you all know what each of you mean to me and how thankful I am.

Mum and Dad, I wanted to write to you at the end to be able to write as much

as I want. Now, I do not even know what to say. I do not know how to thank you and for what. Your sacrifices, dedications, kindness, and supports are just few to mention. What you have done for me and have given me are countless. I could not have come this far without you. You believed in me and gave me the liberty to choose my life and at the same time supported me to do what I wanted. I know it was so difficult for you to let your only child go, but you did it to let my dreams come true. No words can express how grateful I am. I am in debt to you everyday of my life. Many many thanks.

I do not know what I may appear to the world;
but to myself, I seem to have been like a boy
playing on the seashore, and diverting myself
and now and then finding a smoother pebble or a
prettier shell than ordinary, whilst the great
ocean of truth lay all undiscovered before me.
Sir Isac Newton (1642–1727)

Contents

List of Figures

List of Tables

Chapter 1

Mobility Scenarios

True knowledge exists in knowing that you know nothing; and that makes you the smartest of all.

Socrates (470 – 399 BC)

Any physical object's existence brings naturally and automatically with it that at any point in time, it is located somewhere. In the dynamic world in which we live, space is a property that varies over time. Space and time have become such indispensable elements of human beings' daily life, that we almost never think about them, and sense them while having difficulty in describing them.

In the world that is evolving rapidly, *mobility* is an important factor of people's life. The Internet, wireless networks, positioning technology as well as personal devices such as PDAs and cell phones and their related services are being advanced and improved day by day. Over the past few years, rapid advances in miniaturization and personalization of electronic devices have taken place, which consequently have resulted in major price reductions. Performance improvement of general computing technologies on the other hand have made it possible to introduce services that previously were even impossible to think of. The ultimate goal of all these advances is to satisfy the consumers' rising expectations. This can be achieved if information can be timely provided in the right place. In the coming years, delivering appropriate timely (personalized) services based on the position of mobile consumers will become increasingly important. Provision of such information will benefit the consumers in various ways, for instance, in better awareness of their surroundings, in identifying potential problems and bottlenecks, which in turn helps them to more efficiently and accurately plan to tackle them, in better management of available resources and planning for possibly sharing them for efficiency reasons.

Despite some success in fulfilling consumers' requirements, there is still a long way to go and new serious challenges are ahead. One of these challenges is the lack

of database support at present. This stems from the fact that existing databases, which are one of the key elements in making more practical and accurate information available, are at their best good in handling static situations, while the concept of mobility brings up a new set of requirements, dealing with dynamic situations.

The central issue in any mobility scenario is the object whose position continuously changes, i.e., *the moving object*. Although the concept of moving object is rather new in the area of spatio-temporal databases, the variety of applications that may benefit from it is enormous. Urban traffic, especially commuter traffic, and rush hour analysis; fleet management and car theft protection; monitoring animal migration; analysis of shopping behaviour (in a mall or city centre); patient tracking in a hospital; location-based services, such as tourist information, localized advertising, emergency services; these are just a few examples to mention. The potential is simply enormous.

This research points out some of challenges for applications dealing with moving objects. Some of the current challenges and problems that we face and, consequently that we need to solve are addressed and practical solutions are proposed.

1.1 Database support is wanting

In a world facing information explosion, positioning technology is rapidly making its way into the consumer market, not only through the already ubiquitous cell phone but soon also through small, on-board devices in many means of transport and in types of portable equipment. It is thus to be expected that all these devices will start to generate an unprecedented data stream of time-stamped positions for the agents that carry them. This development does not depend on GPS technology alone: in-house tracking technology applies various techniques for up-to-date positional awareness, and adaptable antenna arrays can do accurate positioning on information obtained from calls by cell phones [27].

Thanks to these advances in positioning technology, which makes data about moving objects easily available, soon these objects have become one of the focuses of the spatio-temporal database community. However, in spite of its simple looks, the moving object concept has become a practical challenge in applications dealing with mobility, Internet technology, and Geographical Information Systems (GIS).

Databases have not very well accommodated moving object data in the past, as their design paradigm was always one of 'snapshot representation'. Their present

support for *spatial* time series is at best rudimentary. Consequently, database support for *moving object* representation and computing has become an active research domain, see for instance [6, 160, 54, 117, 8].

Database management systems (DBMSs) have a potential foundation for moving object applications, however, they are currently not used for this purpose and at the moment the aforementioned moving object application domains lack database support. The reason is that moving object databases require a set of critical functionalities to be integrated, and built on top of existing DBMSs [147]. What is needed in current real-world spatio-temporal applications is a small, robust, and highly expressive set of predicates, suitable for implementation based on off-the-shelf DBMS technology. The query processing schemes for the predicates and the accompanying indexing schemes should be supported by the implementation [137]. Over the years, various issues were identified as a set of capabilities that should be provided by a DBMS to effectively and efficiently support mobility and moving objects. Despite their individual nature, these required capabilities are somehow related and any improvement or problem in one will affect the rest. We enumerate a list of important challenges that current DBMSs are facing, concerning moving objects:

1. Data modelling and representation

   Data modelling aims at defining the data types, operations, and relations between them [53] to support application design. In other words, data modelling is the common name for the design effort of structuring a database. This process involves the identification of the kinds of data that will be stored in the database, as well as the relationship among these data types [28]. The requirements of moving object modelling are not fully covered by current data models. We illustrate this below.

   Regarding data modelling, an important question is how to represent a moving object. The efficiency of indexing and query processing methods is highly affected by the chosen method to represent the continuous nature of the moving object. Since computer systems cannot easily represent continuous phenomena, such phenomena must be approximated using finite structures. The approximation methods should faithfully represent the object movement and provide a basis for further analysis, especially because an inappropriate technique will increase the uncertainty. The data model has direct effects on storage space, performance, and access time. The data model defined

for modelling continuously changing locations should be simple, though expressive, and be easy to implement. The more expressive a model is, the closer to the real world the application will be, and the more semantics will be captured. However, the more expressive a data model, the more complex it may be [131]. Furthermore, since the moving object field of research is young, there is room for new concepts and techniques. Therefore, the data model should allow for possible extensions. On the other hand, the possibility of designing new data models based on already existing ones should be investigated.

In addition, the moving object concept brings a new dimension to the definition of operations defined in traditional data models. For instance, the traditional definition of distance between two objects often concerns the Euclidean distance. However, distance in moving object scenario is a time-dependent function, rather than a constant value. On the other hand, for network-constrained moving objects the use of plain Euclidean distance is not advisable, since the underlying network poses extra conditions on manoeuvrability of objects. This means that either existing operations should be revised and adapted to accommodate the moving object concept or new operations should be defined. In addition to the basic spatial and temporal data types and operations, which are supported by traditional data models, extra spatio-temporal data types and operations are needed. However, the question is what these extra data types and operations are, to fully support moving objects. This is an important issue since more powerful data models are the ones with more complete and expressive data types and operators and have the better strategy to reach the closure under the defined operations set. Furthermore, there is the issue of integrating such data types and operators with the DBMS. Data types can be integrated with the DBMS in three different ways, which from tightest to loosest are as follows: integration of data types into the DBMS kernel, using a database extension, and implementing data types as a layer on top of DBMS. The choice of integration method is a crucial one.

2. Query processing

   Most existing query languages are non-temporal and limited to accessing a single database state [30]. Traditional query languages such as SQL were not designed for querying time-varying spatial aspects. Processing moving object data requires new sets of spatial, temporal, and spatio-temporal operators to

be used in query processing. These new operators should be applicable for any kind of object, i.e., those with free movements as well as movement constrained objects. Movement of an object is either constrained by other objects or by the medium via which it is travelling. The question is how to design a query system on top of an existing query system to deal with the dynamic aspects of moving object data. Furthermore, moving object applications often have to use different databases to answer queries. This means that the query processing technique should account for delay, overhead, and inaccuracy [144].

On the other hand, the mobility of objects leads to the invalidity of query answers, simply because some or all of the spatial and temporal criteria that were true at the time of posing the query will be violated very frequently. Therefore, the posed query should be re-processed every now and then. Therefore, the question of when and how often the query should be re-evaluated arises.

3. Indexing

   Moving object databases often have huge amounts of data. Therefore, examining the location of each moving object in the database to answer queries results in high performance overhead. Thus, the location attribute should be indexed. However, straightforward use of spatial indexing is not feasible due to the fact that continuous change of the locations implies that the spatial index has to be continuously updated [149]. Constant updating of the indices is not feasible if not impossible due to huge computing resources required [65]. Therefore, spatio-temporal indexing techniques are required. Previous work on indexing spatio-temporal data concerns either past or present and future data. However, most of these approaches deal with spatial data changing discretely over time [104]. Therefore, an important question is how to index the continuously changing moving object data with satisfiable performance and acceptable cost.

4. Uncertainty handling

   The locations of moving objects are inherently imprecise [103]. This inherent uncertainty has various complications for database modelling, querying and indexing. The more accurate the record of moving objects, the better query results. However, this in turn may result in poor query performance. Therefore, a moving object data model must properly represent

moving objects with reasonable degree of precision, which does not harm query performance [64]. On the other hand, moving object applications may need query-imprecision support, due to imprecision associated with other notions, (e.g., traffic jam) used in the query definition [144]. One of the main research questions in this direction is how to build up new modelling and spatio-temporal capabilities needed for moving objects to handle the inherently imprecise data and their related query analysis, considering the fact that lowering uncertainty is costly.

5. Data mining and prediction

   One important database challenge is to find valuable information hiding in large amounts of data, such as moving object data. Moving objects often have repeated patterns of movement, which can be used for planning and management purposes. The objects' movement can have periodically repeated patterns, e.g., animal migration patterns, commuters, shopping patterns, or sudden patterns. Identifying both these patterns is the key for successful planning in the objects' environment. On the other hand, this identification can be used for classification of objects and consequently, providing appropriate services to objects, which share some profiles. However, due to the multi-dimensional nature of moving object data and dependency of its data to other objects as well as the underlying network conditions, if applicable, current data mining methods used in databases are not suitable for moving objects [64].

6. Keeping information up-to-date

   It is often assumed in existing databases that data does not change unless it is explicitly modified [149]. However, in mobility scenarios the moving object's location continuously changes even if the database is not directly updated. Due to continuously changing nature of moving object data, keeping such data up-to-date is a must. Continuous update of the database is impractical since the location is updated very frequently. On the other hand, the answer of queries may be outdated if data is not properly updated. Furthermore, assuming that the moving objects themselves are responsible for transmitting locational data updates via wireless networks, frequent updating would also impose a serious bandwidth overhead [148]. In addition, strategies are needed to handle possibility that a moving object becomes disconnected and cannot send updates, which results in incomplete and inaccurate data set.

This will bring the attention to an important issue of how to deal with the trade-off between the updating overhead and incomplete and inaccurate data set. One also should pay attention to the fact that too few updates leads to information loss and too frequent updates gives rise to storage and transmission issues. In short, due to continuously changing nature of moving object data, keeping such data up-to-date is a must.

7. Efficient storage mechanism

   Moving object applications have many objects to monitor, the monitoring process may be continuous, and the respective data acquisition rates may be high. Thus, an efficient storage mechanism is required. On the other hand, since not all the acquired data may be necessarily informative, to avoid wasting storage space, some compression mechanisms should be utilized.

8. Visualization

   Graphical visualization is a strong power in moving object applications. However, considering the continuous change of the object data, there is a great concern on how query answer can be visualized. Another question is what dimension to use for the visualization purpose. Considering a 3D environment increases the moving object challenges in all aspects. Working in a 3D environment is not only 3D visualization. Obviously it needs spatio-temporal analysis of the 3D data types, which are not supported in the existing DBMSs. On the other hand, mobile consumers want to see their graphic display updated all the time, with local information.

1.2 Moving object problems from the point of view of this thesis

The mentioned database challenges reveal that although some efforts have been done over the past years to tackle moving object related problems, much more remains to be done in this young field of research. Due to the variety of challenge introduced by the moving object concept, the potential for further research is big. Obviously, answering all the above mentioned issues in one single work is not easy, if not impossible. The distinction between those issues is only for clarification purpose; otherwise there is no clear boundary between each of the problem categories since they do affect each other. For instance, the data uncertainty will directly

affect the query processing and indexing, and the data representation method influences the visualization technique, and vice versa. This study addresses the following major issues:

- Uncertainty handling techniques for moving object data

  Failure of an application to best perform and to provide appropriate results has various reasons. One of these is uncertainty of the raw data that is entered into the system. Obviously, errors can occur in different phases of data acquisition, modelling, processing, or in information use. Researchers agree that most of the application errors are associated with the raw data, and is propagated in later stages. Therefore, this error should be reduced to the best possible extent. The corrected data, which contains less error, can be entered into and used by the application with a higher degree of confidence.

- Faithful trajectory representation

  Most, if not all, moving object applications work around central *where*, *when*, and *what* questions, which involve location, time, and object, respectively. Answering these questions requires a faithful determination of moving object trajectories. Existing techniques, however, fail to do so. Therefore, representing an object trajectory as accurately and realistically as possible, based on erroneous data is a must.

- Trajectory compression techniques

  It is expected that in the coming years large quantities of wireless, portable, on-line objects that are location-enabled will become part of people's daily life. Some predict that each of us will soon have tens of such objects [64]. This means that these objects may generate huge volumes of data. Storing all this data is neither possible, nor wise. This large amount of data results in various complications for storage, transmission, indexing, and query processing. One solution to the problem is using compression techniques. However, existing techniques coming from the data mining field are not suitable for moving object trajectories, as they are mainly for one-dimensional, short time-series and in absence of noise. Three properties that do not hold for trajectories. On the other hand, compression techniques used in GIS treat the trajectory as a simple line, forgetting its time component. Current methods are simply not so promising.

- Similarity measure for trajectories

  In moving object applications, in which tracking of objects is common, identifying objects that moved in a similar way or followed a similar movement pattern is desirable [139]. Such discovery results in the grouping of objects with similar patterns and finding their common spatio-temporal properties. Subsequently, appropriate services can be provided to them. However, this similarity search by no means is an easy task due to different sampling rates, inherent imprecision of object location, different trajectory lengths, shifts in location and/or time components [140, 20]. Therefore, similarity search techniques for moving objects trajectories are required.

1.3 Layout of the thesis

The remaining chapters of this thesis are organized as follows:

Chapter 2 presents an overview of the state-of-the-art in spatio-temporal data modelling, with specific emphasis on modelling moving objects. The chapter also addresses different data acquisition methods and techniques to acquire data about these objects. It also provides the informal definition of moving objects. Finally, it contains fundamental assumptions made in this work, and provides notations used throughout the chapters to come.

Chapter 3 motivates the need for reducing the error associated with moving object data before any further processing. Already available techniques mainly used for unconstrained moving objects are addressed. More intelligent methods are proposed for network-constrained moving objects to narrow down the associated error even more. Experiments are carried out to support the proposed methods.

Chapter 4 brings up the issue of faithful trajectory representation and failure of the two well-known and most often used interpolation techniques. It discusses the disadvantages of these two techniques and introduces a new method to overcome the problems and to represent the object's trajectory more realistic. Results of experiments back up the method.

Chapter 5 emphasizes the fact that dealing with huge volumes of moving object data requires compression techniques to eliminate not so informative data in a way that does not result in detrimental information loss. This research addresses both location and time, by proposing spatio-temporal compression techniques that support possibly erroneous trajectories with different lengths. Experimental results support these techniques.

Chapter 6 explains the need for similarity search for moving object trajectories. Different operators are proposed to present different levels of similarity between trajectories. Consequently, the similarity notion itself is based on different available data types resulting from these operators. Finally, the chapter reports on solid techniques to perform the similarity search.

Chapter 7 summarizes the research achievements, followed by a discussion of the lessons learnt. It also provides some directions for further study in this field.

The thesis outline is shown in Figure 1.1, in which the information flow and dependency between chapters are shown.

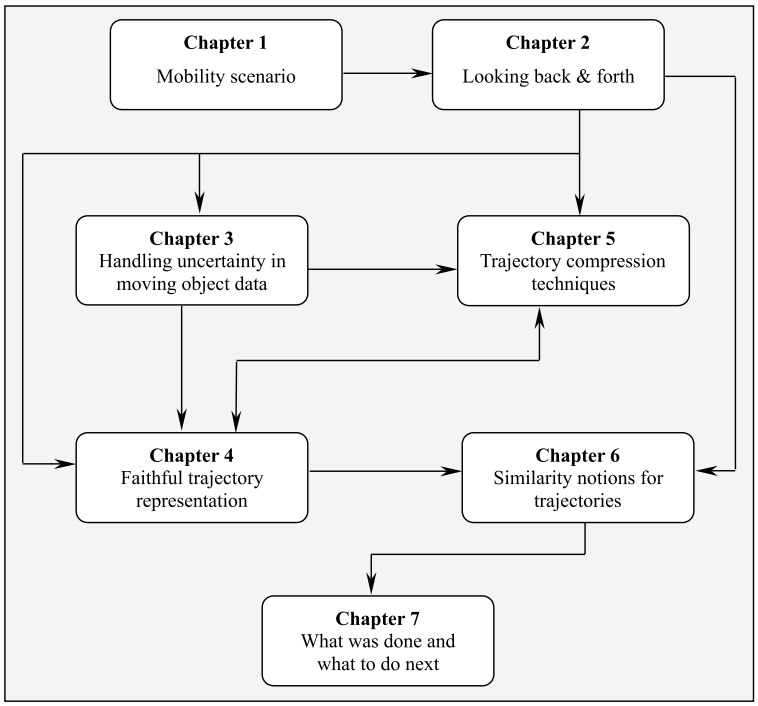


Figure 1.1: *Thesis layout.*

Chapter 2

Looking back & forth

You cannot teach a man anything;
you can only help him find it within himself.

Galileo Galilei (1564 – 1645)

For quite some time, members of the database community considered spatial databases and temporal databases to be new trends. Many database scientists identified and solved related problems. Spatial database research focused on modelling, querying, and integrating geometric and topological information in databases. Temporal database research concentrated on modelling, querying, and recording the evolution of facts under different notions of time and, thus, on extending the knowledge stored in databases about the current and past states of the real world [36]. Despite the many great results achieved, the satisfaction of database community did not last long. It soon realized that in reality, space and time are rarely, if at all, independent. Soon, new efforts were directed towards integrating both concepts in one database. Both the spatial database community and the temporal database community individually tried to extend their databases to support the missing concept. Integration of space and time means dealing with time-varying geometries, which soon became the main focus of a new branch of database, i.e., the spatio-temporal database . Looking at history of spatio-temporal data models reveals how this branch of databases has evolved.

2.1 Non-integrated spatio-temporal data models

In an early attempt in 1964, Berry [14] proposed a simple measurement framework for geographic data called the *geographical matrix*, which in its simplest form is represented as a two-dimensional table. In this table, columns represent places and rows represent characteristics. Time was added as a third dimension. Bullock in 1974 and Hagget in 1977 also took this approach [57]. Later in 1978, Sinton

observed that three factors of location, time, and theme are playing a role when studying real-world phenomena, such that one of these factors is usually fixed, one varies in a controlled manner, and the third is measured [121]. Since his observation was mainly about thematic maps, he only considered static objects and did not discuss objects moving through space [83].

Gradually research on spatio-temporal data models became systematic resulting in the following categories.

2.1.1 Time-stamped data models

During 1980s and beginning of 1990s, three time-stamping approaches were proposed to incorporate time into relational databases, namely, *(i)* time-stamping a relation [44], *(ii)* time-stamping individual tuples [124, 11, 26], and *(iii)* time-stamping individual attributes [45, 119, 43]. Gabbay & McBrien also tried to define a general extension to the relational algebra to include temporal semantics. They used the concept of *since* and *until* in their temporal logic [42]. Temporal relations were extensively discussed by Jensen & Snodgrass [67].

Parallel to the three time-stamping approaches in relational databases, various research was on the way to enhance GIS's spatial data models by inclusion of models of time. Egenhofer & Al-Taha studied gradual changes of topological relations between spatial objects over time and consequently proposed a data model to describe these [32]. Yattaw presented a classification of geographic movement using different models of time in conjunction with different spatial data types [154]. Moreira et al. in [93] proposed that models of time include the *linear model*, for processes where time advances step by step from past into the future, the *cyclic model*, for recurrent processes like weeks, months, and years, or the *branching model* to represent time dependencies between entities. The authors also mentioned, the *discrete model*, in which time is measured at certain points or intervals and variation is discontinuous between these points, hence, the structure of points in time are isomorphic to the natural numbers, versus the *continuous model*, in which time structure is isomorphic to real numbers and thus, contains no gaps.

In the GIS field, a tuple time-stamping approach has often been used to expand the schema of a relation by explicit temporal attributes that are used to describe the validity period of a whole relation [35]. The data models proposed by Armstrong in 1988 [12], Langran & Chrisman in 1988 [80], Beller et al. in 1991 [13], and Rafaat et al. in 1994 [111] are some examples to mention. In this kind of model, any single attribute change leads to duplicating the whole tuple. As

a result, multiple relations contain the information about an object. Continuous change of spatial objects cannot be properly modelled by this approach [35]. In the data model proposed by Armstrong, which is called the *snapshot model*, every layer is a set of temporally harmonized units of theme and the model represents states of one layer at different times. One of the main characteristics of this model is that layers at different time are not necessarily different [12]. Langran's model, namely the *space-time composite model*, expands the idea of Armstrong's model by overlapping multiple snapshot layers into one spatio-temporal composite layer. It conceptually describes the attribute change of spatial objects during a time interval. In case of updates, an object needs to be reconstructed. Consequently, geometrical and topological relationships should be updated as well [80]. A major problem in this model is that only either space or time can be variable at a time, but not both. Therefore, although the model allows to record change, it fails to capture mutation or movement. The model of Beller et al. is called the *Temporal Map Set* (TMS); it is a collection of layers representing time instances at which an event has taken place. In contrary to the snapshot model, the temporal map set model requires layers to be different at different times, therefore, the whole set shows subsequent states of one spatial unit during a specified period [13]. Rafaat et al. discussed spatial and thematic attribute changes in a GIS and consequently proposed a relational method to access spatial and temporal topologies [111]. In this model, attribute changes of an object are recorded using tuple-timestamping approach for various time intervals, until it is destroyed or becomes another entity. States of a single data layer is represented by a set of relations, containing topological, attribute, and a positional relation [7].

In 1992, Worboys explicitly brought up the idea of generalizing a spatial data model to become spatio-temporal [150]. Later in 1994, he proposed a model called the *spatio-temporal object model*, in which spatio-temporal objects are defined as so-called spatio-bitemporal complexes. In this model, spatial properties were represented by simplicial complexes. On the other hand, temporal properties were represented using bi-temporal elements attached to all components of spatial features [151]. The model allows to capture the change in attributes in both time and space. However, it can only represent discrete attribute change.

2.1.2 Process-based data models

Possibility of modelling spatio-temporal phenomena by events or processes were also being considered by researchers and data models such as the *dual ordered*

hierarchical structure by Whigham [142], the *Event-based Spatio-temporal Data Model* (ESTDM) by Peuquet & Duan [100, 101], the *object-oriented data model* by Raper & Livingston [115], *three domain data model* by Yuan [158], *extended-versioning data model* by Claramunt & Thériault [25] were proposed.

Whigham observed that time is relative, has an imposed order and has different resolutions. He therefore, proposed a dual ordered hierarchical structure where time and events are represented in their own hierarchies, placed on a spatial background reference. Events are placed in a hierarchy with locational reference. An appropriate time hierarchy then is linked with the events hierarchy to establish the time frames for the individual events. Links within the events hierarchy can be used to show casual relationships between events [7].

The event-based spatio-temporal data model was a raster-based model to represent spatio-temporal information about spatial change. The main idea behind this model was to distinguish between three representations of spatio-temporal data with respect to specific analytical tasks. With this representation, a triad of object-based, location-based, and time-based data was defined [100]. In this model, events are sequentially recorded. Each event is associated with a list of all changes that occurred since the last update [102]. The same as the TMS model, ESTDM clusters time-stamped tuples to show temporal evolution of a single event. It managed to outperform the temporal map set model in data efficiency and support for analysis of temporal patterns and relationships, since rather than storing each event independently, it stored changes in relation to the previous events. Generally speaking, the event-based model has its capabilities and efficiency to support both spatial and temporal queries [158], however, it suffers from the lack of comprehensive definition.

In the object-oriented data model of Raper & Livingstone, every spatio-temporal phenomenon is represented by a set of form, process, and material objects. Three-dimensional location and one-dimensional time are considered as attributes of spatio-temporal objects. This approach is similar to Worboys' space-time object model but Raper & Livingston emphasized the importance of a physical system and processes. However, while this model can handle point spatial objects well, it has difficulty in dealing with area data and topological relationships [158].

Yuan's three domain model defines semantical, temporal, and spatial objects in three separate domains. Geographical objects are represented by dynamically linking these three domains. Time is considered as an independent concept, instead of being an attribute of location. The data model is able to represent attribute

changes, static spatial distribution, static spatial changes, dynamic spatial changes, mutation of a process, and movement of an entity. The major advantage of this model is to have dynamic linking between the three domains of the model instead of having a pre-defined data schemata [159]. In practice, this model demonstrated capability to represent and analyze behaviour of spatio-temporal objects.

Although these three fairly recent data models are able to show transitions, they have difficulty in handling mutation and movement and answering queries about transitions. In addition, they are limited to raster or point-based models.

In 1995, Claramunt et al. represented a vector-based conceptual approach to incorporate time and topologic and metric dimensions into one data model. The obtained model aimed at representing changes among a set of objects. The proposed model, namely *extended-versioning data model* was based on an extension to the versioning concept. This concept implies that any attribute change leads to explicit storing of a new version of the information in the database. A sequence of these stored information is then used to describe successive events in the real world [25]. The main disadvantage of this model is that explicitly storing all the changes is costly, since for each change relationships among objects as well as the relationships among the versions should be stored.

It soon became clear that extension of spatial or temporal databases by adding temporal support to spatial databases or adding spatial support to temporal databases does not suffice. While spatio-temporal data models describe geometry change, none of the proposed models can model any sort of movement (change in location) and/or mutation (change in shape or size). The conclusion was that the simple aggregation of space and time is inappropriate due to the fact that it fails in capturing and representing continuous change. The temporal characteristics of spatial objects must be investigated to produce inherently spatio-temporal concepts such as unified spatio-temporal data structures, spatio-temporal operators, and spatio-temporal user-interfaces [120].

2.2 Integrated Spatio-temporal data models

Research on integrated spatio-temporal data models concerns *conceptual*, *computational*, *logical*, and *physical* data models. This section looks at the most important efforts that seem to hold promise.

2.2.1 Conceptual data models

Many research in modelling spatio-temporal phenomena were directed towards conceptual data models, in which the following trends can be identified:

- **Data models based on time-dependent functions**

  To the best of our knowledge, the first effort in the area of integrating space and time in a single data model using time-dependent function, was reported by Yeh & Cambray in 1993 and was extended later in 1995 in an approach called *behavioural time sequences* [155, 156]. Each element in this approach, consists of a geometric value, a date, and a behavioural function. The evolution of such a sequence is represented by a behavioural function. Although they worked towards integrating models of space and time, they did not address the embedding of their model in databases [35].

  By the year 1997, the term *moving object* gradually appeared and became one of the major focuses of the spatio-temporal community. Promising results started to come along while using an idea similar to the idea of time-stamped attribute models, in which information about an object is gathered into a single tuple and complex attribute values are allowed. These complex values incorporate the temporal dimension and are frequently modelled as either functions from time into a value domain [35]. Sistla et al. came up with a model, called *Moving Objects Spatio-Temporal* (MOST). MOST and the query language based on it, namely *Future Temporal Logic* (FTL), assume for any spatial object *dynamic spatial attributes*, i.e., attributes that may change value over time, are changing according to a given linear function [123]. A dynamic spatial attribute is represented by three-subattributes; initial position, update time and a function of time, which determines moving object's position between each two consecutive updates using motion vector. For an object moving on a well-defined route, the motion vector is specified by two numbers denoting the upper and lower bound on the speed; for an object moving freely in the two-dimensional space, the motion vector is specified by giving the speed bounds in both X and Y directions [122]. Motion vector can change, but in most cases it does so less frequently than the spatial attribute itself [123]. A rather restrictive assumption they made was that it is possible to automatically update the motion vector when a change in speed or direction is sensed. Moreover, the model did not capture the full trajectory and did not accommodate the whole history of a moving

object, focusing only on present as well as future status of the moving object. Another drawback of this proposal is the fact that the query language was purely temporal. Also, it neither offered a comprehensive set of types and operations nor addressed any other kind of moving geometry but moving point.

A follow-up on the group's proposal was reported by Moreira et al. They proposed a model to represent moving point objects with a decomposition of the trajectory of a moving object into sections. The movement within each section of a trajectory is described by a linear variability function, based on the method of minimum squared deviation. The model allows to provide the complete information about an object's movement whether or not explicit updates of position occurred [93]. A major problem in the model is that locational precision depends greatly on the data acquisition devices used, which might be quite low. One year later, they reported on an extension on their previous data model, proposing a number of basic operators on moving objects with different semantics to provide meaningful operations. They believed that it is necessary to establish relevant operations to be used in queries and appropriate semantics to interpret them to deal with inherent uncertainty of moving objects. Therefore, three semantics were added as prefix or suffix to operations used in queries to cope with uncertainty of moving object data. An important observation about these semantics is that they are integrated at the level of operations rather than being introduced more generically at the query language level. The reason is that they are only meaningful in movement-related operations [92].

Chomicki & Revesz studied a framework in which spatio-temporal objects are described as collections of *atomic geometric objects*. In this framework, the development of each spatio-temporal object is represented by continuous functions [23].

In 2001, Vazirgiannis and Wolfson introduced a model for moving objects on road networks. In this model, the object trajectory is specified by giving the starting address, the starting time, and the ending address. A priori given module computes the shortest cost (distance or travel-time) path in the road network. An innovative aspect of this model is the role that the road network plays in determining the object trajectory. A small set of expressive predicates is defined that could effectively be implemented using existing operators within a DBMS [137].

- **Data models based on abstract data types**

  Since 1987, spatial data types (SDTs) had been widely used in spatial databases. They were formally defined and were utilized as operators in query languages. Even, some prototype systems were implemented to examine how these data types can actually be integrated with database. Despite these efforts, there has not been a completely satisfactory solution available [55]. This was because they were not general, somehow ambiguous, none of the proposed models or implemented prototypes had finite resolution, nor covered the geometry consistently. To overcome these problems, the concept of *realm* was introduced in the data modelling by Güting & Schneider as "a finite, user-defined structure that is used as a basis for one or more system data types" [55]. Later, the *Rose algebra* was defined on top of the realm notion and offered general types to represent spatial primitives, i.e., point, line, and region together with a complete and broad set of operations, which can be used to represent the moving object concept [56].

  Based on the Rose algebra, Erwig et al. in 1998 brought up the interesting idea of encapsulating spatio-temporal data objects as ADT (Abstract Data Types) objects that can be integrated into databases. The idea was to propose an algebra over non-temporal data types that could be lifted and extended for operations over temporal types. They also discussed methods to integrate the defined algebra (data types and lifted operations) into SQL [35]. A type constructor was defined to obtain spatio-temporal objects, whenever it is applied to spatial type objects, i.e. points or polygons. As a result, this model offered better management of spatio-temporal phenomena. The advantages of the ADT approach are manifold; one of such advantages is that ADTs have a better support for modelling continuous changes, thus, are more expressive than time-stamped (attribute, tuple, relation) approaches [34]. Another important advantage of ADTs is due to the fact that they are defined independently of a DBMS data model and query language. Therefore, they can easily be conceptually integrated into other data models and query languages [35]. One of the main principles of Erwig and his colleagues' work was that they tried to support the basic properties of moving points/regions in their data model and query language rather than to use existing applications and adapting them to support moving points/regions. Regardless of the fact that their model was not easy and feasible to implement because of a very large set of data types and opera-

tions proposed, this was a good start and fine step towards building data models supporting moving objects, which in turn could result in developing new applications that were never thought of before.

Another spatio-temporal modelling approach based on ADTs in 1999, came from Parent et al. They believed that previous models had failed to faithfully represent the real world because their conceptual model was improper. Therefore, they proposed a spatio-temporal modelling approach at a conceptual level, called MADS. The proposed data model is one of the few data models to follow the orthogonality principle in space and time integration. MADS supports modelling stationary as well as moving objects, thus both discrete and continuous geometry changes over time [98].

Güting and his colleagues, in 2000 proposed an interesting abstract data model for moving objects, in which spatio-temporal phenomena are represented as attribute data types. Appropriate operations were associated with these data types to provide an abstract spatio-temporal type extension to a DBMS data model and query language. The abstract data model offers some basic data types, which are described by a signature. The most important operator in their data model is a type constructor τ, which maps any given primitive data type α into a function type $\tau(\alpha)$ with semantics $\tau(\alpha) \equiv time \rightarrow \alpha$. So, this allows to define data types such as $\tau(point)$ and $\tau(region)$, respectively [33]. The first represents the position of a point, and the seconds represents the geometry of a region, at any time (t). In this model a moving object is represented by a sequence of $\langle x, y, t, a_1 ... a_n \rangle$, in which x and y refer to object's position at time t and $a_1 ... a_n$ stands for possible attributes. This approach provides complete support for both discrete and continuous spatio-temporal objects. The authors first defined the semantics for data types and operations used in their abstract data model. To implement this model, they later on proposed appropriate data structures and algorithms for the defined data types and operations. This was independently done from the semantics definition. To support time-dependent geometries, a mechanism called *lifting* was used, in which all operations already defined over non-temporal types were uniformly and consistently made applicable to corresponding temporal types. Finally, special operations were offered for temporal types $\tau(\alpha)$ with functional values. The advantage of this model compared to models that use other elements such as polygons and simplicial complices is that the object representation is closer to reality, while in

the others the representation is only an approximation of reality. The main concerns in this model's design were orthogonality in the data types, genericity, simplicity, expressiveness of operations, and closure between structure and operations of non-temporal as well as temporal types. Güting and colleagues' work was the first comprehensive design of spatio-temporal types and operations and their corresponding discrete data model [54].

A continuation of Güting's group work was reported by Forlizzi et al. in which a discrete data model implementing the abstract model was defined. For all data types of the abstract model proposed in [54], corresponding 'discrete' types were defined that can be implemented in terms of finite representations. Constraints were accurately and formally defined to describe a value of the abstract model using finite representation. The interesting part of the model is how temporal types are represented. A novel concept, namely *sliced representation*, is used to represent a temporal evolution of spatio-temporal objects as a set of elements. Each element describes a simple time-dependent function. The authors also addressed how defined discrete representation can faithfully be transformed into data structures used by a DBMS [39].

- **Data models based on unified modelling language**

  Price et al. first in 1999, extended the unified modelling language (UML) to represent the space and time semantics as required for spatio-temporal applications [107]. Their work suffered from lack of formal definition of semantics. One year later, authors in [108] developed a spatio-temporal extension to UML by adding a set of constructs that can be applied to classes, attributes, and associations. The extension is based on spatio-temporal symbols, and a certain section is used to describe the semantics of spatio-temporal properties. One of the unique property of this work is the possibility to group attributes with the same spatio-temporal properties, and consequently, to precisely define characteristics of an object. In general, this work covers a wide range of spatio-temporal semantics with well-defined syntaxes. However, the issue of constraints that can be applied on associations or classes are not discussed [41].

  In 2000, Brodeur et al. developed a geospatial repository compatible with ISO-standards for geographic information. To implement the model, a UML-based visual modelling tool was used [17]. The spatio-temporal characteris-

tics were defined at class and attribute levels. This work suffers from lack of formal definition of constraints and spatio-temporal operators [41].

- **Data models based on modelling unit**

  Tryfona and Jensen in 1999, developed a spatio-temporal extension to the ER model using abstractions and *modelling units* [133]. The modelling units are "generic, autonomous, and semantically meaningful excerpts of conceptual database schemas that capture similar semantic aspects of the application domain" [134]. The authors aimed at defining a set of conceptual structures to be implemented in a DBMS to realistically represent the reality. The proposed model, the *SpatioTemporal ER* model (STER), represents time-stamped (spatial) objects as well as time-stamped tuples by associating temporal and spatial icons to objects, attributes, and relationships. The model offers explicit support for describing and modelling time-varying relation between associated spatial objects [108].

- **Data models based on constraints databases**

  Work in constraint databases [70] is also applicable to spatio-temporal settings. Papers that explicitly address snapshot spatio-temporal examples and models based on constraints include [22, 51, 52]. In this kind of models, a single set of constraints is considered to represent spatial objects. Operations that were originally defined in relational algebra using finite set of points are widely utilized in most of the existing approaches. However, recently, the need to include other operations, e.g., distance, was recognized [51]. Although constraint databases prove to hold conceptual promises, the reported results are still far from satisfactory. Mokhtar et al. who considered the constraint database approach to model moving objects, identified direction, velocity, trajectory, curvature and torsion as important properties of a moving object. Consequently, they argued that these properties can be used as constraints. In their model, a collection of linear constraints over time and a linear function from time to multi-dimensional space represent a time interval and location of a moving point, respectively. Each piece of an object trajectory is modelled as a collection of constraints together with a time interval and a coordinate variables. For instance, $(a_1, a_2, a_3)t+(b_1, b_2, b_3) \wedge t_0 \leq t \leq t_1$ represents a piece of trajectory between $[t_0, t_1]$. To address query processing issue, three types of queries, namely, past, continuing, and future queries were identified. Since traditional constraint databases already properly sup-

port past queries, the emphasis was directed towards new techniques to address continuing and future queries. Considering the fact that distance-based queries, such as the ones previously reported in [126], was the focus of the query language associated with this model, a single *generalized distance* function was defined as a mapping spatial objects to continuous functions from time to space [91]. Later on, the model was extended to cover moving object trajectories with uncertainty. In the new model, a trajectory is modelled as stochastic processes. Uncertainty associated with a trajectory at any point in time is represented using a time-dependent uniform distribution [90].

- **Data models for multidimensional data**

  In an interesting work in 2002, Jensen et al. proposed a multidimensional data model for location-based services, which was an extension to multidimensional model and algebra earlier proposed by Pedersen et al. [99]. One of the main assumptions in this model is that moving objects are active agents, being able to disclose their positional information to location-based services. Using this information, LBSs will provide their agents with appropriate functionalities. A number of requirements for a multidimensional data model for such location-based service was identified, which includes explicit and multiple hierarchies in dimensions, partial containment, non-normalized hierarchies, different level of granularity, many-to-many relationships between facts and dimensions, and handling imprecision of data from different dimensions. Association of the proposed data model with an algebra may provide a basis for formally defining a query language [66].

  In the same year, modelling moving objects over multiple granularities was discussed by Hornsby & Egenhofer. Since various applications require different levels of details, the importance of selecting suitable level of details was addressed. A collection of time-stamped tuples were used to represent movement of an object during a time period. The authors argued that the desired granularity defines which forms of approximation should be used to determine an object movement. Applying a linear approximation on a sequence of time-stamped positions results in determining the simplest form of object trajectory. More detailed and more general views on movement can be captured by refining and coarsening granularity, respectively [59]. This work, however, did not address the issue of moving views between different granularities.

- **Data models for multimedia data**

  In multimedia applications, Nabil et al. proposed a model to retrieve moving objects appearing in multimedia scenes, in which objects were represented in terms of their trajectory, as discrete snapshots [94]. They captured location data representing the centroid coordinate values of a moving object so that direction of the movement of a moving object and the distance between two moving objects could be calculated. In this model, all objects in a study area are represented as a graph, in which spatio-temporal relationships are labelled by edges between object nodes [34]. The model did not address changes in the shape of objects.

  Vazirgiannis et al. in 1996 used topological relationship between static spatial objects as well as temporal relationships between moving objects to extract moving objects from a multimedia scene [136]. They assumed that all objects can be viewed as rectangles and some spatially, temporally, and spatio-temporally relationships may exist between these objects. A set of operators was utilized to represent the temporal relationships between objects. This model does not support any change in location and/or extent.

  Li proposed a method to model multimedia data, in which the trajectory of a moving object was modelled with three parameters: object displacement, the direction of displacement, and the time interval. For any moving object, he considered eight possible directions, namely, north, east, west, north-east, north-west, south, south-east and south-west. The major problems in his model were *(i)* computation of a displacement could be very expensive, and *(ii)* capturing the relationship between objects proved to be difficult. Therefore, he extended his model by definition of *moving spatio-temporal relationship*, in which the relationship between two moving objects was defined by the following three components: the topological relationship between the two objects, the directional relation between two objects and the time interval that these two relations, were holding [81].

2.2.2 Computational data models

Assuming network-restricted movement for moving objects, Speičys et al. in 2003 developed a computational data model. Their proposed data structures can be used as a basis for data storage and query processing as well as to model road networks, moving and stationary objects. The data model includes a two-dimensional

as well as a graph representation of road networks, and static and moving objects [125].

2.2.3 Logical and physical data models

By December 1997, the first attempt to integrate space and time into a relational database was reported by Tryfona & Hadzilacos. The modelling requirements of spatio-temporal applications at the logical level of design were discussed and the essential elements of a spatio-temporal application and the interconnections among them were represented. After identifying key features required for handling spatio-temporal phenomena that are currently lacking in relational data models, they proposed an extension of the relational data model, called the *Spatio-Temporal Relational Model* (STRM), providing a small set of representation constructs [132].

In 1999, Wolfson and his colleagues identified a set of functions that are needed to handle moving objects and consequently they proposed a data model to support such capabilities. In their proposed prototype, called DOMINO, they aimed at integrating these required functionalities in a layer on top of existing DBMSs. They introduced a system architecture that consists of three levels. The first level is an object-relational DBMS, which stores moving object data in a form of a sequence of time-stamped positions. The second level is a GIS that is responsible for storing, querying, and manipulating spatial objects. The third layer, DOMINO, contains temporal predicates and offers support for inherent uncertainty of moving object data [146]. A comprehensive approach to integrate these supports in a commercial DBMS was also proposed [149]. Later, the data model was modified to also support uncertainty of moving object data and deviation between a moving object's actual location and its location as stored in the database. In the new data model, which assumed constrained movements on predefined networks, point objects were either mobile or stationary. If the object is stationary, its location attribute is an (x, y) coordinate pair. However, if the object is mobile, its location attribute has six sub-attributes, namely, the pointer to a line object representing the network segment on which an object is moving, location and time at which the object started its movement, the direction in which the object travelled, the (presumed constant) speed at which the object travelled, and, finally, an uncertainty measure, which could have been either constant or a function of time, representing the threshold of the location deviation. Another contribution of this work was a probabilistic model and an algorithm for query processing. In this model the location of a moving object is a random variable. The density function of this variable

is determined using object location at any point in time and uncertainty derived from the database [147].

Bédard et al. reported on a spatio-temporal visual modelling tool, called Perceptory, which was an extension to UML [109]. It, however, lacks support for raster and multi-dimensional data.

Chen et al. in 2003 developed a prototype called CAMEL (Continuous, Active Monitor Engine for Location-based Services). It was designed to support intelligent location-based services for moving object databases by offering a high performance location management engine [21].

2.3 Spatial change

We have seen that many recent developments in spatio-temporal data models are aimed at supporting the representation of spatial change over time. Two types of spatial change may be distinguished, namely *discrete change* and *continuous change*. Cadastral applications are well-known examples of the former, in which discrete changes are relatively easy to keep track of in a database. This can be achieved by frequently updating the database and recording history [54]. However, with continuous change, it is not feasible to constantly update the database for each such change.

2.3.1 Moving objects

The essence of spatio-temporal data models is to accommodate continuous change over time. Although this phrase almost immediately brings the terms *moving object* and *movement* to mind, it was only recently that researchers came up with formal definitions of both terms in context of spatial data handling.

A *moving object* is an object whose position and/or extent changes over time [34]. The focus of this work is on moving point objects. Therefore, from now on whenever it is referred to a moving object, a moving point object is considered, unless it is mentioned otherwise. Any change in location will be viewed as *movement*. Simply speaking, movement is a mapping of time into space, indicating location at different points in time. The sequence of time-stamped locations visited by a moving object, form that object's *trajectory*. A *trajectory* is an ordered historic trace of locations of a moving object, that can be depicted (simplified) as a line, however, the temporal characteristics are important semantic parameter in definition of trajectory. In fact, a trajectory represents the path taken by an object

together with the time instants at which the object was at every position along the path [137].

In contrast to static objects, moving objects are difficult to represent in a database. Currently, applications dealing with moving objects are being developed in an ad hoc fashion. Despite of all the work on databases, situations in which the whereabouts of objects are constantly monitored and stored for future analysis are an important class of problems that present-day database users will find hard to tackle satisfactorily with their systems. The reason is that special functions needed by such applications are currently lacking. Therefore, there is an essential set of functions that has to be integrated, and built on top of existing DBMSs to support moving objects [146]. Such functions constitute a wide domain ranging from data models, data structure and operations, indexing methods, query processing techniques, and visualization methods that can handle continuous change in moving objects and their large amounts of data.

Applications dealing with time-varying data can be classified in one of the following categories [132], based on the type of change they accommodate:

- Applications that are concerned with changes of *non-spatial characteristics* of objects, e.g., land parcels in a cadastral information system.
- Applications in which the *position* of objects continuously changes, e.g., cars moving in a road network.
- Applications with objects that integrate changes in the above case as well as changes in their *extent*. This case mostly happens in environmental applications, e.g., monitoring water pollution caused by oil spills.

While the first category deals with rather discrete phenomena, the other categories handle continuous change. Since our focus is on continuous change of object positions, here a fundamental issue arises, namely the 'medium' via which the objects are thought to travel. That medium may or may not impose restrictions on movement. This results in at least three scenarios of object movement:

- Free movement in 2D or 3D space, e.g., animal migration.
- Restricted movement in 2D or 3D space, e.g., ships along coastlines.
- Restricted movement on 2D or 3D networks, e.g., car movements.

Obviously, there are scenarios, in which movements occur in combination, for instance, movements in shopping malls. Although these can be seen on the one

hand as free movements, since people can freely move in space in any desired direction and any manner, on the other hand their movements are still restricted to the corridors and spaces between shelves. Another important observation is that situations in which objects are stationary are special cases of movement and should not be ignored. Stationary situations may occur due to physical obstacles (accidents, traffic), permanent constraints on the movement (stops at traffic lights), or personal decisions (waiting for someone in a parked car).

2.3.2 Data acquisition methods

Acquisition of data is one of the concerns in data modelling, since on the one hand it has a direct effect on the data types to be supported by the data model, while on the other hand it requires the data model to provide strategies to handle imprecision and inaccuracy associated with the acquired data.

Data acquisition methods that are used for moving objects can be classified on the following basis:

- **Analog method:**

  In this method, the moving objects themselves take note of their whereabouts (address) at any point in time.

- **Digital methods:**

  - **Passive monitoring:**

    This method requires object identification. Objects are not associated with sensors and are externally monitored. Radar used in air traffic control and photo camera used in car traffic analysis are examples of this method.

  - **Active monitoring:**

    In this method, monitored objects have a positioning device on-board. The positioning device can be used for either of the following purposes:

    * To record a sequence of time-stamped positions. These observations can be either synchronized, if all moving objects are observed at the same instance, or are independent [93]. GPS, Galileo, and cellular phones are examples of this category. Obviously, accuracy of obtained data differs depending on the technology that is used.

* To record speed, direction, and/or acceleration at any point time. This technique is mostly used in function-based data models, in which chip sets associated with objects can detect any change in the mentioned values and consequently record them. Later on, these values are used to find the location of the object.

2.4 Setting the scene

This is a crowded world with mobile inhabitants. Inhabitant mobility gives rise to traffic, which, due to various behavioural characteristics of its agents, is a phenomenon that displays patterns. It is our aim to provide tools to study, analyze, and understand these patterns. We use the moving object phrase as a container term for various organic and inorganic entities that demonstrate mobility that itself is of interest to the application. We target traffic in the widest sense: commuters in urban areas (obviously), a truck fleet at the continental scale, pedestrians in shopping malls, airports or railway stations, shopping carts in a supermarket, pieces of luggage in airport logistics, even migratory animals, under the assumption that one day we will have the techniques to routinely equip many of them with positioning devices. Fundamental in our approach is that we are not only interested in present position, but also in positional history.

Recently, with the coming about of miniature and cheap GPS receivers and cellular phones, data about whereabouts of moving objects can be obtained more easily and more accurately than ever before. This has been one of the reasons to consider active-monitoring technique as the data acquisition method in this work. In particular, it is assumed that people are travelling by some means of transportation equipped with a GPS receiver, enabling data acquisition at regular or irregular time intervals. The acquired data, immediately or with a delay, is transmitted to a central base station.

Generally speaking, GPS data has the following format:

$$\langle header, location, time, altitude, distance, time - interval, speed, direction \rangle,$$

however, since not all positioning techniques are capable of providing momentaneous speed and direction values, we only use part of GPS data, namely, location and time. We derive speed and direction from these two data, whenever it is needed for computational purposes, e.g., Section 4.2.1 and Section 5.2.2. Consequently, $\langle O_{id}, x, y, t \rangle$, represents the position of a moving object, in which O_{id} is

the object identifier, x and y are, respectively, easting and northing location of the object, at time instant t. It should be noted that since location data obtained from GPS represents latitude and longitude using some datums like WGS84, no easy distance function such as Euclidean distance can be directly derived. Therefore this data should be converted to a coordinate system allowing direct distance computations, such as a regular Euclidean grid or Dutch grid (RD).

Here, three important concerns arise. First, *cutting off* a trajectory and *starting a new*. In other words, even if continuous monitoring a moving object for few years is possible, we assume to stop the monitoring process at some point in time and start a new. This results in obtaining various trajectories of a single object. An important question, however, is what computational techniques should be applied to make the cuts. We choose to cut off a trajectory when an object is 'at rest'. Secondly, the frequency in which data is acquired, i.e., the *sample rate*. Small sample rate may result in too few recorded data points, while large sample rate may result in far too many data points. A small sample rate could result in arbitrary wrong results after position interpolation, whereas the latter could lead to storage problems. Although it is quite possible that the application itself adds data points to the data stream, this should be done only on a basis of known physical laws governing movement of specific object. Thirdly, despite its continuous nature, movement can neither be acquired nor represented and stored in a continuous manner. Thus, the knowledge about the movement of an object, as it is stored in a database, can only be a partial representation of reality. Since discrete representations are used for this continuous phenomenon, there is a need to interpolate the data for in-between time instants. We assume that the time interval in which data is acquired is short enough to allow the use of linear interpolation. There are at least four reasons to support our choice of linear interpolation:

1. Linear interpolations are easy to handle mathematically.

2. A linear function has a straightforward representation, which is particulary important for the integration into relations [34].

3. In many of the applications we have in mind, object movement appears to be restricted to an underlying transportation infrastructure that itself has linear characteristics.

4. It is difficult to determine a general function, which can faithfully describe the whole movement.

In using a piece-wise linear interpolation to describe object movement, the following issues should not be ignored:

- Using higher-order polynomials has some advantages, for instance, it could lead to less segments and to a more precise approximation of the movement within each segment [34].
- Result of applying linear interpolation may not be so realistic because movements usually do not have sudden bends. Secondly, using linear interpolation results in stepwise constant speed or velocity and zero acceleration [34].

One should note that an issue that immediately arises as a major concern in applications that people are the moving object of interest, is the privacy issue. Due to this concern, having real data for experiments is not always easy. Therefore, for the experiments in this thesis, simulation technique is used to provide the data whenever having data from real world was impossible.

After discussing the basic assumptions and fundamental issues of this work, now it is time to provide the notations that will be used throughout the following chapters.

2.4.1 Notation

For the sake of clarity, at this early stage it is appropriate to elaborate on terms, concepts, data types and functions that are used later in this thesis. In doing so, we follow the abstract data model proposed by Güting et al. because of the following reasons [54]:

- Defined operations in this data model can be applied to many data types, while they still operate in a consistent way.
- The data model maintains consistency between structures and operations of non-temporal and related temporal types.
- The defined functions in this model are continuous.

The definition and signature of each data type and operation in this thesis are as follows:

- **Data types**

  1. *Natural* and *real* numbers are the two base types; they are denoted as $\mathbb{N}$ and $\mathbb{R}$, respectively. These have the usual interpretations. Furthermore, $\mathbb{T}$ is the set of time instances, considered to be linearly ordered and continuous: $\mathbb{T} \cong \mathbb{R}$.

  2. A point is a suitable representation for an object when only the position, not the extent, is of interest [54]. The *location* l of a point object is of type $\mathbb{L}$, $l : \mathbb{L}$, and it is represented by a pair of (x, y) expressing *easting* and *northing*. We have:

     $$\mathbb{L} \cong \mathbb{R} \times \mathbb{R}.$$

  3. A moving object leaves a trace s of locations visited behind; it is a sequence of locations, the type of which is denoted by $\mathbb{D}$, such that $s : \mathbb{D}$. We have:

     $$\mathbb{D} \cong \mathit{seq}\ \mathbb{L}.$$

  4. If we register with each location in the trace s also the time of visit, we obtain a sequence p of time-stamped locations that is called *trajectory (*or: *path)*, and which has type $\mathbb{P}$, giving $p : \mathbb{P}$. This type is defined as:

     $$\mathbb{P} \cong \mathit{seq}\ (\mathbb{T} \times \mathbb{L}).$$

  5. We view a network g as a directed graph, consisting of nodes and edges. For our purpose, a (directed) edge is represented as a pair of nodes, with the first node the start node. This leads us to a definition of the network type $\mathbb{G}$:

     $$\mathbb{G} \cong \{\ (V, E) : \mathit{set}\ \mathbb{L} \times \mathit{set}(\mathbb{L} \times \mathbb{L}) | E \subseteq V \times V\ \}.$$

     We have $g : \mathbb{G}$.

- **Functions**

  1. $\mathit{len}(p)$: this function takes a trajectory p and returns its number of

locations.

$$p : \mathbb{P} \Rightarrow len(p) : \mathbb{N}.$$

2. $len(s)$: this (overloaded) function, similarly, takes an object trace s and returns its number of locations.

$$s : \mathbb{D} \Rightarrow len(s) : \mathbb{N}.$$

3. $dist(q, r)$: this function takes two locations q and r, and returns the Euclidean distance between them.

$$q, r : \mathbb{L} \Rightarrow dist(q, r) : \mathbb{R}.$$

4. $p +\!\!+ s$: this operator takes two trajectories p and s and returns their concatenation

$$p, s : \mathbb{P} \Rightarrow p +\!\!+ s : \mathbb{P}.$$

This concatenation operator is only defined for two cases:
   - the last timestamp of p falls before the first timestamp of s, in which case the result is that of a simple list concatenation; or,
   - the last timestamp of p equals the first timestamp of s, and their respective locations are identical, in which case the result that of a simple list concatenation of p and s with s's first element removed.

5. $near(q, g)$: is a function that takes a location q and a network g, returning that network segment (edge) of g that is closest to q; if multiple segments fit the qualification, one is arbitrarily chosen.

$$q : \mathbb{L}, g : \mathbb{G} \Rightarrow near(q, g) : \mathbb{L} \times \mathbb{L}.$$

6. $projection(q, w)$: this function takes a location q and a network segment w and returns the location of the projection of q onto (the linear continuation of) w.

$$q : \mathbb{L}, w : \mathbb{L} \times \mathbb{L} \Rightarrow projection(q, w) : \mathbb{L}.$$

7. $ToFrom(w)$: this function takes a network segment w and returns the

set of its end nodes.

$$w : \mathbb{L} \times \mathbb{L}, \Rightarrow \mathit{ToFrom}(w) : \mathit{set}\ \mathbb{L}.$$

8. $\mathit{acptpath}(q, r, g)$: this function takes two nodes q and r lying on a network g, and returns the set of traces between them on the network.

$$q, r : \mathbb{L}, g \in \mathbb{G} \Rightarrow \mathit{acptpath}(q, r, g) : \mathit{set}\ \mathbb{D}.$$

9. $\mathit{dir}(q, r)$: this function takes two locations q and r and returns the direction, expressed as counter-clockwise degrees from the horizontal axis, of the line from q to r.

$$q, r : \mathbb{L} \Rightarrow \mathit{dir}(q, r) : \mathbb{R}.$$

10. $\mathit{speed}(p, s)$: this function takes two time-stamped positions p and s, and returns the average speed of an object between p and s.

$$p, s : \mathbb{T} \times \mathbb{L} \Rightarrow \mathit{speed}(p, s) : \mathbb{R}.$$

11. $\mathit{acc}(p, s, w)$: this function takes three time-stamped positions p, s, and w and returns the acceleration between them

$$p, s, w : \mathbb{T} \times \mathbb{L} \Rightarrow \mathit{acc}(p, s, w) : \mathbb{R}.$$

12. $\mathit{interpolate}(p, s)$: this function takes two time-stamped positions p and s and returns the function of time that is the linear interpolation of the trajectory between them. This function only exists if p's timestamp falls before that of q.

$$p, s : \mathbb{T} \times \mathbb{L} \Rightarrow \mathit{interpolate}(p, s) : \mathbb{T} \rightharpoonup \mathbb{L}.$$

The resulting function's range is $[p_t, q_t]$ (see below).

13. $\mathit{intersec}(p, s)$: this function takes two pairs of time-stamped positions (p, q) and (s, r) and returns the intersection locations of the linear ex-

tensions of the segments derived from each pair.

$$(p, q), (s, r) : (\mathbb{T} \times \mathbb{L}) \times (\mathbb{T} \times \mathbb{L}) \Rightarrow intersec(p, s) : set\ \mathbb{L}.$$

The result of this function is either an empty set (no intersection), a singleton set providing the intersection point, or a doubleton set giving two locations that together determine the intersection line.

- **Other notations**

  1. The i^{th} *data point of trajectory* p, viewed as a time-stamped location
     $p : \mathbb{P}; 1 \leq i \leq len(p) \Rightarrow p[i] : \mathbb{T} \times \mathbb{L}$
  2. The *subsequence of trajectory* p, starting at original index k up to and including index m
     $p : \mathbb{P}; 1 \leq k \leq m \leq len(p) \Rightarrow p[k, m] : \mathbb{P}$
  3. *Projection to time*
     $q : \mathbb{T} \times \mathbb{L} \Rightarrow q_{\mathrm{t}} : \mathbb{T}$
  4. *Projection to space*
     $q : \mathbb{T} \times \mathbb{L} \Rightarrow q_{\mathrm{loc}} : \mathbb{L}$

Chapter 3

Handling uncertainty for moving object data*

> *When one admits that nothing is certain one must, I think, also admit that some things are much more nearly certain than others.*
>
> *Bertrand Russell (1872 – 1970)*

Like databases have in the past, GISes and databases are growing in number and are often created and maintained to help decision-making. Clearly, any decision is worth little if it is based on incorrect or inaccurate data. In other words, the data that is used in any decision-making process needs to be as accurate as possible, or at least as good as is needed for the specific application, if decisions are to be valid and indeed usable. So, it is fair to say that quality of raw data used by an application plays an important role in its success. Any error in raw data might be propagated and affect the result of further analysis. Therefore, an awareness regarding the quality of raw data is essential.

Data quality is defined as the combined characteristics of a data set and its elements and attributes that are important for its proper use [4]. The National Institute of Standards and Technology (NIST) defines *positional accuracy, attribute accuracy, logical consistency, completeness*, and *lineage* as components of data quality, which together represent *forms of accuracy*. According to NIST [4], *accuracy* in general is defined as the closeness of results to true values. Consequently, positional accuracy and attribute accuracy mean closeness of locational information and attribute values to true position and true attribute value, respectively. Due to the importance of time in our work, we identify a separate case for temporal accuracy. Temporal accuracy means how close recorded and true time values are.

* This chapter is partially based on paper published in the book 'Advances in Spatial Data Handling', Proceedings of the joint ISPRS, IGU and CIG Symposium on Geospatial theory, Processing and Applications (SDH), D. Richardson and P. van Oosterom, (Eds.), Springer-Verlag, pp. 391–402, 9–12 July 2002, Ottawa, Canada, ISBN 3-540-43802-5.

Although *precision*, *completeness*, and *accuracy* are often used interchangeably, there is a clear distinction between the three. *Precision* is defined is a measurement of the expected standard deviation. Higher the precision, the lower standard deviation. One should note that having a high precision does not necessarily mean having high accuracy. On the other hand, *completeness* is defined as the degree to which an object and its attributes stored in a system represent the abstract reality. A spatial data set is *logically consistent* if it complies with the structural characteristics of the data model and is compatible with attribute constraints. *Lineage* refers to the recorded history of creation and maintenance of data sources and database.

With data quality arise two important issues, namely, *error* and *uncertainty*. Although these may occasionally be used interchangeably, one should note that in the context of GIS and spatial databases, there is a clear distinction between the two. The former indicates that there is some knowledge about the truth and the differences between that and observations. The latter, however, implies that due to the lack of such knowledge, observations cannot be truly accepted [5].

In a GIS usage, uncertainty may arise from the very first step of *modelling* to the *data acquisition*, *transformation and processing*, or *using the information* [84]. Although modelling, transformation, and data processing may contribute to the uncertainty and lead to wrong outcomes, most of the erroneous results are due to error in the raw data [29]. One should note that errors in raw data can be substantial and that they may be propagated by further processes. On the other hand, uncertainty has a cost, which means that higher uncertainty leads to less correctly retrieved information [149] and consequently results in partially or completely wrong outcomes.

Obviously, error in raw data is accumulated from different sources and research on uncertainty in spatial databases studies effects of imprecision and incompleteness in the observations, modelling, processes and interpretation of digitally represented, geo-referenced phenomena [135]. However, in spite of some similarities, sources of uncertainty are usually application-dependent, because application requirements determine which data acquisition device, modelling, representation, and processing approaches are used. In many of the applications we have in mind, object movement appears to be restricted to an underlying network. Therefore, having both unconstrained and network-constrained moving objects and particularly their locational data uncertainty in mind, the rest of this chapter elaborates on various sources of uncertainty and different proposed methods to handle them.

3.1 Moving object data and uncertainty

There are numerous factors that affect the accuracy and reliability of moving object data, which result in differences between the record of an object's movement and its actual movement. The main reasons for this inherent imprecision are:

- Despite its continuous nature, an object's movement data is acquired in a discrete way.
- Since computer systems do not directly represent continuous phenomena of arbitrary nature, such phenomena must be represented by means of approximation using finite structures.
- The representation of an object's movement is affected by the frequency with which position samples are taken [103].
- Data acquisition devices themselves suffer from various limitations and therefore are associated with errors.
- Factors such as data transmission errors, data transmission delays, or a bad data acquisition environment, e.g., due to adverse weather conditions, may increase the error in raw data.
- Inappropriate sampling methods, data storage, data representation methods, and interpolation techniques affect data uncertainty in processing phase.

In typical databases, it is assumed that the data stored in the database does not change in the reality until an explicit update occurs [149]. However, moving object databases require an extra layer of interpretation, because of discrete data that they use to serve their applications. This extra layer is needed to provide continuous representation of phenomena between two consecutive updates. On the other hand, a proper update policy is required. One should note that regardless of the policy used to update the location of moving objects present in the database, it cannot usually equal the actual location of object. This brings two related but different concepts, i.e., *deviation* and *uncertainty*. The deviation of a moving object's location at a certain point in time is the distance between object's actual location and its database location at that particular time. On the other hand, the uncertainty of a moving object's location at a certain time is the size of the area in which the object may possibly have been [147].

An important question here is when the location of a moving object in the database should be updated. Too few updates lead to ambiguous and possibly wrong outcomes, while too frequent updates may extensively use up storage space and higher data acquisition frequency causes more data traffic, and may therefore be expensive [145]. In 1998, update policies and the imprecision involved in the database representation of moving object data were discussed by Wolfson et al. In a first attempt, an update policy called 'immediate-linear' was proposed [145], in which the overall behaviour of the deviation since the last update was the main factor in deciding when the database should be updated. In this policy, by knowing time, location, and speed of a moving object whenever an update occurs, the database computes the location of the object at any time between two consecutive updates. However, since the object does neither travel at constant speed nor constant direction between two consecutive updates, its actual location may deviate from its database location. This deviation is used to determine when an update should occur. A year later, in follow-up work, the same authors proposed three update policies, namely 'speed dead-reckoning (SDR)', 'adaptive dead-reckoning (ADR)' and 'disconnection detecting dead-reckoning (DTDR)', on the basis of a dead-reckoning policy in which the database is updated whenever the distance between the actual location of the object and its database location exceeds a given threshold [147]. The three policies are different in the sense that the given threshold is either fixed for the total time interval during which the object is monitored, or different for each update being computed using a cost-based approach to minimize the total information cost per time unit until the next update. In contrast to the other policies, in SDR the threshold is fixed for all location updates. For ADR, the threshold is fixed between each pair of consecutive updates, but it may change from pair to pair. For DTDR, the threshold decreases as the time interval between a pair of consecutive updates increases [147]. In 2002, Trajcevski et al. discussed the issue of uncertain trajectories. An uncertain trajectory is obtained by associating an uncertainty threshold with each line segment of the trajectory. For a known path between a given origin and destination, road segments together with the uncertainty threshold for each segment constitute an 'agreement' between the moving object and the central base station. The agreement specifies that the object will send an update of its position to the base station if and only if it deviates from its expected location (according to the known path) by an uncertainty threshold [130, 129].

In 1999, the issue of representation of uncertainty in the position of moving object was analyzed by Pfoser & Jensen. They explained how errors associated with measurement instruments lead to imprecise object data. An object trajectory was determined by applying an interpolation method. Consequently, the fact that inherent imprecision of object data may be increased by used Interpolation technique was discussed. By analyzing these two issues, a technique for estimation of such errors and accounting for that was proposed [103]. Two main assumptions in that work were that presumably using GPS only the spatial dimension but not the time dimension is uncertain and there is no restriction on object movements.

Some efforts were also directed towards accommodating uncertainty in the data model. Moreira et al. believed that it is necessary to establish relevant operations to be used in queries and appropriate semantics to interpret them to deal with inherent uncertainty of moving objects [92]. Therefore, a set of spatio-temporal operations with semantics that embed uncertainty was proposed. Their so-called superset and subset semantics are added as prefix or suffix to operations used in queries to return the set of all candidates that *possibly*, *definitely*, or *probably* match some predicates. Consequently, three semantics, namely 'possibly', 'surely', and 'probably' were defined. To estimate the location of an object at any time instance, they used maximum velocity as a physical constraint on the movement of objects to limit positional uncertainty of the object. Sistla et al. adapted their previous FTL query language to accommodate uncertainty by inclusion of two different semantics for queries, called 'may' and 'must'. The idea behind using such semantics is that due to the uncertainty about the location of objects at any point in time, it may be impossible to answer the queries with absolute certainty; therefore a degree of probability should be used [147]. The answer set of *may* contains all the candidates that, considering the inherent uncertainty associated with object locations, possibly satisfy the query predicate, while the answer set of *must* contains the candidates that definitely satisfy the query predicate [122].

Although uncertainty handling has been studied in different phases of data acquisition, modelling and representation, all researchers agree that error associated with raw data tends be propagated in later stages of data processing and, as a consequence, partial or completely wrong results may be obtained and the whole system may fail to perform well. Therefore, to avoid further malfunctioning of the system, error in raw data should be reduced to the best possible extent. To do so, over the years, different techniques have been used and have been integrated with the data acquisition devices to increase the data accuracy.

Accuracy of collected data strongly depends on the data acquisition technique in use. As previously mentioned we have assumed receivers are used for data collection. Therefore, it is important to closely look at satellite-based positioning techniques for moving objects.

3.1.1 Unconstrained moving objects and uncertainty

Satellite-based positioning techniques form an essential part of spatial information gathering. In these techniques, a number of satellites transmit radio signals. Required time for these signals to be received by receivers on the ground is recorded and, in turn, is used to determine the exact position of the receivers. These techniques can be used for both unconstrained and network-constrained moving objects. Depending on the technique used and presence of constraints, different accuracies are achieved and various methods have been proposed to improve the accuracy. The determination of moving object data by satellite-based techniques has various characteristics:

- Single point (absolute) *vs.* differential (relative) positioning
- Sensor-integrated *vs.* non-integrated positioning
- Static *vs.* kinematic positioning

Single point (absolute) *vs.* differential (relative) positioning

In the single point positioning technique, location of the moving object is determined using a single receiver. To do so, at least four satellite signals are required. The distance between a satellite and the moving object is determined and from four distances four parameters, i.e., latitude, longitude, height, and time, are deduced.

When two (or more) receivers are used, differential techniques can be applied, which work based on correcting systematic errors at the moving object location (one receiver) with measured errors at a known position (second receiver). Therefore, the locational data is determined relative to the location of a fixed receiver or base station (that is known) whose absolute coordinates are known with high precision and accuracy. These techniques, which work on the basis of measuring the location of the object using at least four satellites simultaneously, usually achieve higher accuracy due to reducing or even eliminating systematic errors, such as ionospheric delays, tropospheric delays error, and ephemeris deviations that occur in point-based positioning. Differential techniques involve using the errors

received at the fixed receiver for correcting the computed position of the object. The assumption is that the object witnesses the same errors. This assumption is justified since same systematic errors are received at the object position and the base station. Depending on the application, this can be done in either of the following ways:

- Post-processing (batch).

  This method is a batch process, in which both the object and the base station must simultaneously record data. In this approach, both fixed receivers (base stations) and the moving objects being monitored, collect and record their data from satellites being tracked. This data, later on, is used to mathematically correct and remove the systematic errors. The corrections can be made if and only if both moving object and the base station were receiving signals from the same four satellites [127].

- Real-time processing (on-line).

  Systematic errors can also be corrected in a real-time manner. In this case, the base station plays a less important role in collecting and storing the data, since computation of errors and transfer of corrections must be carried out immediately. Transferring the corrections can be accomplished by radio signal, requiring both the object and base station to have an established radio link. A disadvantage of this is limitation on distance, as most radio signals do not operate well beyond 30 km. One solution to this problem is to use an uplink to a communication satellite. In this case, the object requires a receiver for tracking its current position as well as a communication satellite capability for obtaining the corrections that are transferred from the base station. To provide the same accuracy for a much wider area, satellite augmentation systems are in use. In this approach, a series of base stations operate together, across a wide area providing correction data. Depending on the distance to the nearest base station, position data may be corrected to different levels of accuracy [127].

One of the reasons that GPS signals did not lead to highly accurate positioning for the non-military user segment was due to the effect of selective availability (SA); this was a deliberate signal degradation by the U.S. defence department. With SA turned on the accuracy of the absolute positioning technique was at best 100 m horizontally and 156 m vertically at 95% confidence [2]. With SA turned off the

accuracy of the absolute positioning technique reaches 20 m horizontally and 40 m vertically at 95% confidence [1]. In comparison, existing differential techniques can achieve much higher accuracy, however, they are relatively much more expensive.

Sensor integrated *vs.* non-integrated positioning

Satellite receivers can be thought of as sensors providing discrete time-stamped positions with a defined sampling interval [40]. However, many moving object applications require continuous data streams. In general, satellite-based positioning is not good at providing continuous updates of position in the presence of buildings, trees, clouds, or with passing high vehicles which may hinder the signals and may cause multi-path. In fact, one primary concern is the signal interruption. On the other hand, the connection between an object and a satellite may be lost due to shortage of resources or network traffic. Therefore, the integration of multiple sensors has always been recognized as an essential process to enable continuous data streams [118]. The most popular sensors are inertial sensors, e.g., gyroscope accelerometers, but the list also includes dopplometers, altimeters, speedometers, and odometers [40] to name a few.

The integration of satellite-base positioning with inertial sensors is one of the commonly used techniques, typically accomplished through use of a Kalman filter [48]. Kalman filtering is a iterative technique that aims at determining the parameters of a dynamic system [69], through using both knowledge of the statistical nature of the system's error and knowledge of the system's dynamics. This technique is optimized to produce a minimum error variance [40].

Nevertheless, sensor integration is typically done through associating a satellite receiver with low-cost dead-reckoning (DR) sensors such as magnetic compasses, gyroscopes, and odometers. However, in these techniques error is propagated over time. In addition, they can only operate at times when satellite signals are sufficiently available. An alternative is to use more expensive inertial sensors, which do operate better over longer duration of satellite outages but are usually too costly [118].

Static *vs.* dynamic (kinematic) positioning

Static positioning is a process by which the moving object has to remain at a point, continuously recording its position. The longer time the moving object spends to record its position at that location, the higher accuracy is achieved, since object position is determined by averaging a number of data points. Usually, the moving

object has to stay in a position for a minimum of 15 seconds. In this time interval, it records a number of data points. By averaging these points, the actual position of the object can be determined. Because of number of data points used to compute the actual position, static positioning often offers higher accuracy. However, if not enough satellites are visible or environmental factors, e.g., atmosphere, multipath, are affecting the satellite signal, a longer duration may be required. On the other hand, kinematic positioning is the most productive because a large number of data points can be measured in little time. The idea behind this technique is to collect satellite-based positioning data while moving. Depending on the type of application, or accuracy required, kinematic technique may use a real-time link. It is most commonly found in transportation applications [127].

3.1.2 Network-constrained moving objects and uncertainty

Although the current accuracy provided by satellite-based positioning may be quite acceptable for some applications, e.g., tourist information services, many others require higher accuracy. Car and airplane navigation applications are just two examples. For the former, according to the Institute of Transportation Engineering (ITE), the standard lane width is 12 ft (about 3.6 m) [3] and the best accuracy offered by satellite-based positioning systems at present may result in mapping the object to the wrong side of the road, or even not on the road at all. For the latter, the U.S. Department of Transportation has determined that the desired requirement for aircraft landing are 4.1 m horizontal and 0.6 m vertical accuracy [127], which can hardly be provided at the moment.

In addition to methods explained earlier, for network-constrained moving objects, intelligent approaches can be proposed that aim at improving the accuracy of raw data obtained from the positioning technology and sensors through the integration of measurements with additional spatial information contained in a GIS.

There is no doubt that moving object applications strongly rely on location data. Locational data is not particularly valuable without an underlying database that can match the data to actual locations. Since the data about whereabouts of objects is acquired in a discrete way, the medium via which they move plays an important role in determining their locations. This stems from the fact that discrete data requires interpolation and network data can help out in providing better results. After all the medium may impose restrictions on where objects go. This fact can be used to narrow down the object's location. An essential

assumption that we have made about data is that there is no timing error, while the location data may have error. This assumption seems to be justified considering satellite signal receivers as data acquisition devices in our work and given their accurate clocks.

For network-constrained moving objects, one proposal to reduce the inaccuracies associated with the raw data is to apply a so-called *snapping technique*. The idea behind such a technique is to replace each location obtained from a receiver with its 'most probable' location on the network. These positions together with the original time stamps represent new data points on the network, which we call *snapped data points*. In other words, whenever the registered position is not on the network we want to define the correctness by replacing it with a sufficiently 'close' location on the network. Different criteria, as explained in the following section, can be used for achieving this.

One should note that our snapping technique is not identical to snapping as performed in (GPS) receivers. The latter is opportunistic, only applying on the latest; the current position. Our technique can look ahead and be better informed.

3.2 Snapping criteria

The purpose of applying a snapping technique is to sufficiently correct the registered (raw) data to the best possible extent to avoid error propagation in further processing. We define the following potential criteria as three possible measures to be used in snapping. However, a closer look at each of them reveals that the first two criteria fail to perform well enough.

- Distance from the network only,
- Distance from the network and direction of movement,
- Distance from the network and accessability of network segments visited.

3.2.1 Why a distance-based criterion alone does not work

The idea behind using this criterion in a snapping technique is to snap each registered data point to the 'closest' point on the network. Current GIS and spatial databases have the ability to perform nearest neighbour analysis on points and consequently, to find for each registered data point the 'closest' point on the net-

work. The network segments on which snapped data points are located form the 'most probable' trajectory of the object at hand.

Snapping technique based on this criterion suffers from situations where network segments are densely spaced, as the possibility of having mis-snapped data, i.e., registered data points that are snapped to network segments on which an object was actually not, dramatically increases. This is also true when a registered data point is close to a network node (intersection). Therefore, an additional procedure is required to identify and possibly correct any wrongly snapped data.

On the other hand, using only distance as a criterion in snapping technique requires an additional method to define the 'most probable' trajectory obtained from snapped data points. After all, such a trajectory may not be determined from snapped data points alone as network nodes may also be required. Figure 3.1 illustrates the actual trajectory of an object and result of applying a snapping technique using distance only criterion.

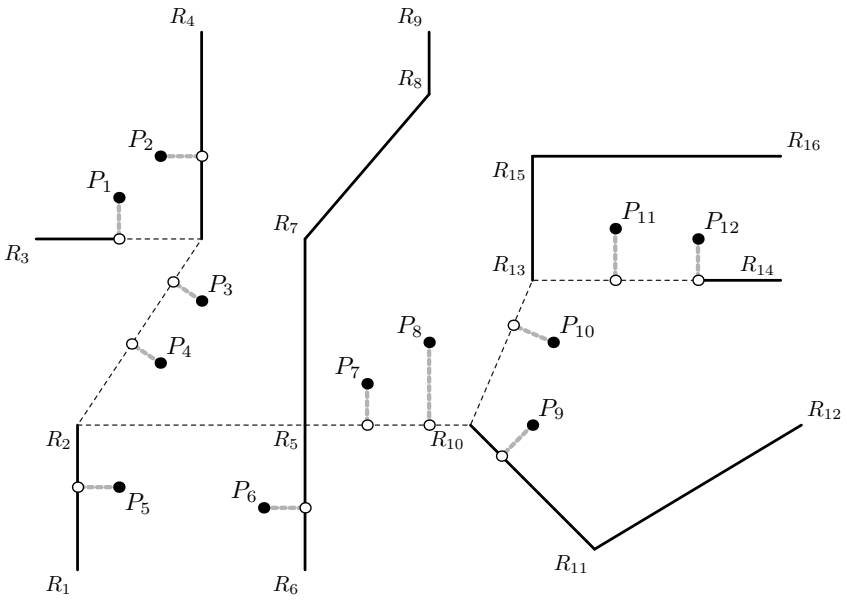


Figure 3.1: *The nearest neighbour analysis determines the nearest point on the network for each registered data point. Distance only criterion may wrongly snap registered data points on network segments, which the object actually did not visit, i.e.,* P_2, P_5, P_6, *and* P_9. *An additional procedure is required to determine object trajectory from snapped data. The actual trajectory of the object is illustrated in gray dashed line. Registered data points in black; snapped data in white.*

3.2.2 Why distance-and-direction-based criteria are not good enough

An improvement to the previous criterion can be made by considering not only the distance from the network but also the direction of an object's travel when snapping the registered data point. This means that for each registered data point, the search for the 'most probable' point on the network is carried out on that closest network segment with heading most similar to that of the moving object. If direction is not directly provided by data acquisition device used, it should first be calculated using two subsequent time-stamped positions.

The largest potential for error in snapping technique based on these criteria occurs when changing direction, as in turning a corner. In such a case, it may happen that neither the segment the object is turning from nor the segment it is turning to match the object orientation [118]. In these situations, distance will be given priority, sacrifying the directional match to overcome these cases.

3.2.3 Distance-and-accessability-based criteria

The two previous criteria suffer from the following drawbacks:

1. Due to positional error, the identified 'closest' network segment (distance-wise as well as direction-wise) may not be the segment on which the object actually was.

2. The closest segment for each registered data point is independently identified without consideration of results of snapping previous or next registered data point.

The latter is the more problematic, since the segments visited by an object should follow a logical order, and therefore, each segment should be *accessible* from its successor.

Distance from the network and accessability of network segments visited are the two criteria that hold promise for correctly snapping moving object data. Our proposed snapping technique based on these criteria is discussed in the following section.

3.3 A snapping technique for moving object data

The key to successfully snap moving object data to an underlying network is to find a closest sequence of *connected* segments between first and last registered data points (preserving their order). To be as general as possible in our proposed snapping technique, we consider both directed and undirected networks.

3.3.1 Snapping to a directed network

To snap moving object data to a directed network, we propose the following steps:

1. Snap each registered data point to its closest network segment

   To do so, a nearest neighbour analysis must be performed. If more than one closest segment is found, one is arbitrary selected.

2. Associate with each registered data point the identifier of the segment to which it was snapped

   This results in a list of network segments, which we call the *assigned segments list*.

3. Check the connectivity of every two consecutive network segments in the assigned segments list

   Two network segments, R_j and R_k (in that order) are connected if one of the following conditions is met. Here $From(R)$ and $To(R)$ represent start and end node of a uni-directional segment R, respectively. For a bi-directional segment R, $ToFrom(R)$ represents the set of both end nodes.

   - $To(R_j) \in ToFrom(R_k)$

     End node of uni-directional segment R_j meets one of end nodes of bi-directional segment R_k.

   - $ToFrom(R_j) \cap ToFrom(R_k) \neq 0$

     One of end nodes of bi-directional segment R_j meets one of the end nodes of bi-directional segment R_k.

   - $From(R_k) \in ToFrom(R_j)$

     One of end nodes of bi-directional segment R_j meets start node of uni-directional segment R_k.

- $To(R_j) = From(R_k)$
  End node of uni-directional segment R_j meets start node of uni- directional segment R_k.

If every two consecutive segments from the assigned segments list are connected, the snapping procedure ends. The assigned segments list will represent the most probable trajectory of the object at hand. The snapped data points, in turn, represent the 'most probable' locations on the network.

Any two of consecutive segments that fail to meet the connectivity conditions are believed to be part of an unacceptable sequence. The unconnected segments should be replaced by those that are maintaining the connectivity of the assigned segments list. How to do so is explained in the following step.

Figure 3.2 illustrates an acceptable and unacceptable sequence of network segments according to the connectivity rule. It is obvious that point P_5 is incorrectly snapped and is violating the connectivity of network segments forming the object trajectory.

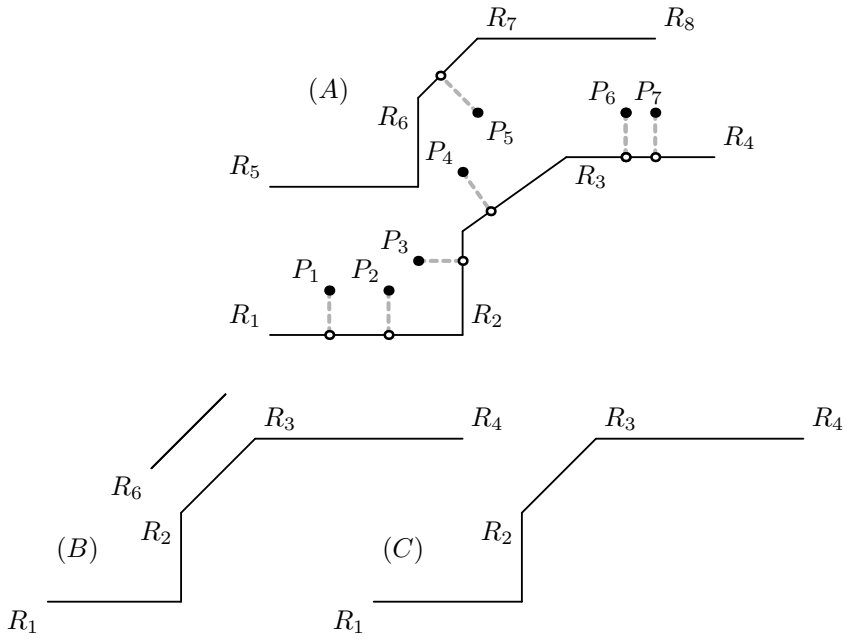


Figure 3.2: *Unacceptable (B) as well as acceptable (C) network segments according to connectivity rule for registered data sequence shown in (A)*

4. Replace unconnected segments

   One should note that not meeting connectivity conditions only identifies two segments that are not connected. Of course, incorrect snapping is the cause of connectivity violation. Either or both segments that do not meet the connectivity conditions may be responsible for this. It is also possible that previous or subsequent segments are responsible. Missing data is a reason for having a unconnected sequence of network segments. As a result, registered data will not be available for all network segments visited. Nevertheless, the connectivity of network segments visited should be kept.

   An important assumption we have made is that the moving object itself is able to make its origin and destination known. This implies that the first and the last segment identifiers in the assigned segments list represent segments that the object actually visited. This assumption does not mean that the object had a priori defined plan. It only means that the object can declare its origin and destination right after its journey was started and right after its journey was ended, respectively. Therefore, if (R_j,R_k) pair does not meet the connectivity rule, R_k is marked as unconnected.

   To replace unconnected segments, the assigned segments list is sequentially starting from the first data point is checked. For each segment marked as unconnected, a search area between the first immediate connected segment before that segment, and the first immediate connected segment after that, is considered. Consequently, we look for all possible connected network segments in the search area. If the search area contains no sequence of connected segments, we need to stretch the search area from both side, moving to the second immediate connected segments before and after the unconnected segment. In contrast, since far too many such sequences may exist, for instance due to cycles, an additional strategy is required to eliminate those that are less desired. Using time stamps of the two registered data points right after and right before the search area and considering a reasonably high speed threshold for the moving object at hand, we can determine the maximal possible distance that the object could have travelled within that period. Consequently, connected sequences that were found, are examined against this maximum possible travelled distance to eliminate those that are longer. Among the rest, the one with the minimal RMSE from the registered data that were snapped in that search area, is chosen. Among two or more connected sequences of network segments with equal RMSE, one can arbitrarily

be selected.

After replacing all unconnected segments, steps 1, 2, and 3 are performed on registered data points that were snapped to unconnected segments. The procedure continues by checking the connectivity conditions for the newly constructed assigned segments list.

The projection of the raw data points onto the 'most probable' network segment leads to identification of error associated with the data. As seems logical (and as our experiments of Section 3.4.2 demonstrate), the error is considered to be homogeneous. This means that error along the segment equals error perpendicular to the segment on average.

One should note that backward movement is a special cases in an object's trajectory, which may result in an unconnected, and consequently unacceptable sequence of network segments. Since the direction of network segments plays an important role in connectivity rules described earlier, this snapping technique fails to correctly snap registered data points in this situation.

An important observation in this snapping technique is that to successfully snap the moving object data to the underlying network, we often sacrify distance for direction, direction for distance, or even both just to keep the logical flow of the data points and connectivity of the visited network segments.

3.3.2 Snapping to an undirected network

Snapping to an undirected network is the same as to a directed network. The only difference is in the connectivity rules. Two undirected network segments, R_j and R_k (in that order) are connected if they fulfill the following condition, in which $ToFrom(R)$ represents the set of end nodes of segment R.

$$ToFrom(R_j) \cap ToFrom(R_k) \neq 0$$

Unlike before, backward movement does not cause any problem in this technique, since the direction of segments visited does not play a role.

The pseudocode for the snapping algorithm to an undirected network is provided below. The pseudocode for the snapping algorithm to a directed network is the same, just the connectivity conditions differ. The notations used in the algorithm have already been described in Section 2.4.1.

```
procedure Snapping(p, g)
p ∈ ℙ
g ∈ 𝔾
for i ← 1 until len(p) do
    assigned_seg[i] ← near(p_loc[i], g)
    snapped_data_loc[i] ← projection(p_loc[i], assigned_seg[i])
    snapped_data_t[i] ← p_t[i]
end for
loop: if len(assigned_seg) < 2 then
        return  (snapped_data, assigned_seg)
else
        check ← [ ]
        connected ← true
        for i ← 2 until len(p) do
            if (ToFrom(assigned_seg[i]) ∩ ToFrom(assigned_seg[i − 1]) = 0) then
              if i = len(p) then
                check[i − 1] ← 1
              else
                check[i] ← 1
              end if
            connected ← false
            end if
        end for
        if connected then
          return  (snapped_data, assigned_seg)
        else
          i ← 2
          flag ← true
          while flag do
                if check[i] = 1 then
                  flag ← false
                  begin_index ← i − 1
                  end_index ← i + 1
                  while check[end_index] = 1 do
                        end_index ← end_index + 1
                  end while
                end if
          i ← i + 1
          end while
          flag ← true
```

```
        while flag do
            q ← ToFrom(assigned_seg[begin_index])
            r ← ToFrom(assigned_seg[end_index])
            cp ← acptpath(q, r, g)
            if len(cp) = 0 then
              if begin_index > 1 then
                begin_index ← begin_index − 1
              end if
              if end_index < len(p) then
                end_index ← end_index + 1
              end if
            else
              flag ← false
            end if
        end while
        d = ∞
        for j ← 1 until len(cp) do
            total_dist ← 0
            for i ← begin_index + 1 until end_index − 1 do
                correct_seg[j, i] ← near(p_loc[i], cp[j])
                correct_snap[j, i] ← projection(p_loc[i], correct_seg[j, i])
                total_dist ← total_dist + dist(p_loc[i], correct_seg[j, i])
            end for
            ave_dist ← √(1/(end_index − begin_index − 2) ∗ total_dist)
            if ave_dist ≤ d then
              index_cp ← j
              d ← ave_dist
            end if
        end for
        for i ← begin_index + 1 until end_index − 1 do
            assigned_seg[i] ← correct_seg[index_cp, i]
            snapped_data_loc[i] ← correct_snap[index_cp, i]
        end for
      end if
end if
go to loop:
```

3.4 Comparisons and results

We tested our snapping technique through simulation. The evaluation procedure started by simulating so-called actual (true) data for both moving object and network. In the next two sections, the data description and analysis of the results are discussed.

3.4.1 Data description

For the experiments, object movement was captured in a sequence of $\langle o_{id}, x, y, t\rangle$ records, in which (x, y) represents the position of the moving object with identification o_{id}, at time instant t. A network consists of nodes with geometry and segments with topology. Triples $\langle r_{id}, \mathit{ToFrom}_1, \mathit{ToFrom}_2\rangle$ are assumed to be the main attributes of a network segment. Nodes ToFrom_1 and ToFrom_2 are the two nodes forming the segment with identification *rid*. To cover the more general case, an undirected network is assumed. Other attributes can be added. The geometry is stored in the nodes. We assume that the network data is known more accurately than the moving object data. Actually, no error is assumed to be associated with the network position.

Ten moving objects were simulated in a way that each moving object was set up to start from a randomly selected network segment and to continue by maintaining the connectivity between the consecutive segments. Normally distributed errors of $\mu = 0$ and $\sigma = 10\ m$ were introduced to locational data to provide the registered data (this data is assumed to be obtained from a data acquisition device). Various statistics of our data set are provided in Table 3.1.

stat	*average*	*standard deviation*
duration	00:28:15	00:12:53
displacement	11.99 km	6.11 km
# of data points	209.6	100.74

Table 3.1: *Statistics on the ten moving object trajectories used in snapping experiments.*

3.4.2 Experimental results

Comparing actual (true) and registered data, the computation of the root mean square error (RMSE) indicated a 12.62 m error on average in the registered data. Although the directions of the error vectors differ, this value gives an indication

about the average size of the error buffers. The procedure continues by applying our proposed snapping technique based on distance from network and accessability of network segments. Comparing the registered and the snapped data, the estimated error results in 8.02 m, measured perpendicularly to the segment directions. Clearly, error in data has been reduced by 36.45% on average.

As previously mentioned, we assumed that the error is homogeneous and therefore the error component in the direction of the segment equals that in the perpendicular direction. Our assumption was justified by results of comparing the RMSE for the snapped data and actual data (which we have only in the simulation). The comparison resulted in 8.03 m error on average. Figure 3.3 illustrates a comparison between RMSE of actual, registered and snapped data points for ten trajectories used in our experiments. From the figure, we can clearly see that error along the network segments (represented by actual *vs.* snapped data) almost coincides with the error perpendicular to the segment (represented by registered *vs.* snapped data). The computed $\mu = 8.2\ m$ and $\sigma = 0.18\ m$ for error along and perpendicular to the network segments is another indication of having homogeneous error.

Results of the mentioned comparisons are summarized in Table 3.2.

Compared data	*Error on average*
Actual & registered	12.62 m
Actual & snapped	8.02 m
Registered & snapped	8.03 m

Table 3.2: *Result of comparison RMSE on average for ten trajectories used in snapping experiments.*

After reducing the uncertainty in the data, different interpolation techniques, such as spline and linear, can be performed on the snapped data to provide the moving object's trajectory. Moving objects can be used to provide valuable information about the environment through which they pass. Obtaining a faithful representation of such object movement opens the door to study the patterns and classes of moving objects, which consequently assist in obtaining more valuable spatio-temporal information about the environment in which that object is travelling. Accessing such information helps to analyze history of moving objects in addition to predicting future behaviour of objects. This brings to our attention the issue of faithful representation of moving object trajectories that is discussed in the next chapter.

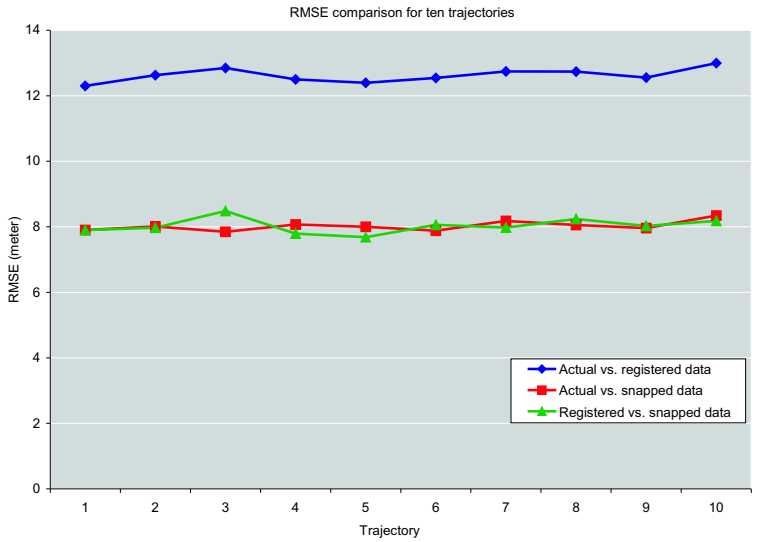


Figure 3.3: *RMSE comparison between actual (true), registered, and snapped data. The fact that error along the network segment almost coincides with the one perpendicular to the network segment is an indication of having homogeneous error.*

3.5 Lessons learnt and plans ahead

In this chapter different satellite-based positioning techniques were reported and the issue of error associated with data obtained from positioning devices was discussed. Since this error may be substantial and it may be propagated by further processing steps of modelling, data processing, and use of the information, it should be reduced as much as possible.

Available techniques for reducing such error for unconstrained moving objects were addressed. Although these techniques can also be used for constrained objects, the obtained results may not be accurate enough for some (mainly network-constrained) applications. One proposal to reduce the inaccuracies associated with raw moving object data is to apply a so-called snapping technique, i.e., a technique to replace each registered locational data with its 'most probable' location on the network. Three snapping criteria, namely, distance from the network, direction of the moving object, and accessability of network segments forming the moving object's trajectory, were discussed as possible measures to be used in a snapping technique. It was shown that distance only, or even distance and direction are not good enough to successfully snap the moving object to underlying network.

Therefore, the snapping technique based on distance from the network and accessability of network segments, which proves to hold conceptual promise, is proposed. Experiments show that the proposed technique effectively reduces the error associated with raw data. The snapping technique can be followed by determining the actual trajectory of the object, in which interpolation techniques are used.

We assumed that error only exists in locational data, while it is recommended to study the timing error as well. Also error in the GIS itself was ignored, while it needs to be taken into account in future work.

Performing the method on real data is useful to overcome problems were not addressed.

Moving objects can be used to provide valuable information not only about themselves but also about the environment through which they pass. To be able to do this, a faithful representation of the object's trajectory is needed.

Chapter 4

Faithful trajectory representation*

What we have to learn to do, we learn by doing.

Aristotle (384 – 322 BC)

The manipulation of discretely acquired moving object data suffers from data unavailability between each two consecutive epochs and consequently, any further computation requires the use of techniques to provide the data when no observation is available and to determine the historical path, or future locations. To do so, techniques such as fuzzy logic [97], Bayesian network [114], and hidden Markov models [106] have been used in robotics and in artificial intelligence. However, in scenarios more similar to ours, the most often used approach is inter-extrapolation techniques. Having a data set of time-stamped positions, finding a function that best represents the whole trajectory using these known values is called *spatial interpolation* [77]. The estimation of positions of the moving object outside this data set is called *spatial extrapolation*. Obviously, estimates outside the data set are less reliable and are prone to higher error rates [127]. There are several different ways to classify spatial interpolation procedures, for instance:

- *Point interpolation vs. areal interpolation*:

  Point interpolation is applied on a given number of spatial data points with known values, and aims to determine the values at other spatial locations. Areal interpolation is applied on polygons and grid structures and has been defined as the transfer of data from one set of areas to another set of overlapping, non-hierarchical, areal units [79].

*This chapter is partially based on paper published in the Proceedings 3^{rd} IEEE Workshop on Web and Wireless Geographical Systems (W2GIS 2003) in conjunction with the 4^{th} International Conference on Web Information Systems Engineering (WISE), G. Santucci, W. Klas, M. Bertolotto, C. Calero, L. Baresi, (Eds.), pp. 18–24, 13 December 2003, Rome, Italy, ISBN 0-7695-2103-7.

- *Global vs. local*:

  Another differentiation among interpolation techniques is the number of data points that is used. Global interpolation determines a single function which is applied across the whole data set allowing a change in a single input value to affect the whole function. It typically provides dominant general trends and ignores local behaviour. Local interpolation, on the contrary, applies an algorithm repeatedly to a small number of immediate neighbours causing a change in an input value to only affect the result within its corresponding neighbourhood.

- *Stochastic vs. deterministic*:

  Stochastic interpolation incorporates the concept of randomness by using probability theory, in which the interpolated result is conceptualized as one of many that might have been observed, all of which could have produced the known data points. In contrast, deterministic interpolation is used when there is sufficient knowledge about the surface being modelled.

- *Gradual vs. abrupt*:

  The gradual interpolation method produces smooth interpolated surface and it is suitable for interpolating data of low local variability. On the other hand, abrupt interpolation results in sudden changes and it is appropriate for interpolating data of high local variability or discontinuous data.

As the method of interpolation is entirely dependent on the characteristics of the moving object, a choice of interpolation technique is important. Since we are dealing with time-stamped positions, local point interpolations can be used. Some of the best known point interpolation techniques are splines, polynomials, Fourier series, and moving average/distance weighted averaging, which in different fields have proved to be more popular than others.

4.1 Problems in trajectory representation

Most of the previous work on trajectory representation used the simple linear interpolation technique [10, 137, 87, 103, 123]. Bézier curves or splines [38] have also often been used to represent the movement of an object. However, none of these methods seems to be entirely suitable enough to describe movement. There

are three important reasons why existing trajectory representation techniques are not accurate and realistic:

1. They work on the basis of defining a single general function to describe the *whole* movement. This means that existing techniques have paid no attention to the fact that a moving object may behave differently at different parts of its trajectory and consequently, a single representation may not adequately describe such differences. The bottom line, as we will shortly show, is that the use of global interpolations to describe a complete movement proves to be difficult and somewhat unrealistic.

2. Existing techniques have often ignored the fact that the trajectory of a moving object may be affected by the movement of other moving objects.

3. In network-constrained movement, the medium via which the object is moving may impose movement restrictions. This has been disregarded in most interpolation techniques. Objects with free require less parameters to be taken into account in determining their trajectories. However, various constraints at different parts of network, as well as the fact that the object can only be on the network and nowhere else, introduce multiple factors that for network-constrained movements should be identified and taken into account to determine the moving object's trajectory as accurately as possible.

Figure 4.1 illustrates the importance of the third argument. In this example, the object is visiting segment R_1 and continues its journey to segment R_2. The use of plain interpolation technique results in a trajectory directly connecting points P_i, P_{i+1}, P_{i+2}, and P_{i+3}. This provides no problem to determine the trajectory between points P_i, P_{i+1}, and P_{i+2}, however, the portion of the determined trajectory between P_{i+2} and P_{i+3} is practically problematic, since the object cannot directly reach point P_{i+3} without passing through the intersection of the two segments. This should be taken into account when using interpolation techniques and the interpolated function should be forced to pass through the intersection node between R_1 and R_2, instead of directly connecting data points P_{i+2} and P_{i+3}, and in so doing ignoring the underlying network. Therefore, in network-constrained movement, the actual trajectory of the object cannot faithfully be determined without taken into account the underlying network.

To identify the problem in using plain interpolation techniques for trajectory determination, analyzing behaviour of these techniques is a must. The major

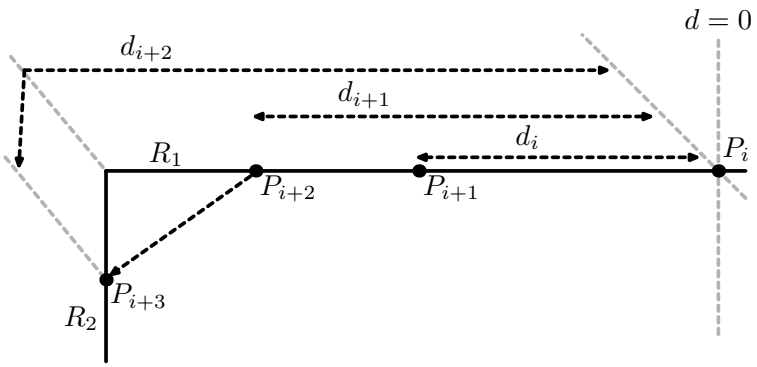


Figure 4.1: *Effects of underlying network in determining the object's trajectory in network-constrained movements. Segments R_1 and R_2 represent two network segments. Travelled distance between nodes P_i and P_{i+1}, nodes P_{i+1}, and P_{i+2}, and nodes P_{i+2} and P_{i+3} are measured along the segments and are represented as d_i, d_{i+1}, and d_{i+2}, respectively. Inclusion of network data forces d_{i+2} to pass through intersection of network segments instead of directly connecting nodes P_{i+2} and P_{i+3}.*

problems introduced by linear and spline interpolation, the two most often used techniques, and solutions to them are explained in the following sections.

4.1.1 Problems introduced by linear interpolation

For a moving object whose trajectory is determined by linear interpolation, a constant speed between each two consecutive data point is assumed. This does not cause a major problem if and only if the time interval between samples are short enough to allow such an assumption. A more severe problem of linear interpolation is caused by abrupt changes in speed, direction, and acceleration that may happen at data points. This is particularly problematic since in reality moving object trajectories do not contain abrupt bends.

To make this more clear, let us consider a simple example, as shown in Figure 4.2, in which profiles of distance D(t), velocity V(t), and acceleration A(t) functions of time (t) for a typical movement are given. In this example an object travels with constant speed, decelerates to stop, waits, accelerates, and travels again with constant speed.

Using linear interpolation to represent such movement results in functions shown in Figure 4.3. As previously mentioned, linear interpolation results in piecewise constant speed and consequently zero acceleration, which are both not the case for the mentioned example.

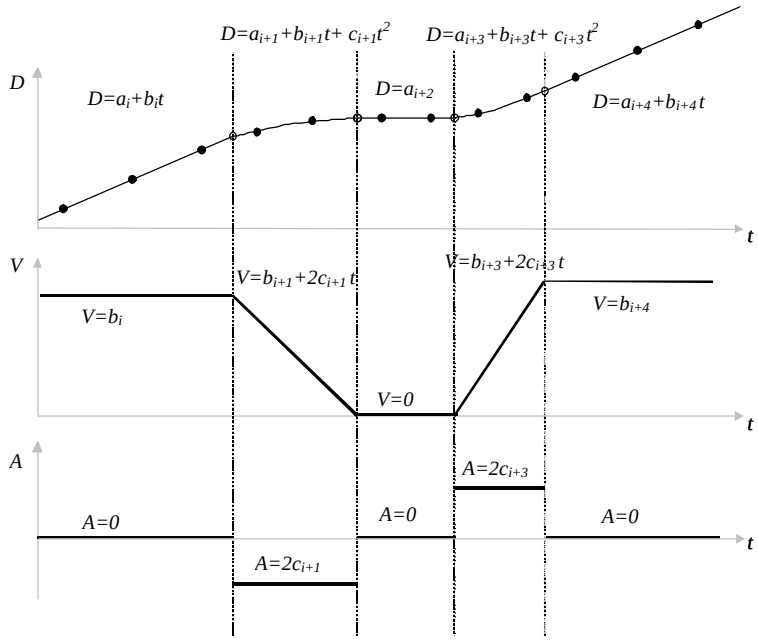


Figure 4.2: *Profiles of distance D(t), velocity V(t) and acceleration A(t) functions representing a single moving object. Raw data on whereabouts of the object is represented in black circles.*

4.1.2 Problems introduced by spline interpolation

Although a spline technique leads to a smooth function, and thus, prevents the abrupt changes of linear interpolation from happening, and though it supports varying speed, it suffers from unrealistic representation of a moving object trajectory especially where the object undergoes behavioural changes. This unrealistic representation can be gleaned from Figure 4.4, in which unusual behaviour of velocity and acceleration functions in the period that the object is likely stationary, i.e., between t' and t'', is indicated.

Spline techniques cannot faithfully represent the behaviour of moving objects during time intervals in which the moving object decelerates, stops, waits, and accelerates, because of their smoothing effects. Using a spline technique, velocity is a quadratic function for all time intervals except the first and last. When the moving object is stationary, the velocity function should be zero. This means that the velocity function has either two or no solutions, leading to the following cases:

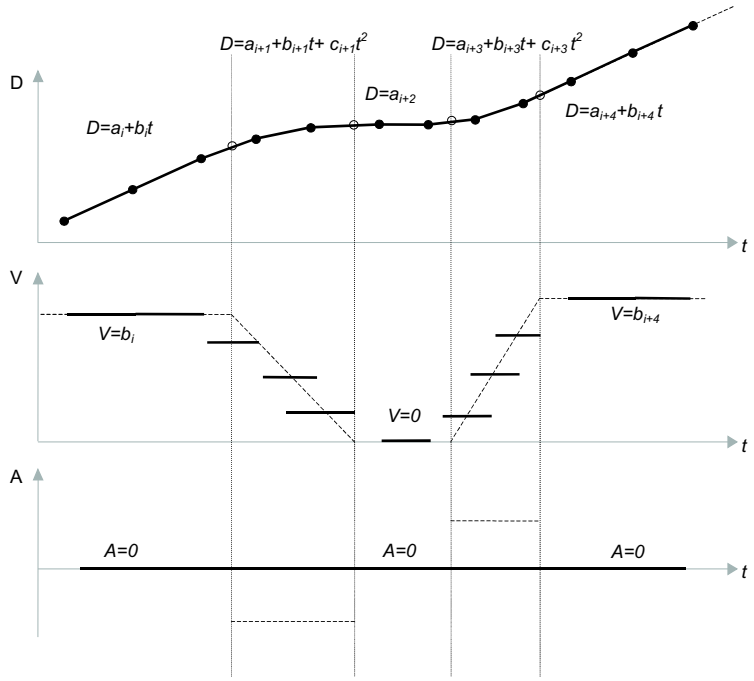


Figure 4.3: *Results of applying linear interpolation on data points shown in Figure 4.2 for the functions D(t), V(t), and A(t).*

- Long stay:

  Figure 4.5 illustrates the velocity function when it has two solutions between two consecutive data points P_i and P_{i+1}. When the object is stationary for a relatively long period of time, spline interpolation will represent the object's movement in a back and forth order to compensate for the waiting period. This means that the object is thought to be changing its direction and is moving back- and forward during the waiting, which in practice is unexpected.

- Short stay:

  On the other hand, as a spline is a smoothing function, to maintain the continuity of the function, it may misrepresent object's movement during the waiting. This means it represents a travelled distance while the object is stationary, as can be seen in Figure 4.6. This case, in which the waiting time is relatively short, happens when the velocity function has no solution.

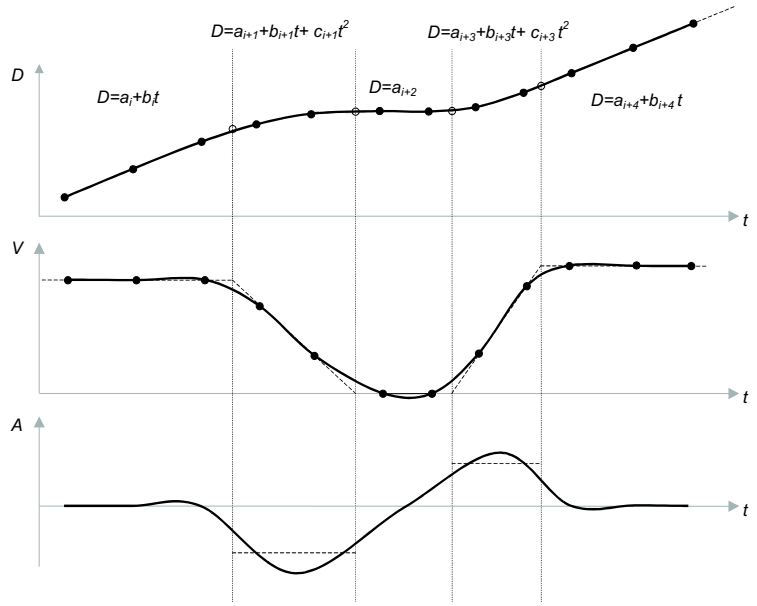


Figure 4.4: *Results of applying spline interpolation on data points shown in Figure 4.2*

Another important issue that can be deduced from Figures 4.3 and 4.4 is that most errors in trajectory representation occur at points at which there is an abrupt change in movement, i.e., in direction, speed, or acceleration. Such abrupt changes may occur due to approaching obstacles, bends, or turns. Regardless of the reason for these abrupt changes, the moving object will not follow the same routine pattern of movement as before, and therefore a smooth interpolation function such as the spline technique or a piece-wise interpolation such as linear interpolation cannot describe the changes that actually occur.

Based on all these facts, using a single function to describe the whole movement should be avoided. This means that faithful representation of a moving object trajectory requires a clear distinction between different patterns of movement (states of the trajectory) and consequently, movement in each of these states should be described by a different function that best suites that portion of the object's movement.

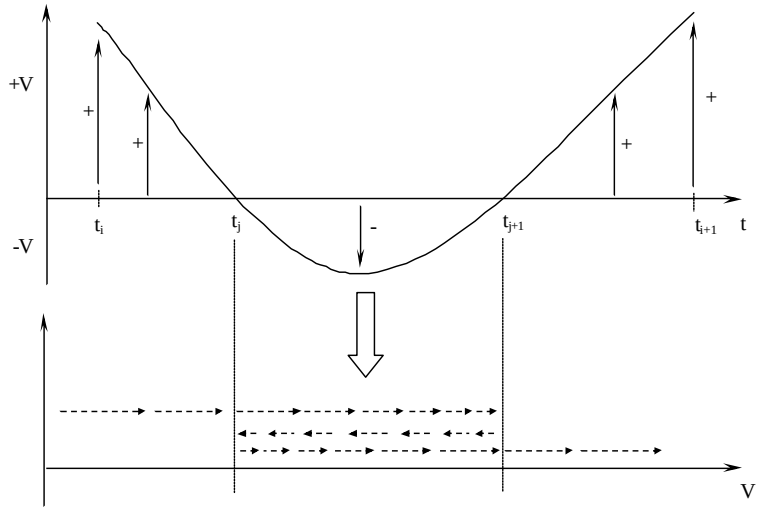


Figure 4.5: *Spline technique forces the moving object to move back and forth to compensate relatively long stationary mode. The object starts to decelerate at t_i and ultimately stops at time t_j. After being stationary during $[t_j\ t_{j+1}]$, it starts to accelerate till time t_{i+1}.*

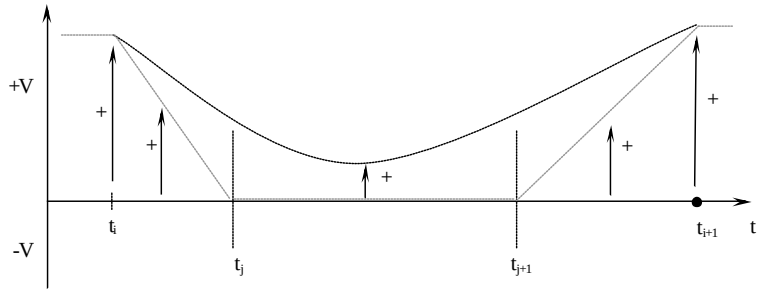


Figure 4.6: *Spline technique may misrepresent the moving object velocity for relatively short stationary mode. As it can be seen from the actual profile of velocity function, shown in gray, the object is stationary during $[t_j\ t_{j+1}]$. However, spline technique maintains its smoothness by representing positive speed and consequently, travelled distance for that period.*

4.2 How to overcome interpolation disadvantages

In using interpolation techniques, the choice of sampling rate is important. In general, the higher the sampling rate, the better. It leads to more data points, and thus a more faithful representation of the object's trajectory. However, this has consequences, e.g., the waste of storage space. To choose an appropriate sampling rate requires prior knowledge about the object's behaviour. This is particularly beneficial when using linear interpolation, since it makes the assumption of constant speed between two consecutive data points more acceptable.

In network-constrained movements, the interpolation function should be defined along the network. This means that all movement restrictions imposed by the network should be taken into account in the best possible way.

To overcome the problems mentioned for the use of linear and spline techniques, the following procedure is proposed.

4.2.1 Break point extraction

To address the previously mentioned problems of spline and linear interpolation and to more realistically represent the object's trajectory, we need to distinguish between different states of a trajectory. This is because an object will follow different patterns in its trajectory. If we identify, when and where their pattern changes, we can define independent single functions that each best represent a single pattern. The changes in pattern occur at positions in the trajectory that we call *break points*. These are points in which abrupt changes in velocity and acceleration occur. These break points are in fact the distinction points between different patterns of movement. Ignoring such points while using interpolation results in a failure to faithfully represent trajectory at these points. Due to their importance, break points require special attention, and we need to determine them. To do so, we propose the following steps:

- *Data point grouping*

  This step is carried out to detect and group consecutive data points that are apparently belonging to a single movement pattern, i.e., constant speed (including stationary mode), constant acceleration, constant deceleration. The overall knowledge about spatial error (uncertainty associated with the data) derived from the snapping technique (and as explained in Section 3.3) is used in this step.

- *Function fitting*

  The aim of this step is to represent different patterns of object's trajectory by fitting a function of time through the data points in each detected group.

- *Intersecting consecutive functions*

  To determine the break points, the spatial intersection between functions representing consecutive patterns are determined. If the intersection points are found at the entrenous of functions ranges, break points may not be as accurate as we want to.

Our method aims at using the identified spatial error in the data when determining trajectory. Therefore, before elaborating more on each of the above mentioned steps, we first explain the error propagation law that is used in our approach.

Error propagation law

We often have mathematical models representing the output as a function of the input. Due to the fact that error associated with the input affects the accuracy of the output and the error can be propagated by the function used, we want to know the uncertainty of the output, as contained in its variance (σ^2). Error propagation law deals with this issue and as Mikhail et al. note can be formulated as follows [88]:

"If $\tilde{a_i}$ is a random variable taking many values a_{ik}, each of which having a random error da_{ik}, then the variance $\sigma^2_{a_i}$ of a_i is given by

$$\sigma^2_{a_i} = \lim_{m\to\infty} \frac{1}{m} \sum_{k=1}^{m} (da_{ik})^2 \tag{4.1}$$

In a similar manner the covariance of two elements $\tilde{a_i}$, $\tilde{a_j}$ of a random vector $\tilde{a}$ may be defined by

$$\sigma_{a_i a_j} = \lim_{m\to\infty} \frac{1}{m} \sum_{k=1}^{m} (da_{ik}\ (da_{jk})^2 \tag{4.2}$$

With these two basic definitions, the propagation of variances and covariances from one set of random variables $\tilde{a}$ to another set $\tilde{b}$ can be formulated.

We let

$$\begin{aligned}
&\tilde{b}_1 = b_1(\tilde{a}_1, \tilde{a}_2, \ldots, \tilde{a}_n)\\
&\vdots \quad \vdots \qquad \vdots\\
&\tilde{b}_q = b_q(\tilde{a}_1, \tilde{a}_2, \ldots, \tilde{a}_n)
\end{aligned}$$

be a set of q functions relating the variables $\tilde{a}$ to $\tilde{b}$. To propagate the errors $da_{1k}, da_{2k}, da_{nk}$ into all $\tilde{b}_i$, let us first define the partial derivatives

$$b'_{ij} = \frac{\partial b_i}{\partial a_j} \qquad i = 1, 2, \ldots, q \quad j = 1, 2, \ldots, n$$

Thus

$$\begin{aligned}
&db_{1k} = b'_{11}\ da_{1k} + b'_{12}\ da_{2k} + \cdots + b'_{1n}\ da_{nk}\\
&\vdots \quad \vdots \qquad \vdots\\
&db_{qk} = b'_{q1}\ da_{1k} + b'_{q2}\ da_{2k} + \cdots + b'_{qn}\ da_{nk}
\end{aligned}$$

or in matrix form

$$db_k = B' \cdot da_k, \tag{4.3}$$

where db_k is a $q * 1$ vector, and da_k is a $n * 1$ vector; B' is a $q * n$ matrix of partial derivatives which is the same as the *Jacobian* J_{ba}. Hence, Equation 4.3 may be written as $db_k = J_{ba} \cdot da_k$.

If we choose the i^{th} and r^{th} elements to represent any two typical elements, then we have $db_{ik} = j_i \cdot da_k$ and $db_{rk} = j_r \cdot da_k$, where $\mathbf{j_i}$, $\mathbf{j_r}$ are the i^{th} and r^{th} rows of J, respectively. Next, we will have

$$db_{ik}\ db_{rk} = j_i \cdot da_k \cdot (da_k)^T \cdot j_r^T,$$

in which $(da_k)^T$ and j_r^T are transpose of (da_k) and j_r vector. Now let

$$M_k = da_k \cdot (da_k)^T = \begin{bmatrix} (da_1)^2 & (da_1\ da_2) & \dots & (da_1\ da_n) \\ \vdots & \vdots & \vdots & \vdots \\ (da_n\ da_1) & (da_n\ da_2) & \dots & (da_n)^2 \end{bmatrix}_k$$

then $db_{ik}\ db_{rk} = j_i \cdot M_k \cdot j_r^T$. We let k run from 1 to m, take the sum, divide by m, and take the limit. Therefore, we will have

$$\lim_{m\to\infty} \frac{1}{m} \sum_{k=1}^{m} (db_{ik})\ (db_{rk}) = j_i \left[\lim_{m\to\infty} \frac{1}{m} \sum_{k=1}^{m} M_k \right] j_r^T \tag{4.4}$$

In views of the definitions of Equations 4.1 and 4.2, we can conclude that Equation 4.4 leads to

$\sigma_{b_i b_r} = j_j \cdot \Sigma_{aa} \cdot j_r^T$,

which represents one element on the i^{th} row and r^{th} column of the total covariance matrix Σ_{bb}. By extending this equation, we will have

$$\Sigma_{bb} = J_{ba} \cdot \Sigma_{aa} \cdot J_{ba}^T, \tag{4.5}$$

which is known as the error propagation law."

Data point grouping

To detect consecutive data points likely belonging to the same movement pattern, speed, direction and acceleration values in consecutive data points must be compared. Since we assume these values are not directly available, these values should first be computed.

Change in two consecutive direction and speed values can be expressed as

$$\Delta\theta = \begin{bmatrix} -1 & 1 \end{bmatrix} \begin{bmatrix} \theta_i \\ \theta_{i+1} \end{bmatrix} = arctan\left(\frac{y_{i+2} - y_{i+1}}{x_{i+2} - x_{i+1}}\right) - arctan\left(\frac{y_{i+1} - y_i}{x_{i+1} - x_i}\right). \tag{4.6}$$

$$\Delta v = \begin{bmatrix} -1 & 1 \end{bmatrix} \begin{bmatrix} v_i \\ v_{i+1} \end{bmatrix}$$

$$= \frac{\sqrt{(y_{i+2} - y_{i+1})^2 + (x_{i+2} - x_{i+1})^2}}{t_{i+2} - t_{i+1}} - \frac{\sqrt{(y_{i+1} - y_i)^2 + (x_{i+1} - x_i)^2}}{t_{i+1} - t_i}. \quad (4.7)$$

If there was no error in the data, $\Delta\theta = 0$ and $\Delta v = 0$ would mean a constant direction and speed, respectively, for the trajectory between the three data points, which are therefore considered members of the same pattern. However, as data may be contaminated by error, we need to tolerate variation in direction and speed difference and introduce a threshold for the determination of patterns. The same applies for acceleration, as we shortly show. To define such a threshold error propagation analysis should be performed.

If $[-1 \;\; 1]$ is denoted as K, in views of the definitions of Equation 4.5, we may rewrite Equations 4.6 and 4.7 as

$$\sigma^2_{\Delta\theta} = K \cdot \Sigma_{\theta_i,\theta_{i+1}} \cdot K^T \quad (4.8)$$

$$\sigma^2_{\Delta v} = K \cdot \Sigma_{v_i,v_{i+1}} \cdot K^T \quad (4.9)$$

Two determine similarity of direction and speed values, their differences should be compared against the value obtained from Equations 4.8 and 4.9, which represent expected uncertainty in the direction and speed, respectively. To define $\Sigma_{\theta_i,\theta_{i+1}}$ and $\Sigma_{v_i,v_{i+1}}$ according to error propagation law we have:

$$\Sigma_{\theta_i,\theta_{i+1}} = \begin{bmatrix} \sigma^2_{\theta_i} & \sigma_{\theta_i\theta_{i+1}} \\ \sigma_{\theta_i\theta_{i+1}} & \sigma^2_{\theta_{i+1}} \end{bmatrix} = J_\theta \cdot \Sigma_{\theta_{x,y}} \cdot J_\theta^T, \quad (4.10)$$

$$\Sigma_{v_i,v_{i+1}} = \begin{bmatrix} \sigma^2_{v_i} & \sigma_{v_iv_{i+1}} \\ \sigma_{v_iv_{i+1}} & \sigma^2_{v_{i+1}} \end{bmatrix} = J_v \cdot \Sigma_{v_{x,y}} \cdot J_v^T, \quad (4.11)$$

where

$$J_\theta = \begin{bmatrix} 0 & \frac{\partial\theta_i}{\partial y_{i+1}} & \frac{\partial\theta_i}{\partial y_i} & 0 & \frac{\partial\theta_i}{\partial x_{i+1}} & \frac{\partial\theta_i}{\partial x_i} \\ \frac{\partial\theta_{i+1}}{\partial y_{i+2}} & \frac{\partial\theta_{i+1}}{\partial y_{i+1}} & 0 & \frac{\partial\theta_{i+1}}{\partial x_{i+2}} & \frac{\partial\theta_{i+1}}{\partial x_{i+1}} & 0 \end{bmatrix} \tag{4.12}$$

$$J_v = \begin{bmatrix} 0 & \frac{\partial v_i}{\partial y_{i+1}} & \frac{\partial v_i}{\partial y_i} & 0 & \frac{\partial v_i}{\partial x_{i+1}} & \frac{\partial v_i}{\partial x_i} & 0 & \frac{\partial v_i}{\partial t_{i+1}} & \frac{\partial v_i}{\partial t_i} \\ \frac{\partial v_{i+1}}{\partial y_{i+2}} & \frac{\partial v_{i+1}}{\partial y_{i+1}} & 0 & \frac{\partial v_{i+1}}{\partial x_{i+2}} & \frac{\partial v_{i+1}}{\partial x_{i+1}} & 0 & \frac{\partial v_{i+1}}{\partial t_{i+2}} & \frac{\partial v_{i+1}}{\partial t_{i+1}} & 0 \end{bmatrix} \tag{4.13}$$

$$\Sigma_{\theta_{x,y}} = \begin{bmatrix} \sigma^2_{y_{i+2}} & 0 & 0 & 0 & 0 & 0 \\ 0 & \sigma^2_{y_{i+1}} & 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma^2_{y_i} & 0 & 0 & 0 \\ 0 & 0 & 0 & \sigma^2_{x_{i+2}} & 0 & 0 \\ 0 & 0 & 0 & 0 & \sigma^2_{x_{i+1}} & 0 \\ 0 & 0 & 0 & 0 & 0 & \sigma^2_{x_i} \end{bmatrix} \tag{4.14}$$

$$\Sigma_{v_{x,y}} = \begin{bmatrix} \sigma^2_{y_{i+2}} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \sigma^2_{y_{i+1}} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma^2_{y_i} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \sigma^2_{x_{i+2}} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sigma^2_{x_{i+1}} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \sigma^2_{x_i} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \sigma^2_{t_{i+2}} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \sigma^2_{t_{i+1}} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \sigma^2_{t_i} \end{bmatrix} \tag{4.15}$$

In these two covariance matrices, no correlation is assumed. Another assumption we have made is that positional error is homogeneous and time variances for different data points are equal, which means $\sigma_l = \sigma_{y_{i+2}} = \sigma_{y_{i+1}} = \sigma_{y_i} = \sigma_{x_{i+2}} = \sigma_{x_{i+1}} = \sigma_{x_i}$ and $\sigma_t = \sigma_{t_i} = \sigma_{t_{i+1}} = \sigma_{t_{i+2}}$. Error σ_l is obtained from snapping technique, as explained in the previous chapter. We have assumed that the timing error equals zero ($\sigma_t = 0$), however, the formula can accommodate for uncertainty

in time data. Therefore, we find

$$\Sigma_{\theta_{x,y}} = \begin{bmatrix} \sigma_l^2 & 0 & 0 & 0 & 0 & 0 \\ 0 & \sigma_l^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma_l^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & \sigma_l^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sigma_l^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & \sigma_l^2 \end{bmatrix} \tag{4.16}$$

$$\Sigma_{v_{x,y}} = \begin{bmatrix} \sigma_l^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \sigma_l^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma_l^2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \sigma_l^2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sigma_l^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \sigma_l^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \sigma_t^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \sigma_t^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \sigma_t^2 \end{bmatrix} \tag{4.17}$$

Next, Equations 4.12 and 4.16 in Equation 4.10, and Equations 4.13 and 4.17 in Equation 4.11 should be substituted. Equations 4.10 and 4.11 in turn are substituted in Equations 4.8 and 4.9, respectively. Consequently, $\sigma_{\Delta\theta}^2$ and $\sigma_{\Delta v}^2$ are defined, which in turn are used to determine an appropriate threshold to group data points based on similar direction and speed values. One should note that $\sigma_{\Delta\theta}^2$ and $\sigma_{\Delta v}^2$ are not necessarily the thresholds and depending on desired *confidence interval*, a ratio of them may be used. A confidence interval is an interval in which a measurement falls corresponding to a given probability [141]. We have assumed that data about the object at hand is normally distributed. For a normal distribution, the probability that a measurement s falls within n standard deviations ($n\sigma$) of the mean μ, having $1-\alpha$ degree of confidence is given by [141]

$$P(\mu - n\sigma < s < \mu - n\sigma) = 1 - \alpha \tag{4.18}$$

Choosing $n = 1$, $n = 2$, and $n = 3$ result in 68.2%, 95.4%, and 99.7%, respec-

tively [72].

A similar procedure should be performed to find a suitable threshold to group data points based on their acceleration values. By comparing two consecutive acceleration values derived from four data points P_i, P_{i+1}, P_{i+2}, and P_{i+3} we have:

$$\Delta a = \begin{bmatrix} -1 & 1 \end{bmatrix} \begin{bmatrix} a_i \\ a_{i+1} \end{bmatrix} = \begin{bmatrix} -1 & 1 \end{bmatrix} \begin{bmatrix} \frac{v_{i+1}-v_i}{t_{i+2}-t_i} \\ \frac{v_{i+2}-v_{i+1}}{t_{i+3}-t_{i+1}} \end{bmatrix}$$

$$\Delta a = \begin{bmatrix} -1 & 1 \end{bmatrix} \begin{bmatrix} \frac{\frac{\sqrt{(y_{i+2}-y_{i+1})^2+(x_{i+2}-x_{i+1})^2}}{t_{i+2}-t_{i+1}} - \frac{\sqrt{(y_{i+1}-y_i)^2+(x_{i+1}-x_i)^2}}{t_{i+1}-t_i}}{t_{i+2}-t_i} \\ \frac{\frac{\sqrt{(y_{i+3}-y_{i+2})^2+(x_{i+3}-x_{i+2})^2}}{t_{i+3}-t_{i+2}} - \frac{\sqrt{(y_{i+2}-y_{i+1})^2+(x_{i+2}-x_{i+1})^2}}{t_{i+2}-t_{i+1}}}{t_{i+3}-t_{i+1}} \end{bmatrix} \tag{4.19}$$

Now, we want to define a reasonable threshold for grouping data points based on similar acceleration. In so doing, if $[-1 \quad 1]$ is denoted as K, in views of the definitions of Equation 4.5, we may rewrite Equation 4.19 as

$$\sigma^2_{\Delta a} = K \cdot \Sigma_{a_i,a_{i+1}} \cdot K^T \tag{4.20}$$

To define $\Sigma_{a_i,a_{i+1}}$ according to error propagation law we have:

$$\Sigma_{a_i,a_{i+1}} = \begin{bmatrix} \sigma^2_{a_i} & \sigma_{a_i a_{i+1}} \\ \sigma_{a_i a_{i+1}} & \sigma^2_{a_{i+1}} \end{bmatrix} = J_a \cdot \Sigma_{a_{x,y}} \cdot J_a^T, \tag{4.21}$$

where J_a can derived from extending Equation 4.13 and $\Sigma_{a_{x,y}}$ is represented in Equation 4.22. Again, no correlation, but homogeneous positional error, and

equal time variances can be assumed.

$$\Sigma_{a_{x,y}} = \begin{bmatrix} \sigma_l^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \sigma_l^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma_l^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \sigma_l^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sigma_l^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \sigma_l^2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \sigma_l^2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \sigma_l^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \sigma_t^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \sigma_t^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \sigma_t^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \sigma_t^2 \end{bmatrix} \quad (4.22)$$

By substituting respective J_a and Equation 4.22 in Equation 4.21, and consequently by substituting Equation 4.21 in Equation 4.20, and considering a desired degree of confidence, a suitable acceleration threshold is defined.

Now, it is time to group data points with similar patterns. First, we group them based on their directions. To detect a sequence of points with constant direction, a procedure of *sequence growing* is used. In such a procedure, the first two data points form the initial group and from there on, each potential new point is compared to its two previous data points. For the three data points combined, $\Delta\theta$ is computed with Equation 4.6, and is compared to the threshold computed from Equation 4.8. Based on such a comparison, either the new point is classified within the group, or is used to start a new group. This results in data point segments, each of which having constant direction. In each of these segments, now, we need to identify data points that have constant speed. A sequence growing procedure is once again used. The first two data points form the initial group and from there on, each potential new point is compared to its two previous data points. For the three data points combined, Δv is computed with Equation 4.7, and is compared to the threshold computed from Equation 4.9. Based on such a comparison, either the new point is classified within the group, or is used to start a new group.

One should note that while a group of points with similar speed (different from zero) can be classified as a group of similar acceleration (resp., zero acceleration),

a group of constant acceleration (different from zero) cannot be classified as having similar speed. Therefore, the groups of constant speed are detected first. Then, for the rest of the unclassified data points we detect groups of similar acceleration.

To detect the sequence of points with similar acceleration, the same sequence growing procedure as for detecting constant direction and speed data points can be used. The only difference is that this time the first three data points form the initial group and from there on, each point is compared to its three previous data points. For these four data points, Δa is computed using Equation 4.19, and is compared to the threshold computed from Equation 4.20.

Function fitting

After classifying all data points and detecting the groups of constant direction and speed and those of constant acceleration, simple distance functions can be used to fit through the data of each group.

To do so, for points belonging to each similar pattern, distance between each data point and the first data point in the group, along the network as a linear function of time in the form of $d(t) = a + bt$ will be defined.

Intersecting consecutive functions

After the parameters of the functions have been determined in the previous step, an intersection of the adjacent functions representing consecutive groups is computed. Data points that did not fit in any of the groups are considered as outliers. As can be seen from Figure 4.7, intersecting consecutive functions results in identifying the break points. These break points are used to define a simple function representing the portion of the trajectory between two consecutive break points. One should notice that the adjacent functions may intersect each other at negative velocities. In these cases, as can clearly be seen from Figure 4.7, an intersection of fitting functions with the time line, i.e. with $v(t) = 0$, should be computed.

The pseudocode for the break point extraction algorithm is provided below. The notations used in the algorithm have already been described in Section 2.4.1.

procedure *Breakpoint_extraction*$(p, dir\_threshold, speed\_threshold, acc\_threshold)$
$p \in \mathbb{P}$
$dir\_threshold, speed\_threshold, acc\_threshold \in \mathbb{R}$
if $len(p) < 2$ **then**
  return p
else

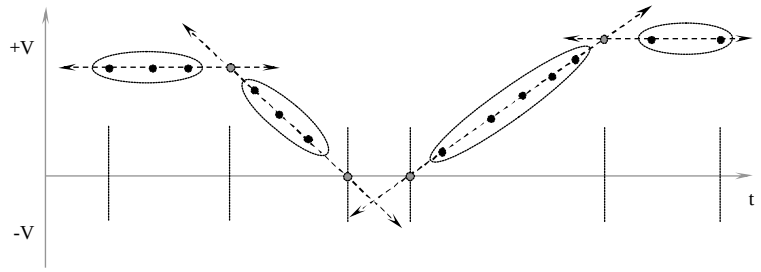


Figure 4.7: *Identifying break points by finding the intersection of adjacent functions through groups of data points. Black circles, gray circles and dashed arrows represent original data, break points and the fitting functions, respectively.*

$j \leftarrow 2$
$k[1] \leftarrow 1$
for $i \leftarrow 1$ **until** $len(p) - 2$
  if $|dir(p_{loc}[i], p_{loc}[i+1]) - dir(p_{loc}[i+1], p_{loc}[i+2])| > dir\_threshold$ **then**
    $k[j] \leftarrow i + 2$
    $i \leftarrow i + 1$
    $j \leftarrow j + 1$
  end if
end for
$k[j] \leftarrow len(p)$
$check \leftarrow [\ ]$
$check[1] \leftarrow 1$
for $j \leftarrow 1$ **until** $len(k) - 1$
  for $i \leftarrow k[j]$ **until** $k[j+1] - 3$
    if $|speed(p[i], p[i+1]) - speed(p[i+1], p[i+2])| > speed\_threshold$ **then**
      $check[i+1] \leftarrow 2$
      $check[i+2] \leftarrow 2$
    end if
  end for
  $check[k[j+1]] \leftarrow 1$
end for
for $i \leftarrow 1$ **until** $len(check)$
  if $check[i] = 2$ **then**
    $j \leftarrow i - 1$
    while check[i+1] = 2 **do**
      $i \leftarrow i + 1$
    end while

```
        m ← i − j
        flag ← true
        while flag ∧ m ≥ 3 do
            δa ← |acc(p[j], p[j + 1], p[j + 2]) − acc(p[j + 1], p[j + 2], p[j + 3])|
            if δa > acc_threshold then
              flag ← false
              i ← j + 3
            else
              j ← j + 1
            end if
            check[j + 2] ← 2
            check[j + 1] ← 0
            m ← m − 1
        end while
    end if
end for
k ← 1
j ← 1
m ← 1
break_point[1] ← p[1]
for i ← 1 until len(check) − 1
    if check[i + 1] = 0 then
      i ← i + 1
    else
      if check[i + 1] = 1 then
        d[k] ← interpolate(p[j], p[i])
      else
        d[k] ← interpolate(p[j], p[i + 1])
      end if
      k ← k + 1
      if j > 1 then
        if intersec(d[k], d[k − 1]) = 0 then
          break_point[m] ← p[j]
          break_point[m + 1] ← p[i]
          m ← m + 2
        else
          break_point[m] ← intersec(d[k], d[k − 1])
          m ← m + 1
        end if
      end if
```

$j \leftarrow i + 1$
end if
end for
end if

4.3 Experimental results

The proposed break point extraction method was tested using a simulation technique. The evaluation procedure starts by simulating so-called true data for both the moving object and the network. The moving objects were randomly simulated in the network. Normally distributed errors of $\mu = 0$ and $\sigma = 10$ m were introduced to the locational data to provide the registered data. Other statistics of our data set are provided in Table 3.1.

An important observation is that existing interpolation techniques do not account for errors or uncertainty measures. On the other hand, our proposed method requires error measures of the locations along the network. From Section 3.4.2, we have shown that the error is homogeneous. Therefore, the error along network segment is assumed to equal the error perpendicular to the segment. This is the error previously computed in Section 3.3. After detecting the data sequences with similar characteristics, a least squares polynomial fitting was implemented. Intersecting the functions representing the consecutive groups resulted in break point detection.

Corresponding root mean square errors (RMSE) were computed and were found to be 7.4 m, 7.8 m, and 1.9 m, for spline interpolation, linear interpolation, and the break point extraction method, respectively. Positional error resulted from applying spline interpolation, linear interpolation and the break point extraction method on three moving objects are shown in Figures 4.8, 4.9, and 4.10.

Comparing the root mean square error values reveals that spline technique and linear interpolation methods are highly affected by the noise contamination in the data. As Spline interpolation already performs better than linear interpolation, to enable a better comparison, much more zoomed views of interpolation error in spline interpolation and break point extraction method for one of the trajectories are shown in Figure 4.11.

Another advantage of break point extraction method is that, unlike the spline method, it deals with each data point individually. Therefore, the time complexity involved in computations is O(N), where N is the number of data points. Another

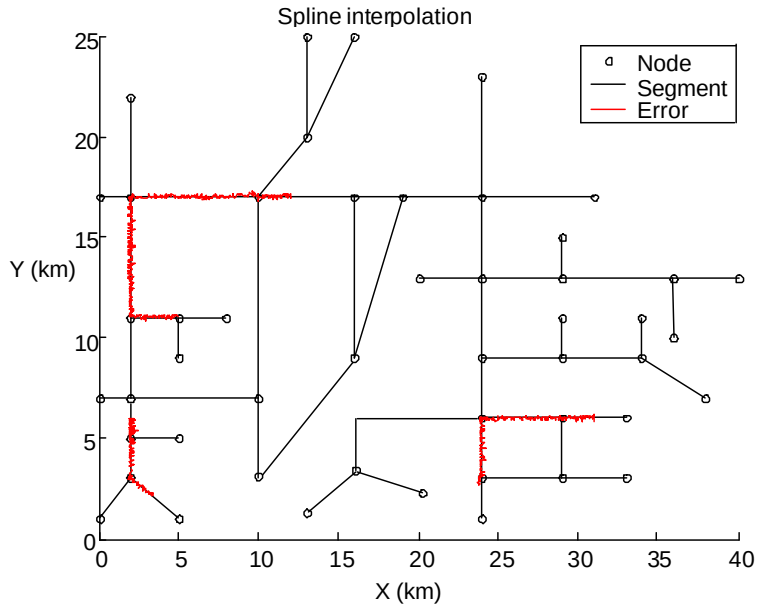


Figure 4.8: *Positional error obtained from applying spline interpolation.*

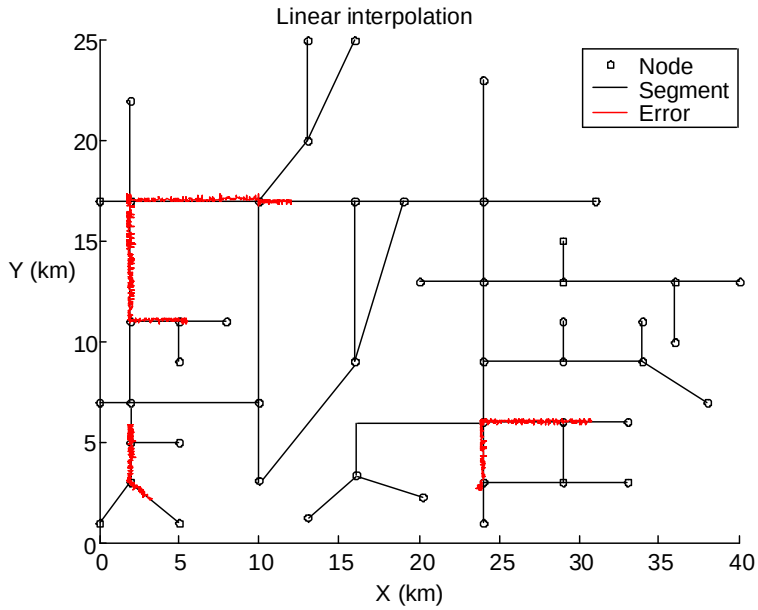


Figure 4.9: *Positional error obtained from applying linear interpolation.*

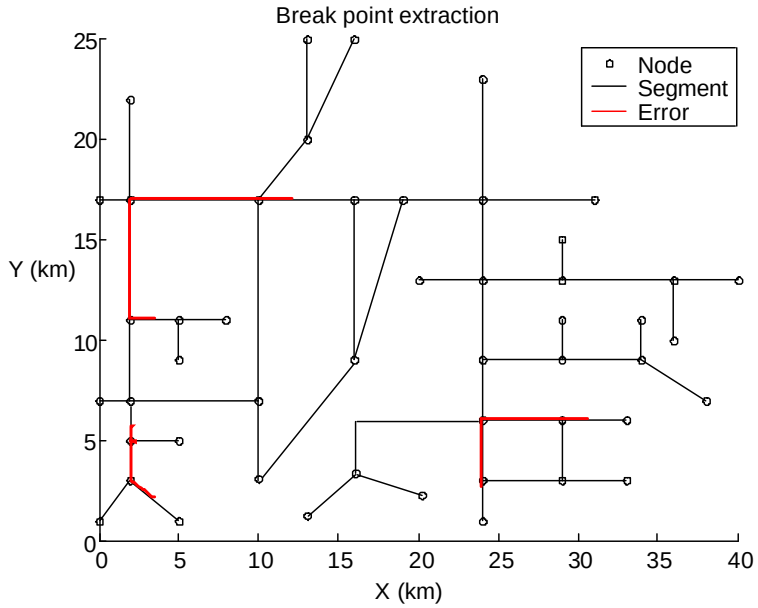


Figure 4.10: *Positional error obtained from applying break point extraction technique.*

important advantage of our method is its contribution to data compression, as will be shown in Section 5.4.1.

4.4 Lessons learnt and plans ahead

The use of discretely acquired moving object data requires techniques of data provision when no observation is available. To do so, interpolation techniques and in particularly spline and linear interpolations are widely used. However, they fail to faithfully represent the trajectory of a moving object in many cases. In this chapter, the important problems of each of these techniques were discussed.

The concept of 'break points' was introduced as points along the moving object's trajectory in which an abrupt change in movement seems to occurs. A method for extracting break points based on data sequence grouping was presented. Data points that proved to be similar in direction, speed, or acceleration are added to an a priorily defined sequence. The similarity depends on the uncertainty in the data, which was defined using an error propagation law. The procedure also makes use of least squares functional fitting and intersections. Ex-

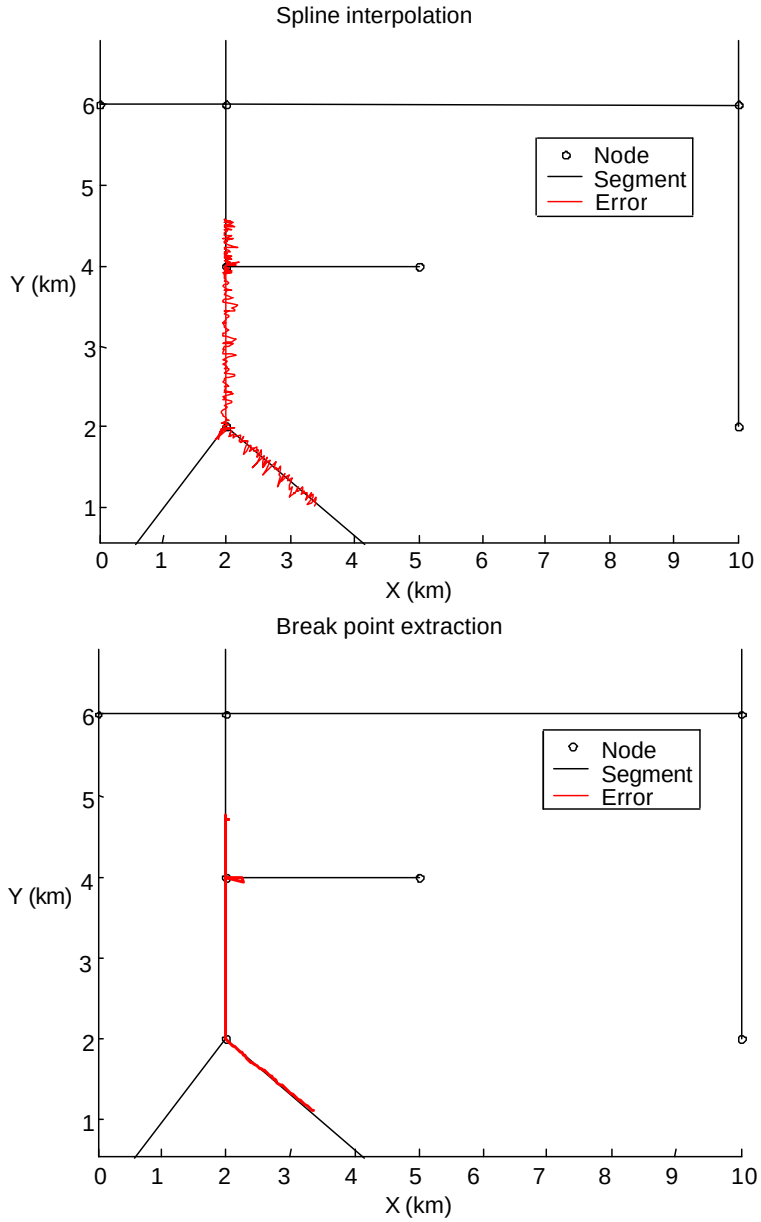


Figure 4.11: *More zoomed views: (Up) Interpolation error in spline technique, (Bottom) Interpolation error in break point extraction method.*

periments showed that by applying the break point extraction techniques, a fair representation of a moving object trajectory can be achieved. On the other hand, this technique holds conceptual promises for compressing moving object trajectories.

For network-constrained moving objects, the obtained information about individual objects may provide overall information about the network itself, which enables more efficient query analysis. This issue together with data compression are to be discussed in future work.

More error analysis is recommended including the estimation of the confidence intervals. Issues such as defining the maximum effective time interval (in other words, the minimum data for economical reasons) to faithfully represent the object within tolerance values of space and time, and defining a cost model (in terms of time) for further spatio-temporal analysis are areas to be addressed in future work as well.

Chapter 5

Trajectory compression techniques*

Not everything that can be counted counts,
and not everything that counts can be counted.

Albert Einstein (1879 – 1955)

Monitoring and analyzing moving objects necessitate the availability of complete geographical traces to determine locations that objects have had, have, or will have. Due to the intrinsic limitations of data acquisition and storage devices such inherently continuous phenomena are acquired and stored (thus, represented) in a discrete way. Right from the start, we are dealing with approximations of object trajectories. Intuitively, the more data about the whereabouts of a moving object is available, the more accurate its true trajectory can be determined. Availability of data is by no means a problem as the coming years will witness an explosion of such positional data for all sorts of moving objects. Moreover, there seem to be few technological barriers to high position sampling rates. However, such enormous volumes of data lead to storage, transmission, computation, and display challenges. Hence, there is a definite need for *compression techniques* for moving object trajectories.

Perhaps an example could help to better illustrate the essence of an effective compression technique. Let us assume data for a moving object as a sequence of $\langle t, x, y \rangle$, in which x and y represent coordinates of the moving object at time t. If such data is collected every 10 seconds, a simple calculation shows that 100 Mb of

* This chapter is partially based on papers published in (1) The Proceedings 9^{th} International Conference on Extending Database Technology (EDBT 2004), E. Bertino et al. (Eds.), Lecture Note of Computer Science Vol. 2992, Springer-Verlag, 14–18 March 2004, Heraklion, Crete, Greece, pp. 765–782, ISBN 3-540-21200-0 (2) The Proceedings of ISPRS Commission II and IV, Working Groups II/5, II/6, IV/1 and IV/2 Joint Workshop on Spatial, Temporal and Multi-dimensional Data Modelling and Analysis, 2–3 October 2003, Quebec city, Canada, 8 Pages

storage capacity is required to store the data for just over 480 objects for a single day, barring any data compression. We obviously want to monitor many more moving objects, and for much longer time periods.

In essence, *compression techniques* aim at substantial reductions in the amount of data without serious information loss. Lossless compressions have no information loss and are often based on optimizing the information capacity per byte used. The main shortcoming of lossless compression techniques is their limited amount of compression. Lossy compressions do suffer from information loss, and are often based on discarding the least informative data parts.

Previously some work has been done in data compression for GIS. Spatial data in GIS are stored either in vector or in raster format. Since we are dealing with vector data, i.e., time-stamped positions, it is appropriate to look into existing data compression techniques for vector data in GIS. The basic units of vector data are points, lines, and polygons that are composed of one or more pairs of coordinates. Therefore, compression can be achieved by removing unnecessary details from them, which in turn results in generalizing the features. Since this type of compression permanently removes some parts of data, it will be a lossy compression [95]. Wavelet and Fourier transformations have been used to compress linear features [60, 105]. While using these transformations, a set of coefficients are used to represent linear spatial objects. The low frequency coefficients stand for important characteristics of the object and obviously should be retained. On the other hand, the high frequency coefficients represent the unimportant details and may be removed [105]. Both of these approaches do not provide high compression since their simplification is limited to a certain amount [95]. On the other hand, research has shown that they suffer from serious problems. For instance, Plazanet et al. reported that Fourier transformation lacks spatial information, therefore, it may mis-represent topological relationships. In addition, it does not produce smooth results, which, in turn, leads to deformation of line curvatures and possibly shortening the bends by enlargement [105]. Compared to Fourier transformation, Wavelet transformation often produces much better results. However, it still suffers from location drifts and the shortening of the bends [95].

To compress vector data, an alternative to transformation approach is simplification of linear features through selective elimination of unnecessary vertices. This approach results in keeping the most critical points in the original representation of features and only removing the non-critical points [95]. Line simplification methods are the most often applied compression technique and are widely used

in commercial GIS software packages [138]. However, in our case, time-stamped positions data do not form a line even if they are simplified and represented as line, as they are historically traced points. On the other hand, although data compression techniques have widely been studied and used in areas dealing with time series, e.g., data mining, they mainly deal with one-dimensional time series and are good for short time series and in absence of noise, three characteristics not met by moving objects.

By targeting applications in which present and past positions of objects are important, our main focus is on compression of moving object trajectories. The central theme of this chapter concerns a lossy compression technique for moving object data streams that attempts to preserve the major characteristics of the original trajectory. We, first, apply some older techniques of line generalization, and compare their performance against algorithms that we have specifically designed for compressing lengthy streams of time-stamped positions data, i.e., moving object trajectories.

5.1 Spatial compression techniques

Object movement is continuous in nature. Acquisition, storage, and processing technology, however, force us to use discrete representations. Therefore, intuitively, the more data about such a continuous phenomenon is available, the better movement can be understood. However, we cannot ignore storage, representation, computation and transmission challenges for huge amounts of data.

Our objectives for data compression are:

- to obtain a lasting reduction in data size; i.e., we will not attempt to decompress at any point in the future,
- to obtain a data series that still allows various computations at acceptable (low) complexity; i.e., the data compression should itself not lead to computational overhead once it has taken place,
- to obtain a data series with known, small margins of error, which are preferably parametrically adjustable.

The reasons for the last objective are that *(i)* we know our raw data to already contain error, *(ii)* depending on the application, we could permit additional error, as long as we understand the behaviour of error under settings of the algorithmic parameters. Setting the parameters of a compression algorithm always

means making a trade-off between the compression rate achieved and the errors allowed. Trade-off characteristics of different algorithms, as we shall see, differ tremendously. To achieve these objectives, our interest is with lossy compression techniques.

Compression algorithms can be classified not only based on wether they permit information loss but also on other base as well. For instance, they are either *batch* or *on-line* algorithms, based on whether they require the availability of the full data series. Batch algorithms do so; on-line algorithms do not, and are typically used to compress data streams in real-time. Batch algorithms consistently result in higher quality outcomes [75] when compared to on-line algorithms, as is to be expected. It appears, however, that most moving object applications are inherently dynamic and thus on-line algorithms seem the more appropriate to use.

Most compression algorithms for data series of the type that we consider here can be classified into one of the following four classes of computational strategy:

Top-Down : Starting from the whole data series, it is recursively segmented until some halting condition is met [75].

Bottom-up : Starting from the best representation of the data series, successive data points are joint together until some halting condition is met. The algorithm may not visit all data points in sequence.

Sliding Window : Starting from one end of the data series, a window of fixed size is moved over the data points, and compression takes place only on the data points inside the window.

Opening Window : Starting from one end of the data series, a data segment, i.e., a subseries of the data series, is grown until some halting condition is met. Then, a compression takes place on the data points inside the window. This will decrease the window size, after which the process is continued. The 'window' size is the number of data points under consideration at any one point during the execution of the algorithm.

We have coined the term 'opening window' to do justice to the dynamic nature of the size of the window in such algorithms.

One should note that to describe the behaviour of data series between each consecutive data points, interpolation techniques are used. Since we use linear interpolation, such behaviour is described by the segment connecting each two consecutive data points.

Various halting condition can be applied but algorithms do not always support these conditions in combination. Possible conditions are [75]:

- The number of data points, thus the number of segments, exceeds a user-defined value.
- The maximum error (*max_error*) for a segment exceeds a user-defined threshold.
- The sum of the errors of all segments (*total_error*) exceeds a user-defined threshold.

An obvious class of candidate algorithms for our problem are the line generalization algorithms. Some of these compression algorithms are very simple and do not take into account any relationship between neighboring data points. They may eliminate all data points except some specific ones, e.g., leaving in every i^{th} data point, where i is a fixed integer based upon the desired degree of compression [128]. Another sort of compression algorithm exploits the characteristics of the neighboring data points to determine whether to eliminate one of them. Particulary, these algorithms may use the Euclidean distance between two neighbour points. If it is less than a predefined threshold, one is eliminated. All these algorithms are sequential in nature, that is they gradually process a line from one end to the other.

Although these two groups of algorithms are computationally efficient, they are not so popular or widely used. Their primary disadvantage is due to the fact that *(i)* some critical data points may be eliminated or misrepresented, and *(ii)* straight lines may be over-represented [31], unless small differences in angle are used as another discarding condition.

An algorithm to overcome the first limitation was first reported by Lang [78]. In this method, the perpendicular distance from a line connecting the first data point (the anchor) and the third data point (the float) in the series to an intermediate data point is evaluated against a pre-defined user threshold. As long as all distances of intermediate data points, i.e., those between anchor and float, are below a distance threshold, an attempt is made to move the float one point up in the data series. When the threshold threatens to become exceeded, two strategies can be applied: either, the intermediate data point with a threshold violation or the data point just before it becomes the end point of the current segment, and it also becomes the anchor of the next segment. If no threshold excess would take place, the float is moved one up the data series, and the method continues,

until the entire series has been transformed into a piecewise linear approximation. Jenks basically applied the same idea on every two consecutive points [62]. To tackle the second disadvantage, he utilized the angular change between each three consecutive data points [63]. In this method, the angle between the vector connecting the first and the third point is calculated. If this angle is greater than a pre-defined angular threshold, the intermediated point is retained. Otherwise, the intermediate point is ignored. This algorithm is likely to discard critical points along the line with small curvature [82].

Regarding halting condition in their approach, Shahriari et al. proposed a method to avoid applying the same threshold for a whole data series [95]. In this approach, a user-defined threshold is first defined, and then the proper threshold, i.e., a value that results in obtaining maximum simplification, while retaining the original user-defined threshold, is automatically computed for each line in the data series.

The most important compression algorithms that seem to hold conceptual promise are reviewed in Sections 5.1.1 and 5.1.2.

5.1.1 Top-down compression algorithms

The idea behind the top-down algorithms is to split the data series at the best possible position amongst them, i.e., where the committed error is above some user-defined threshold. The algorithm then continues by recursively segmenting the resulting subseries until committed error for all segments is below the threshold [75].

An often used and quite famous top-down method is the *Douglas-Peucker* (DP) algorithm [31]. It was originally proposed for line simplification, and aims at preserving directional trends in the approximation line using a pre-defined distance threshold. McMaster who gives a detailed study of mathematical similarity measures, ranked the DP algorithm as 'mathematically superior' [85] and called the DP algorithm one of the most geometrically efficient algorithms in processing sequences of coordinate pairs [86]. White carried out a study on simplification algorithms on critical points of psychological data series and showed that the DP algorithm was best at choosing splitting point; she referred to the obtained results as 'overwhelming' [143]. However, Li describes the method as highly time consuming [82].

The DP algorithm works on the following basis. The first point of the data series is selected as the *anchor point*; the last data point is selected as the *float*

point. For all intermediate data points, the (perpendicular) distance to the line connecting anchor and float points is determined. If the maximum of these distances is greater than a pre-defined threshold, the line is cut at the data point that causes that maximum distance. This cut point becomes the new float point for the first segment, and the anchor point for the second segment. The procedure is recursively repeated for both segments. The algorithm is illustrated in Figure 5.1.

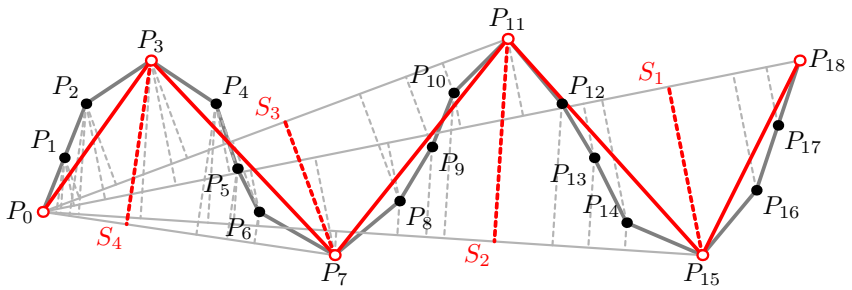


Figure 5.1: *Top-down Douglas-Peucker algorithm. Original data series of 19 points. In this case, only the first segment was recursively cut at data points P_{15}, P_{11}, P_7 and P_3. Compressed data points were also original data points. Original 19 point trajectory in dark grey; final compressed trajectory in red. Also indicated are intermediate segments S_i (in pale grey) created during the computation together with the respective distance chords (dashed, also pale grey) and the threshold violating distance chord, dashed in red. The first chord resolved is S_1, the second S_2, et cetera.*

The DP algorithm clearly is a batch algorithm, as the whole data series is needed at the start; the time complexity of the original algorithm is $O(N^2)$ with N denoting the number of data points in the series. Due to its simplicity, different implementations have been proposed. One such proposal, which succeeded to reduce the complexity of the method from $O(N^2)$ to $(Nlog_2N)$ was Hershberger's proposal [58], who defined the path hull as a basis for their implementation.

5.1.2 Opening window compression algorithms

In opening window (OW) algorithms the start point of a potential segment is considered as the anchor point. The algorithm, then, attempts to approximate the subsequent data series with increasingly longer segments. In other words, the method starts by defining a segment between a first data point (the anchor) and the third data point (the float) in the series. As long as all distances of intermediate data points, i.e., those between anchor and float, are below the distance threshold,

an attempt is made to move the float one point up in the data series. When the threshold is going to be exceeded, two strategies can be applied: either,

- the data point causing the threshold violation (Normal Opening Window, a.k.a. NOW), or
- the data point just before it (Before Opening Window, a.k.a. BOW)

becomes the end point of the current segment, and it also becomes the anchor of the next segment. If no threshold excess takes place, the float is moved one up the data series — the window opens further — and the method continues, until the entire series has been transformed into a piecewise linear approximation. The results of choosing either strategy are illustrated in Figures 5.2 and 5.3.

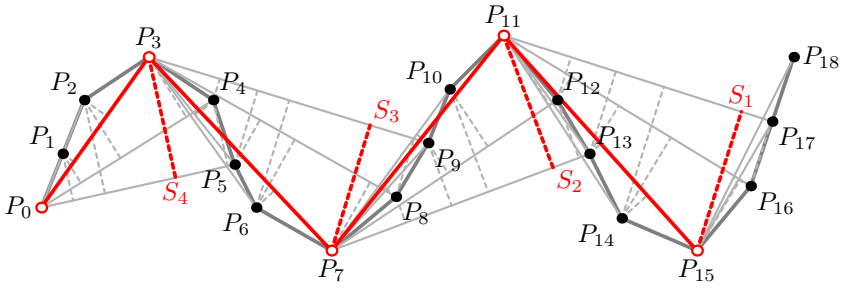


Figure 5.2: *Data series compression result of NOW strategy: the threshold excess data point is the break point. The data series was broken at data points P_3, P_7, P_{11} and P_{15}. For other legend information, see Figure 5.1.*

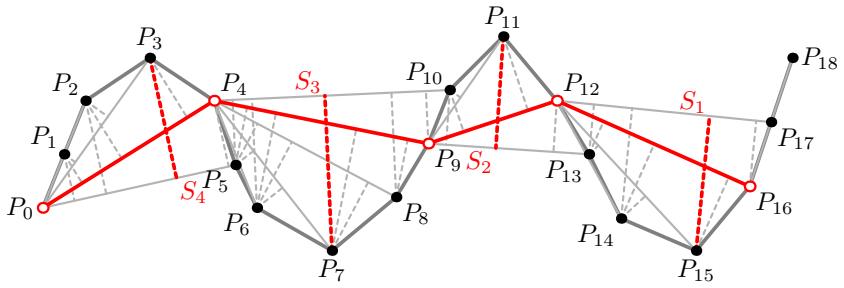


Figure 5.3: *Data series compression result of BOW strategy: the data point just before the threshold excess data point is the break point. The first window opened up to point P_5 (point P_3 causing excess), making point P_4 the cut point; second window opened up to point P_{10} (point P_7 causing excess) with point P_9 becoming the cut point etc. For other legend information, see Figure 5.1.*

An important observation, easily made from Figures 5.2 and 5.3, is that in OW algorithms the last few data points of the data series might be lost. Without countermeasures, this may lead to a inadvertent information loss.

Although OW algorithms are computationally expensive, they are popular. Their popularity comes on the one hand from the fact that they are on-line algorithms, and on the other hand because they can work reasonably well in presence of noise though only for relatively short data series. The time complexity of these algorithms is $O(N^2)$.

5.2 Spatio-temporal compression techniques

5.2.1 Why line generalizations do not quite apply

We discussed the above algorithms because they are well-known techniques for generalizing line structures. All of them use *perpendicular distance* of data points to a proposed generalized line as the condition to discard or retain that data point. This is the mechanism at work also when we apply these algorithms to our data series, the moving object trajectories, viewed as lines in two-dimensional space.

But our trajectories have this important extra dimension, time. Intrinsically, they are not lines, but historically traced points. As a consequence, the use of perpendicular distance as condition is at least challenged, and we should look at more appropriate conditions.

A trajectory is represented as a time series of positions. Generalizing a trajectory means to replace one time series of positions with another one. Like before, we can measure how effective this generalization is by looking at *(a)* the compression rate obtained, and *(b)* the error committed. Unlike before, the error committed is no longer measured through (perpendicular) distances between original data points and the new line, but rather through distances between pairs of temporally synchronized positions, one on the original and one on the new trajectory. This is a fundamental change that does justice to the spatio-*temporal* characteristic of a moving point trajectory.

5.2.2 Single-parameter class of algorithms

The computational consequence of the above arguments is that the decision of discarding a data point must be based on its position *and* timestamp, as well as

on the approximated position of the object on the new trajectory. This gives a distance not necessarily perpendicular to the new, approximated, trajectory.

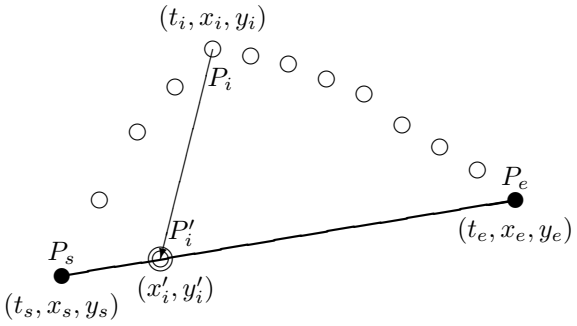


Figure 5.4: *Original data points (open circles), including P_i, the start and end points P_s and P_e of the approximated trajectory, and P_i's approximated position P'_i.*

The situation is illustrated in Figure 5.4, in which the original data point P_i and its approximation P'_i on the new trajectory $P_s - P_e$ are indicated. One should note that our fundamental working assumption has been that object trajectory representation has a piecewise linear approximation. Consequently, we do assume constant speed over the linear piece during the time interval and for any t in the interval we can compute where the object was. The coordinates of P'_i are calculated from the simple *ratio* of two time intervals Δe and Δi, indicating respectively travel time from P_s to P_e (along either trajectory) and from P_s to P_i (along the original trajectory), respectively. These travel times are determined from the original data, as timestamp differences. We have

$$\begin{aligned} \Delta e &= t_e - t_s \\ \Delta i &= t_i - t_s \\ x'_i &= x_s + \frac{\Delta i}{\Delta e}(x_e - x_s) \qquad (5.1) \\ y'_i &= y_s + \frac{\Delta i}{\Delta e}(y_e - y_s) \qquad (5.2) \end{aligned}$$

After the approximate position P'_i is determined, the next step is to calculate the distance between it and the original P_i, and use that distance as a discarding criterion against a user-defined threshold. This is an important improvement not only because we are using a more accurate distance measure but also because the temporal factor is now included. The continuous nature of moving objects neces-

sitates the inclusion of *temporal* as well as *spatial* properties of moving objects.

The application of the above distance notion for moving object trajectories, leads to a class of algorithms that we call here *time-ratio algorithms*.

Further improvements can be obtained by exploiting other spatiotemporal information hiding in the time series. Our time-ratio distance measurement was a first step; a second step can be made by analyzing the derived speeds at subsequent segments of the trajectory, when these are available. A large difference between the travel speeds of two subsequent segments is another criterion that can be applied to retain the data point in the middle. For this, we will assume a *speed difference threshold* will also have been set, indicating above which speed difference we will always retain the data point. The application of the speed threshold notion for moving object trajectories, leads to a class of algorithms that we call here *speed-based algorithms*.

Observe that in principle both these concepts allow application in top-down and opening-window algorithms, as will shortly be shown.

The notations used in following algorithms have already been described in Section 2.4.1.

A top-down time-ratio algorithm (*TD-TR*)

We denote our top-down time-ratio algorithm as *TD-TR*. *TD-TR* is obtained from the DP algorithm through application of the time-ratio distance measuring technique described in Section 5.2.2. The pseudocode for the top-down time-ratio algorithm is provided below.

procedure $TD\_TR(p, dist\_threshold)$

$p \in \mathbb{P}$
$dist\_threshold \in \mathbb{R}$

if $len(p) \leq 2$ **then**
  return p
else $max\_dist\_threshold \leftarrow 0$
  $found\_index \leftarrow 0$
  $\Delta e \leftarrow p[len(p)]_{\mathrm{t}} - p[1]_{\mathrm{t}}$
  for $i \leftarrow 2$ **until** $len(p) - 1$ **do**
    $\Delta i \leftarrow p[i]_{\mathrm{t}} - p[1]_{\mathrm{t}}$
    $(x'_i, y'_i) \leftarrow p[1]_{\mathrm{loc}} + (p[e]_{\mathrm{loc}} - p[1]_{\mathrm{loc}})\, \Delta i / \Delta e$
    if $dist(p[i]_{\mathrm{loc}}, (x'_i, y'_i)) > max\_dist\_threshold$ **then**

```
            max_dist_threshold ← dist(p[i]_loc, (x'_i, y'_i))
            found_index ← i
        end if
    end for
    if max_dist_threshold > dist_threshold then
        return TD_TR(p[1, found_index], dist_threshold) ++
                                    TD_TR(p[found_index, len(p)], dist_threshold)
    else
        return [p[1], p[len(p)]]
    end if
end else
```

A top-down speed-based algorithm (*TD-SP*)

As mentioned in Section 2.4, since we have assumed speed is not directly available from data acquisition phase, it is derived when it is needed for computational purposes. Therefore, two consecutive time-stamped positions are used to calculate the speed of moving object in that time interval. This is done within our top-down speed-based algorithm, which is obtained from the DP algorithm through application of the speed-based algorithm described in Section 5.2.2. The pseudocode for the top-down speed-based algorithm is provided below.

procedure *TD_SP*$(p, speed\_threshold)$

$p \in \mathbb{P}$
$speed\_threshold \in \mathbb{R}$

```
if len(p) ≤ 2 then
  return p
else
    max_speed_threshold ← 0
    found_index ← 0
    for i ← 2 until len(p) − 1 do
        v_{i−1} ← dist(p[i]_loc, p[i − 1]_loc) / (p[i]_t − p[i − 1]_t)
        v_i ← dist(p[i + 1]_loc, p[i]_loc) / (p[i + 1]_t − p[i]_t)
        if ‖v_i − v_{i−1}‖ > max_speed_threshold then
            max_speed_threshold ← ‖v_i − v_{i−1}‖
            found_index ← i
        end if
    end for
```

```
    if max_speed_threshold > speed_threshold then
        return TD_SP(p[1, found_index], speed_threshold) ++
                                        TD_SP(p[found_index, len(p)], speed_threshold)
    else
        return [p[1], p[len(p)]]
    end if
end else
```

An opening-window time-ratio algorithm (*OW-TR*)

An opening-window algorithm sets the anchor point, and then gradually 'opens the window'. In each step, halting conditions are verified. In an opening-window time-ratio algorithm the halting condition is verified on the synchronous distance measure (using the time interval ratio).

We denote our opening window time-ratio algorithm as *OW-TR*. By *OW-TR* we mean an opening-window algorithm applying the same time-ratio distance measuring technique described in Section 5.2.2. The pseudocode for the opening-window time-ratio algorithm is provided below.

```
procedure OW_TR(p, max_dist_error)

p ∈ ℙ
max_dist_error ∈ ℝ

if len(p) ≤ 2
then return p
else is_error ← false
     e ← 3
     while e ≤ len(p) and not is_error do
          i ← 2
          while (i < e and not is_error) do
               Δe ← p[e]_t − p[1]_t
               Δi ← p[i]_t − p[1]_t
               (x'_i, y'_i) ← p[1]_loc + (p[e]_loc − p[1]_loc) Δi/Δe
               if dist(p[i]_loc, (x'_i, y'_i)) > max_dist_error
                    then is_error ← true
                    else i ← i + 1
               end if
          end while
          if is_error then
```

```
            return [p[1]] ++ OW_TR(p[i, len(p)], max_dist_error)
          end if
          e ← e + 1
        end while
        if not is_error
          then return [p[1], p[len(p)]]
        end if
      end if
```

An opening-window speed-based algorithm (*OW-SP*)

The halting condition in an opening-window speed-based algorithm is based on difference in speed values between previous and next trajectory segment. These speeds are *not* measured speeds, as we do not assume these to be available; rather, they are speed values derived from timestamps and positions.

The pseudocode for the opening-window speed-based *OW-SP* algorithm is provided below.

```
procedure OW_SP(p, max_speed_error)
p ∈ ℙ
max_speed_error ∈ ℝ
if len(p) ≤ 2
then return p
else is_error ← false
     e ← 3
     while e ≤ len(p) and not is_error do
          i ← 2
          while (i < e and not is_error) do
            v_{i-1} ← dist(p[i]_loc, p[i − 1]_loc) / (p[i]_t − p[i − 1]_t)
            v_i ← dist(p[i + 1]_loc, p[i]_loc) / (p[i + 1]_t − p[i]_t)
            if ‖v_i − v_{i-1}‖ > max_speed_error
              then is_error ← true
              else i ← i + 1
            end if
          end while
          if is_error
            then return [p[1]] ++ OW_SP(p[i, len(p)], max_speed_error)
          end if
          e ← e + 1
        end while
```

```
    if not is_error then return [p[1], p[len(p)]]
    end if
  end if
```

5.2.3 Double-parameter class of algorithms

By integrating the concepts of *speed difference threshold* and the *time-ratio distance* discussed in Section 5.2.2, we obtain a new algorithmic approach, that we call the class of *full spatio-temporal algorithms*. An important observation is that in this class of algorithms, we deal with two thresholds and therefore have to make a choice which one to exploit first. For a top-down approach, this decision means applying either *TD-TR* or *TD-SP* algorithm first, which is then augmented by the other algorithm. The same applies for an opening-window approach, in which this decision means applying either *OPW-TR* or *OPW-SP* algorithm first and the other one secondly. As we will shortly see, in Section 5.3.3 the results of the two may significantly differ.

We call the algorithm obtained from applying *TD-TR* first and then *TD-SP*, *TD-TR-SP* and the algorithm obtained from applying *TD-SP* first and then *TD-TR*, *TD-SP-TR*.

5.3 Comparisons and results

To assess their appropriateness and relative performance, both spatial and spatiotemporal compression techniques were tested using real moving object trajectory data. In total, we obtained 10 trajectories through a GPS mounted on a car, which travelled different roads in urban and rural areas. The data includes short and lengthy time series; various statistics of our data set are provided in Table 5.1. The original trajectory data were all time series with elements in the form of $\langle t, x, y \rangle$.

First, for each time series the original data was used to determine its corresponding trajectory. Depending on the trip, each trajectory had its own behaviour, ranging from being simple and one-directional to more complicated consisting of several loops, or being bi- or multi-directional. Moving objects like cars are subject to sudden stops, abrupt changes of direction and multiple visits to the same location in case of bi- or multi-directional trajectories. Therefore, the type of the trip may influence the performance of the compression techniques in use. Our experimentation data set had such non-trivial examples.

stat	*average*	*standard deviation*
duration	00:32:16	00:14:33
speed	40.85 km/h	12.63 km/h
length	19.95 km	12.84 km
displacement	10.58 km	8.97 km
# of data points	200	100.9

Table 5.1: *Statistics on the ten moving object trajectories used in compression experiments.*

As already mentioned, in using lossy compression techniques there is always a trade-off between compression rate achieved and error allowed. In our case, the optimal compression technique should find a subseries of the original time series that has a high enough compression and a low enough error. This error notion is the main tool for evaluating the quality of the compression technique. But the literature has not paid a lot of attention to explicitly measuring error. Mostly, attention has been given to the identification of proper heuristics for discarding data points.

Such heuristics may or may not be proper for compressing moving object trajectories. But since a notion of error committed is highly needed, we introduce an error formula for moving object trajectory compression in Section 5.3.1.

5.3.1 Error notions

The quality of compression techniques is evaluated by using some notion of error. Various mathematical measures have been proposed for this error notion, for instance by [85, 61]. Error notions can be based on different principles: length, density, angularity and curvilinearity. Some of these notions have been used to improve the appeal of the approximation of linear features in GIS context.

Distance-based error notions seem less biased towards visual effect. In plain line generalization, perpendicular distance is the error measure of choice. A simple method determines all distances of original data points to the approximation line, and determines their average, but this is sensitive to the actual number of data points. A method least sensitive to this number essentially determines the area between original line and approximation. It effectively assumes there were infinitely many original data points.

How would such an error measure translate to the spatio-temporal case of object movement? Applying still perpendicular distances, insensitivity to the number

of data points can be obtained by considering progressively finer sampling rates, as illustrated in Figure 5.5a. In this figure, p represents an original trajectory with five segments, a represents its 1-segment approximation. The t_i are equally spaced time instants. On slower segments (like the first and third), distance chords become more condensed. For progressively finer sampling rates, this error notion becomes the sum over segments of weighted areas between original and approximation. The associated formulas are simple and direct.

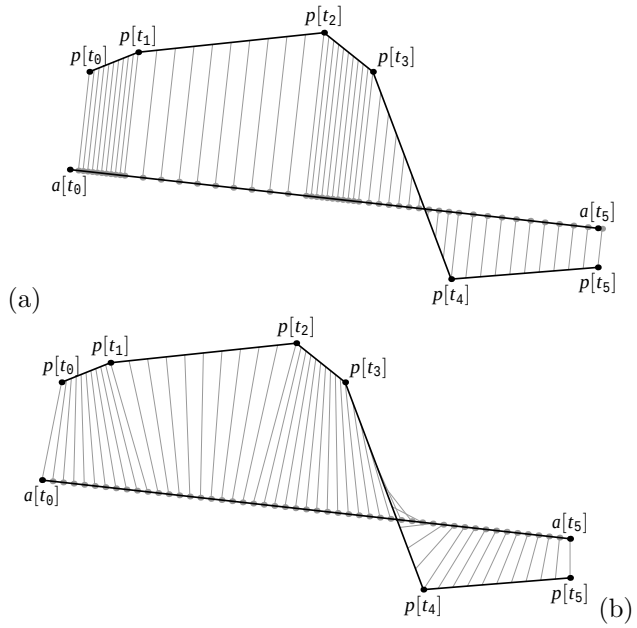


Figure 5.5: *Two error notions for a trajectory p being approximated by a. (a) error measured at fixed sampling rate as sum of perpendicular distance chords; (b) error measured at fixed sampling rates as sum of time-synchronous distance chords.*

5.3.2 A spatio-temporal error notion

Given an original trajectory ($p : \mathbb{P}$) and an approximation trajectory ($a : \mathbb{P}$) of it, we are interested in finding a measure that expresses without bias how well the second approximates the first. The plain intuition we have for this is that the *average distance between the original object and the approximation object*—both *synchronously* travelling along their respective trajectories p and a—during the time interval of their trajectories is the best measure that one can have. We develop

the formulas for that average distance. It should be mentioned that Nanni [96] has also used a similar concept to address spatio-temporal distance measure, though in a more general setting, and not as an error notion, thus not providing the full details of our formulas.

We assume that both trajectories (original p and approximation a) are represented as series of time-stamped positions, which we will interpret as piecewise linear paths. Given the classes of compression algorithms that we study, we can also safely assume that the time stamps present in the trajectory a form a subseries of those present in the original p. After all, we have obtained that series by discarding data points in the original, and we never invented new data points, let alone time stamps. Assume that p has data points numbered from 1 to k.

Following from the above, we can define the *average synchronous error* $\alpha(p, a)$ between p and a as a summation of the weighted contributions of all linear segments in p.

$$\alpha(p,a) = \frac{\sum_{i=1}^{k-1} (p[i+1]_{\mathrm{t}} - p[i]_{\mathrm{t}}) \cdot \alpha(p[i:i+1], a)}{\sum_{i=1}^{k-1} p[i+1]_{\mathrm{t}} - p[i]_{\mathrm{t}}}. \tag{5.3}$$

We have not defined the function α fully in this way. Equation 5.3 covers the case that p is a multi-segment trajectory, but not the case that it is a single segment. We will cover that case through Equation 5.4 below. Do observe, however, that the denominator of Equation 5.3 can be simplified to $p[k]_{\mathrm{t}} - p[1]_{\mathrm{t}}$, the length of p's time interval.

Now let us derive the *single segment average synchronous error*. Let q be a single segment trajectory, thus defined by only two time-stamped data points. Observe that the position of an object moving along q is a function of time that can be expressed in the style of Equations 5.1 and 5.2 as

$$loc\_one(q,t) = q[1]_{\mathrm{loc}} + \frac{t - q[1]_{\mathrm{t}}}{q[2]_{\mathrm{t}} - q[1]_{\mathrm{t}}} (q[2]_{\mathrm{loc}} - q[1]_{\mathrm{loc}}) \qquad \text{for } t \in [q[1]_{\mathrm{t}}, q[2]_{\mathrm{t}}].$$

We generalize this notion of object position for an arbitrary length trajectory p in the obvious way, such that $loc(p,t) : \mathbb{P} \to (\mathbb{T} \to \mathbb{L})$ is a total function such that for any p, $loc(p)$ is a partial function with domain $[p[1]_{\mathrm{t}}, p[len(q)]_{\mathrm{t}}]$. Formally, we

have

$$loc(p,t) = \begin{cases} p[i]_{\text{loc}} & \text{if } t = p[i]_{\text{t}} \text{ for some } i \text{ with } 1 \le i \le len(p) \\ loc\_one(p[i:i+1],t) & \text{if } t \in \langle p[i]_{\text{t}}, p[i+1]_{\text{t}} \rangle. \end{cases}$$

It is a trivial exercise to show that $loc(p)$ is a continuous function within its domain, since at the potential discontinuities—the stored time stamps $p[i]_{\text{t}}$—the function formats coincide value-wise.

With the single segment average synchronous error $\alpha(p[i:i+1],a)$ we express the average distance between original and approximate object during the time interval between indices i and $i+1$, for convenience written as $[t_i, t_{i+1}]$. It can be expressed as

$$\alpha(p[i:i+1],a) = \frac{1}{t_{i+1}-t_i} \int_{t_i}^{t_{i+1}} dist(loc(p,t), loc(a,t))\, dt. \tag{5.4}$$

In our further derivations, differences in coordinate values at time instants t_i and t_{i+1} between p and a (in that order) will be important. We will denote these (four) differences, respectively as $\delta x_i, \delta x_{i+1}, \delta y_i$ and δy_{i+1}. Observe that these are constants; they are illustrated in Figure 5.6 and are derived from the following formulas:

$$\begin{aligned} \delta x_i &= loc(p,t_i).x - loc(a,t_i).x \\ \delta y_i &= loc(p,t_i).y - loc(a,t_i).y \\ \delta x_{i+1} &= loc(p,t_{i+1}).x - loc(a,t_{i+1}).x \\ \delta y_{i+1} &= loc(p,t_{i+1}).y - loc(a,t_{i+1}).y \end{aligned}$$

Since both p and a during the given time interval are represented as straight segments, the distance formula in Equation 5.4 can be derived as having a well-known polynomial format:

$$dist(loc(p,t), loc(p,t)) = \frac{1}{c_4}\sqrt{c_1 t^2 + c_2 t + c_3}, \qquad \text{where}$$

$$
\begin{aligned}
c_1 &= (\delta x_i - \delta x_{i+1})^2 + (\delta y_i - \delta y_{i+1})^2 \\
c_2 &= 2 \cdot ((\delta x_{i+1} t_i - \delta x_i t_{i+1}) \cdot (\delta x_i - \delta x_{i+1}) + (\delta y_{i+1} t_i - \delta y_i t_{i+1}) \cdot (\delta y_i - \delta y_{i+1})) \\
c_3 &= (\delta x_{i+1} t_i - \delta x_i t_{i+1})^2 + (\delta y_{i+1} t_i - \delta y_i t_{i+1})^2 \qquad \text{and} \\
c_4 &= t_{i+1} - t_i \ .
\end{aligned}
$$

Using the above results we can rewrite Equation 5.4 to

$$\alpha(p[i:i+1], a) = \frac{1}{(t_{i+1} - t_i)^2} \int_{t_i}^{t_{i+1}} \sqrt{c_1 t^2 + c_2 t + c_3}\, dt. \tag{5.5}$$

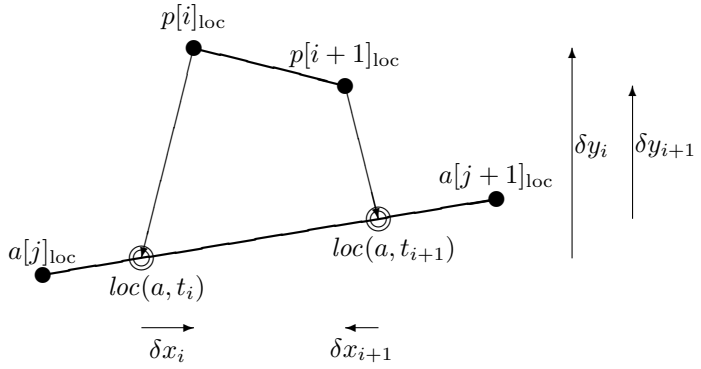


Figure 5.6: *Definition of $\delta x_i, \delta x_{i+1}, \delta y_i$ and δy_{i+1}. Illustrated is the i-th segment of original path p (at top), between points $p[i]_{\text{loc}}$ and $p[i+1]_{\text{loc}}$, and its approximation $loc(a, t_i)$–$loc(a, t_{i+1})$ on trajectory a. Observe that this approximation will be part of an a-segment, but that its start and end points may or may not coincide with real data points of a.*

The solution to Equation 5.5 depends on the values for the constants c_i. We provide a case analysis, omitting cases that cannot happen due to the structure—expressed in the equations for constants c_i—of the problem.

Case $c_1 = 0$: This happens when $\delta x_i = \delta x_{i+1}$ and $\delta y_i = \delta y_{i+1}$. If so, then also $c_2 = 0$, and the solution to Equation 5.5 is

$$\alpha(p[i:i+1], a) = \frac{\sqrt{c_3}}{t_{i+1} - t_i}.$$

The geometric interpretation of this case is simple: equality of δ's indicates

that the approximation of this p segment is a vector-translated version of that segment. The distance is thus constant, and its average over the time interval equals that constant. We have:

$$\boxed{\alpha(p[i:i+1],a) = \sqrt{\delta x_i^2 + \delta y_i^2}.}$$

Case ($c_1 > 0$) : This happens when $\delta x_i \neq \delta x_{i+1}$ or $\delta y_i \neq \delta y_{i+1}$. The solution to the integral part of Equation 5.5 is non-trivial, and deserves a case analysis in itself. We have the following cases:

Case $c_2^2 - 4c_1c_3 = 0$: In other words, the determinant of the quadratic subformula of Equation 5.5 equals 0. This happens only when $\delta x_i \delta y_{i+1} = \delta x_{i+1}\delta y_i$. If so, the solution to Equation 5.5 is

$$\boxed{\alpha(p[i:i+1],a) = \frac{1}{(t_{i+1}-t_i)^2}\left|\frac{2c_1t+c_2}{4c_1}\sqrt{c_1t^2+c_2t+c_3}\right|_{t_i}^{t_{i+1}}.}$$

The geometric interpretation of this case follows from the δ product equality mentioned above. That equality holds in three different cases. These cases are not mutually exclusive, but where they are not, the formulas coincide value-wise.

Case segments share start point : Consequently, we have $\delta x_i = \delta y_i = 0$. The above formula simplifies to

$$\boxed{\alpha(p[i:i+1],a) = \tfrac{1}{2}\sqrt{\delta x_{i+1}^2 + \delta y_{i+1}^2}.}$$

Case segments share end point : Consequently, we have $\delta x_{i+1} = \delta y_{i+1} = 0$. The above formula simplifies to

$$\boxed{\alpha(p[i:i+1],a) = \tfrac{1}{2}\sqrt{\delta x_i^2 + \delta y_i^2}.}$$

Case δ ratios respected : This situation is illustrated in Figure 5.7. It turns out that under the condition of respected δ ratios, the synchronous distance chords (indicated as grey edges in Figures 5.5 and 5.7) all lie parallel to each other, and this simplifies the related formulas.

Case $c_2^2 - 4c_1c_3 < 0$: The determinant of the quadratic subformula of Equa-

tion 5.5 is less than 0. This is the general, non-exceptional case. The resulting formula is:

$$\alpha(p[i:i+1],a) = \frac{1}{(t_{i+1}-t_i)^2}\,|F(t)|_{t_i}^{t_{i+1}} \quad \text{where}$$
$$F(t) = \frac{2c_1t+c_2}{4c_1}\sqrt{c_1t^2+c_2t+c_3} - \frac{c_2^2-4c_1c_3}{8c_1\sqrt{c_1}}\operatorname{arcsinh}\left(\frac{2c_1t+c_2}{\sqrt{4c_1c_3-c_2^2}}\right) .$$

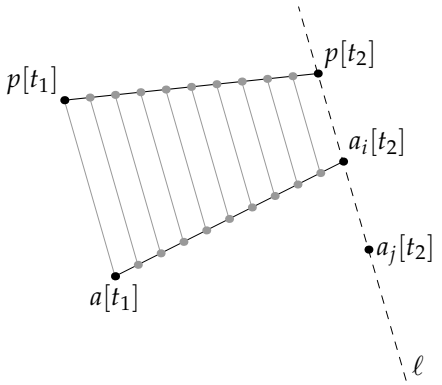


Figure 5.7: *The case of respected δ ratios illustrated. The synchronous distance chords lie parallel to each other. Given original segment $p[t_1]p[t_2]$ and start of approximation segment $a[t_1]$, any end point $a_i[t_2]$ of that segment on line ℓ will cause the δ ratios to be respected. The end point $a_j[t_2]$ is a special case of these, as this reverts back to case ($c_1 = 0$).*

5.3.3 Experimental results

As mentioned earlier, all the compression techniques were tested on real data as summarized in Table 5.1; i.e., ten different trajectories, for fifteen different spatial threshold values ranging from 30 to 100 m, and three speed difference threshold values ranging from 5 to 25 m/s. The obtained results for each experiment consist of error produced and compression rate achieved. We used the time synchronous error notion derived in Section 5.3.2. It is important to note that the choice of optimal threshold values is difficult and might differ for various applications.

The first experiment concerned the comparison between conventional top-down (Normal) DP *(NDP)* and our top-down time-ratio *(TD-TR)* algorithm. Figures 5.8 shows the results.

Clearly, the TD-TR algorithm produces much lower errors, while the compression rate is only slightly lower. An important observation is that both compres-

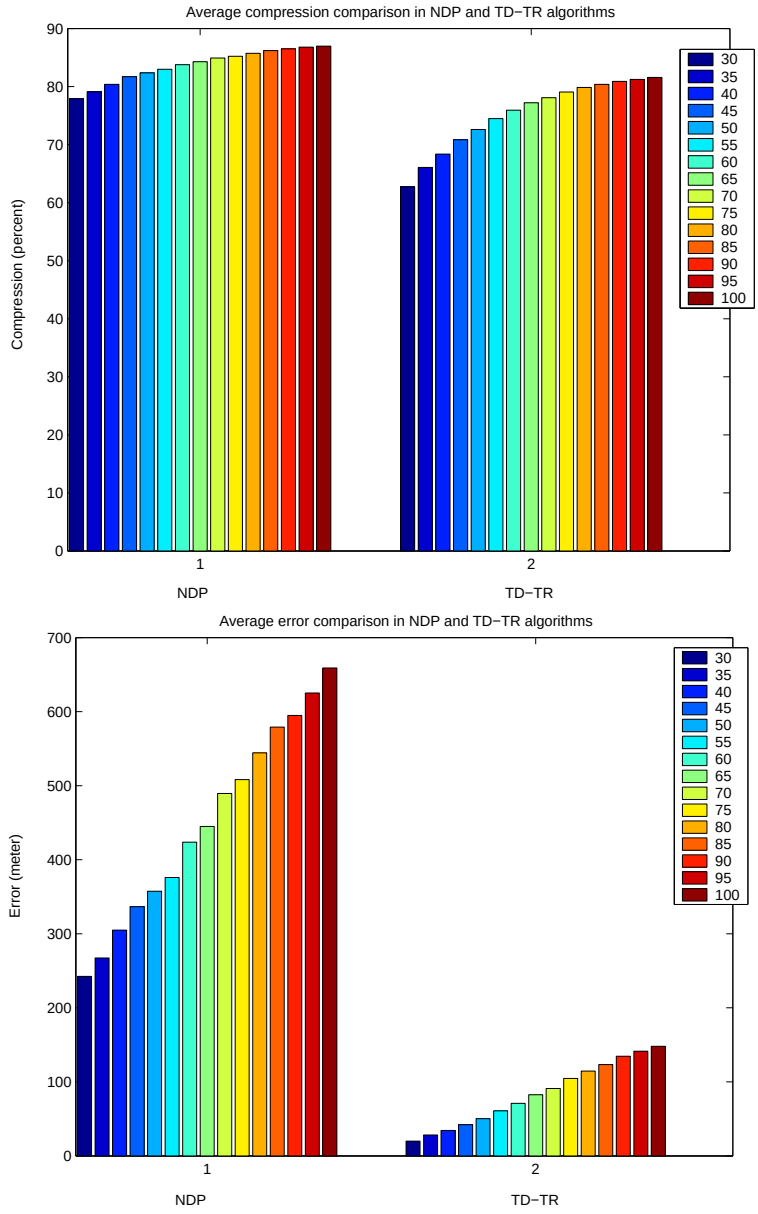


Figure 5.8: *Comparison between NDP (on left) and TD-TR (on right) algorithms.Colour bars indicate different settings for distance threshold, ranging from 30 to 100 m. (a) Comparison of compression percentages achieved; (b) Comparison of errors committed. Figures given are averages over ten different, real trajectories.*

sion rate and error *monotonically* increase with distance threshold, asymptotically reaching a maximum.

The second experiment concerned the choice of break point in opening window algorithms; i.e., whether to break at the data point causing threshold excess (NOW) or at the one just before (BOW). We found that BOW results in higher compression but worse errors. It can be used in applications where the main concern is compression and one is willing to favour it over error. For most of our application domains, we aim at lower error and high enough compression rate, and we will therefore ignore the BOW method for further comparisons. Results of comparison between BOW and NOW algorithms are shown in Figure 5.9.

One may observe that error in both cases of the NOW and BOW algorithms suddenly drops at 80 meter threshold. (Similar effects can be found in Figures 5.10.) This effect is likely only an artifact caused by the small set of trajectories in our experiments. Nevertheless, it shows importance of choosing an appropriate threshold in opening window algorithms.

In Figure 5.10, we illustrate the findings of a third experiment, comparing our opening window time-ratio with a normal opening window algorithm. It demonstrates the superiority of the first (OW-TR) with respect to the latter (NOW) algorithm. As can be seen, for OW-TR a change in threshold value does not dramatically impact error level. Therefore, unlike the NOW algorithm, in which a choice of threshold is important for the error results, in the OW-TR algorithm this is not the case. This allows choosing a higher threshold value to improve compression while not loosing much on error performance.

Our second spatio-temporal algorithm (SP) was applied in both opening window and top-down fashion. In case of top-down SP (TD-SP), experiments show a high sensitivity towards speed threshold settings, for error committed. Among the three speed difference threshold values (5, 15, 25 m/s), only the first value provided reasonable results, and is therefore included in the comparisons. In opening window fashion SP (OW-SP) changes in threshold value did not lead to dramatic changes in the results. Comparing the results of the three speed threshold in OW-SP algorithm shows that 15 and 25 m/s speed threshold give the best outcomes in both compression and error and increasing the threshold after 15 m/s do not improve the results. As can be seen from Figure 5.11, the two OW-SP (15 m/s, 25 m/s) algorithms as well as OW-TR have very similar behaviour in compression rate, while OW-SP has lower error. On the other hand, choosing a speed difference threshold of 5 m/s in TD-SP and OW-SP results in improved compression

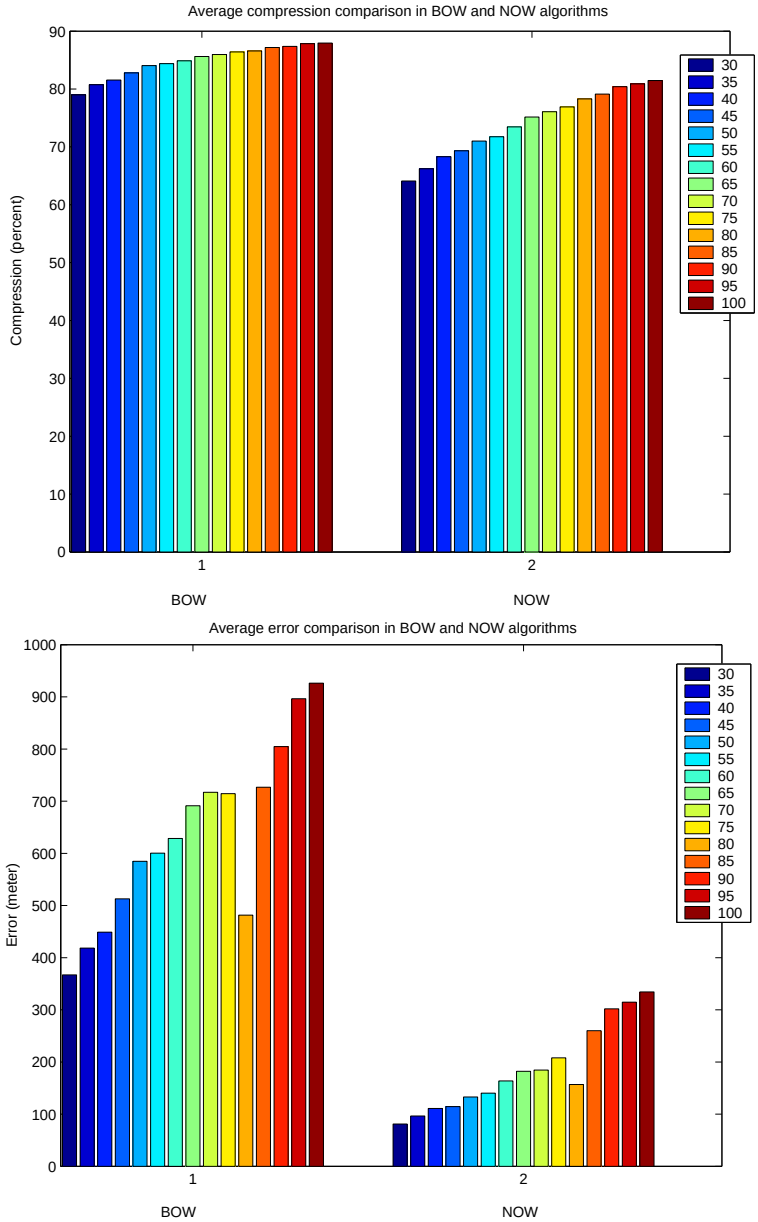


Figure 5.9: *Comparison between two opening window algorithms, BOW (on left) and NOW (on right) algorithms. For other legend information, see Figure 5.8.*

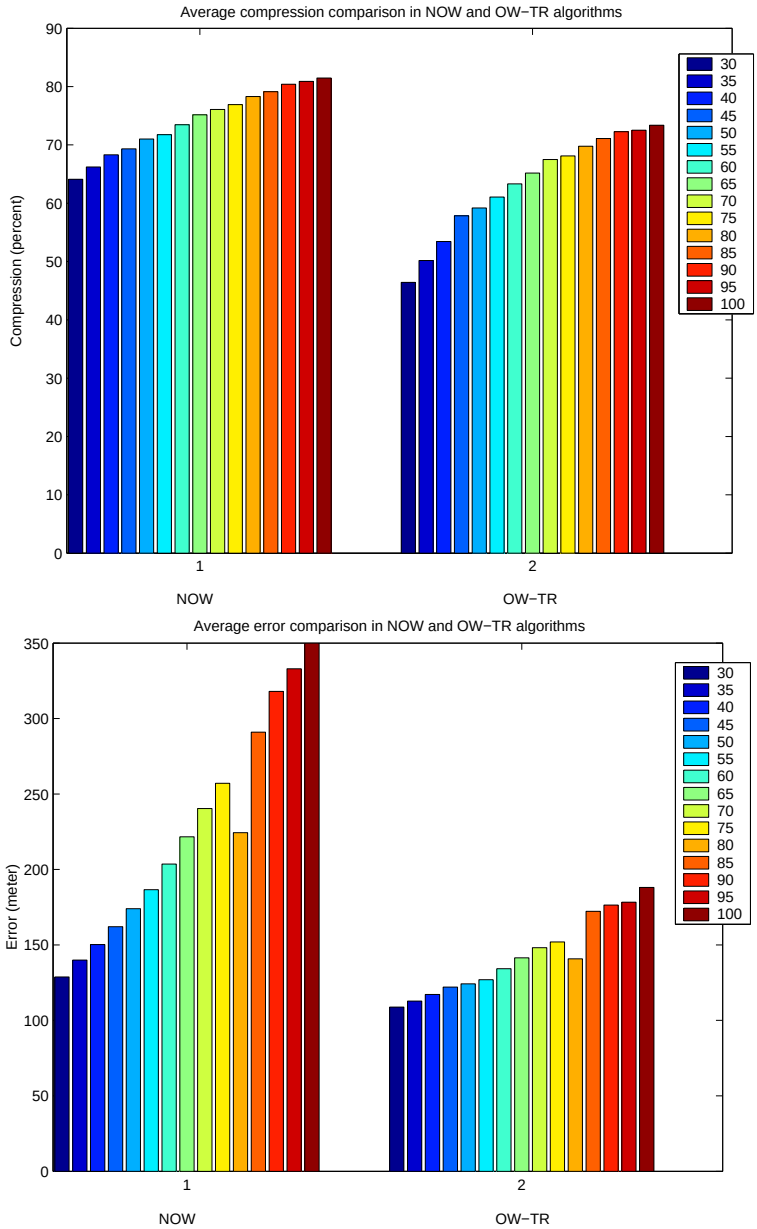


Figure 5.10: *Comparison between NOW (on left) and OW-TR (on right) algorithms. For other legend information, see Figure 5.8.*

but higher error in TD-SP.

As mentioned in Section 5.2.3, since there are two thresholds in our spatiotemporal algorithm, the order in which these thresholds are applied matters. Our experiments show that the result of applying speed threshold first and then distance threshold (TD-SP-TR) is better. In comparison between TD-SP-TR and TD-TR-SP, the former results in lower error committed, while compression rates achieved are just slightly different. On the other hand, results of TD-TR-SP shows once again the importance of choosing a right threshold, since slight change in distance threshold dramatically changes the error committed. Results of these experiments as well as a comparison between conventional DP and our single-parameter class of algorithms are shown in Figure 5.12. Clearly, TD-TR algorithm outperforms all.

As experiments show reasonably high compression rates as well as low errors for our TD-TR, OW-TR and OW-SP algorithms, they effectively fulfill the requirement of *good* compression techniques. A final comparison between these algorithms ranks the TD-TR slightly above their counterparts because of better compression rate and an acceptable error committed. The results of this final comparison are shown in Figure 5.13.

However, two issues should not be forgotten. One is that TD-TR is a batch algorithm, while OW-TR and OW-SP are on-line algorithms. Therefore, TD-TR can be used stand alone. The other issue is that the higher compression rate in TD-TR is accompanied by slightly higher error. Therefore, depending on data availability and error allowed, any of the mentioned algorithms can be used.

It clearly shows that algorithms developed with spatiotemporal characteristics outperform others. One should note that although a noticeable improvement is made by applying speed threshold only, obtained results are significantly enhanced by integrating speed as well as time-ratio distance characteristics. Another important observation is that the choice of threshold value for TD-TR-SP is crucial, as it may rank it higher or lower than TD-SP-TD.

5.4 Compression of network-constrained trajectories

Up to here, our technique of data compression was to identify data points that are not so informative and then to discard them. This makes the trajectory sequence smaller, and thus more compressed. We continued to work with original data

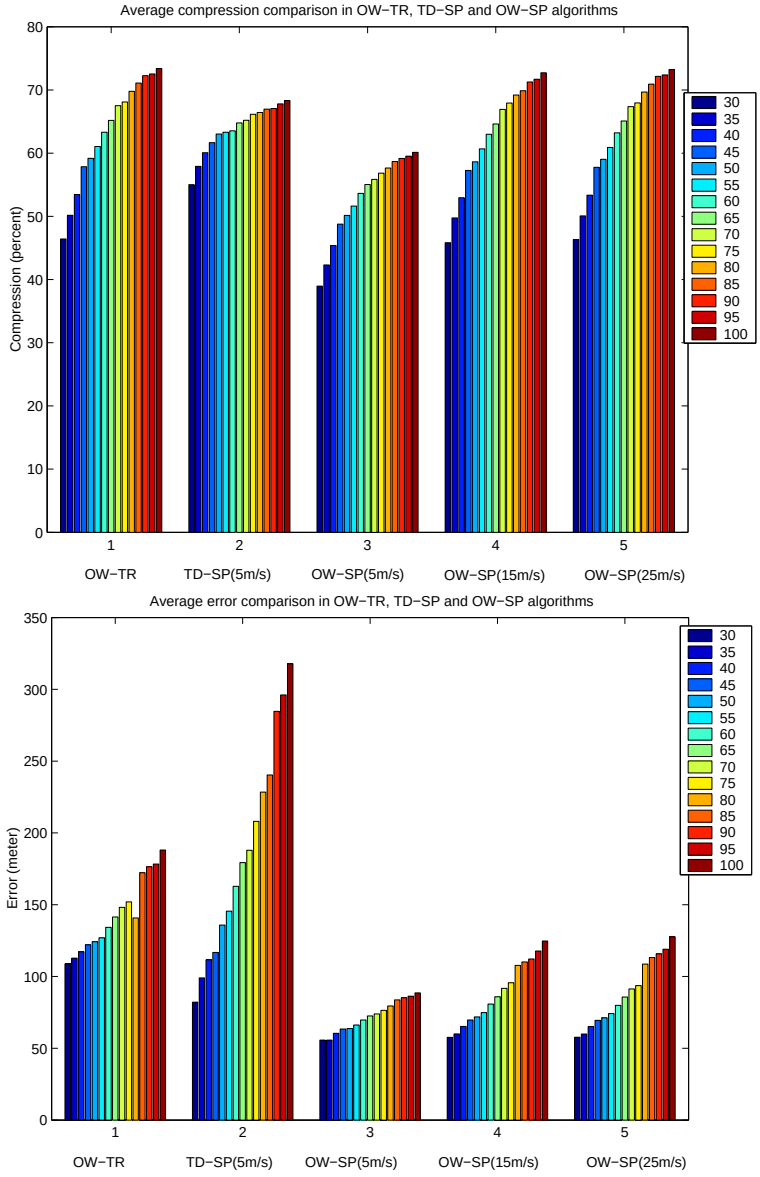


Figure 5.11: *Comparison between OW-TR, TD-SP and OW-SP algorithms; Errors committed; Compression obtained. In both figures, the graph for OW-SP-15m/s coincides with that of OW-SP-25m/s.*

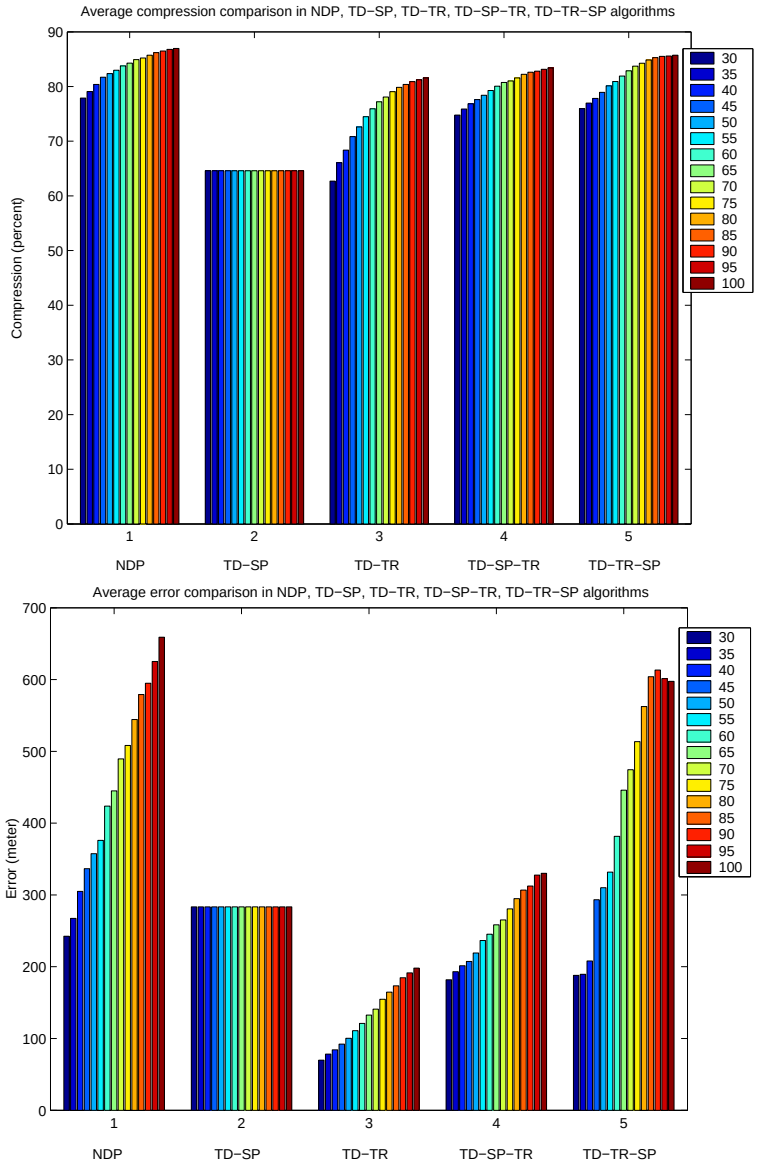


Figure 5.12: *Comparison between NDP, TD-SP, TD-TR, TD-SP-TR, TD-TR-SP algorithms. For other legend information, see Figure 5.8*

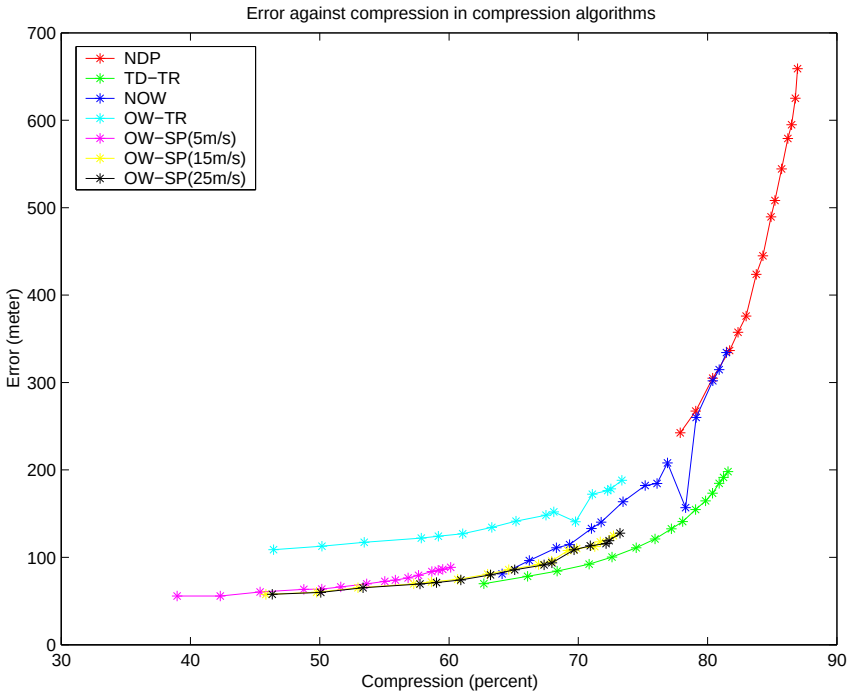


Figure 5.13: *Error versus compression in NDP, TD-TR, NOW, OW-TR and OW-SP algorithms. As before, the graph for OW-SP-15m/s coincides with that of OW-SP-25m/s.*

points. Essentially, we tried to find a subsequence of the original sequence that has a high enough compression and a low enough error. However, in network-constrained trajectories, it is tempting to derive a technique that invents data points to add to the sequence so that *(i)* we still obtain good data compression, and *(ii)* lower the error even more. Inclusion of data about the medium of travel on the one hand and speed and direction in the obtained data on the other hand, seem to provide more information to facilitate achieving such a goal.

5.4.1 Inclusion of network

Having a network-constrained moving object trajectory in the form of a sequence of time-stamped positions, performing the following steps will result in compression of the object trajectory:

- Determine the object's trajectory on the network:

  By applying the snapping technique explained in Section 3.3, the sequence of network segments visited by the object is identified, which in turn results in determining the trajectory of the object on the network.

- Determine the position of intersection nodes between visited network segments as well as time of object transit:

  The position of the intersection node, $P(R_i; R_j)_{\text{loc}}$, between network segments R_i and R_j can easily be found by intersecting the two network segments. The time at which the object transited this node, $P(R_i; R_j)_{\text{t}}$, can be determined with the following equation, in which p[i] and p[i+1] represent the last data point on segment R_i and the first data point on segment R_j:

$$P(R_i; R_j)_{\text{t}} =$$

$$\frac{dist(p[i]_{\text{loc}}, P(R_i; R_j)_{\text{loc}}).p[i+1]_{\text{t}} - dist(P(R_i; R_j)_{\text{loc}}, p[i+1]_{\text{loc}}).p[i]_{\text{t}}}{dist(P(R_i; R_j)_{\text{loc}}, p[i+1]_{\text{loc}}) - dist(p[i]_{\text{loc}}, P(R_i; R_j)_{\text{loc}})} \quad (5.6)$$

A sequence of time-stamped positions of the first and last data points (origin and destination of the object) on the trajectory on the network plus the ordered intermediate intersection nodes results in a compressed trajectory of the object.

This method provides a rather abstract, though, reasonable understanding of a trajectory and its general behaviour. It performs fairly well on simple trajectories, i.e., without loops, U-turns, and backward travelling. Figure 5.14 shows principles

of the explained compression technique, in which the first and the last data points of the trajectory snapped to the network plus five intersection nodes of network segments together with their corresponding time-stamps form the compressed trajectory, i.e., $P_{orig}, P_{(R_3;R_2}, P_{(R_2;R_5}, P_{(R_5;R_{10}}, P_{(R_{10};R_{13}}, P_{(R_{13};R_{14}}, P_{dest}$.

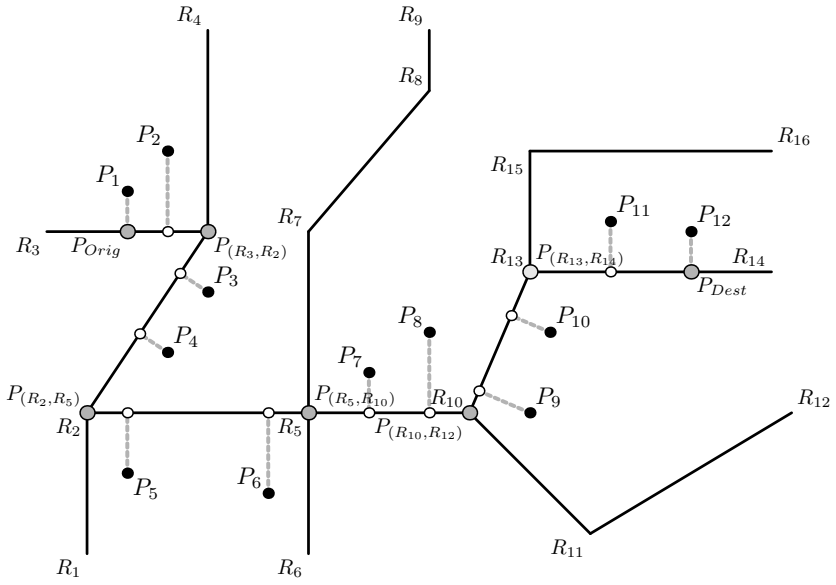


Figure 5.14: *Compressing a network-constrained trajectory with help of network data. Original data points shown in black circles, snapped data points to the network shown in white circles, and compressed data points shown in gray circles.*

In dense networks and networks with short segments, the achieved compression rate may be enhanced by eliminating intersections at which the direction of the two intersecting segments do not change dramatically. This can be observed in Figure 5.14. Since there is no directional change between segments R_5 and R_{10} (which intersect in $P_{(R_5;R_{10})}$), eliminating this node improves the compression rate achieved without loosing too much information.

One should note that this method is applied on the data points mapped to an underlying network rather than on original data points. It also assumes that the object has constant speed and direction on each segment. Thus, the error committed may not be so desirable. Nevertheless, it is good for applications that are more concerned with the compression rate and pay little attention to the error committed.

5.4.2 Inclusion of speed and direction

As explained earlier, since the previous method assumes piece-wise constant speed and direction for the moving object, it fails to identify and, thus, to represent the object's different patterns of movement on each network segment. The direct consequence of this is to lose information that may be useful (depending on application). To avoid information loss as well as to lower the error committed, a change in speed and direction can be considered. Obviously, speed and direction data should either be directly available from the data acquisition phase, or like in our case, be computed during compression.

The idea behind this compression technique is the same as explained for our break point extraction algorithm explained in Section 4.2.1. This means that any abrupt change in either direction or speed should be identified and the data point responsible for this should be kept in the compressed trajectory. Earlier, in Section 4.2.1, we called such data points break points. Therefore, in this method a sequence of time-stamped positions of the first and the last data point together with all intermediate break points form the compressed trajectory of the object.

In principle, since this method is independent of an underlying network, it can also be applied on data for unconstrained moving objects. However, for constrained moving objects it is applied on data points of the object's trajectory on the network. The reason for this is to avoid incorporation of the error associated with the original data in compression. This effect can be clearly seen from Figure 5.14. For instance, applying the technique on original data results in keeping both P_5 and P_6 points, due to the dramatic directional change between the two. However, these two data points should actually be on the network and the error associated with them has prevented the method from doing so. Therefore, if this compression technique is applied once again and this time on the object trajectory on the network, both these data points are disregarded from the compressed trajectory.

Compared to the previous method, a lower error is expected, while the compression rate achieved highly depends on the behaviour of the object itself, especially on each segment.

While integration of these two compression techniques is also possible, one should be aware of the trade-off between achieved compression rate and information loss. Inclusion of both segments' intersection nodes and break points will increase expressiveness of the compressed trajectory while at the same time it decreases the compression rate. Therefore, any compression technique should be

chosen based on the application requirements.

5.5 Lessons learnt and plans ahead

Existing compression techniques commonly used for line generalization in the GIS domain or for time-series data in the data mining field are not suitable for moving object trajectory applications. They are good for short time series and in absence of noise, which is definitely not the case for moving objects. Another important reason for their inapplicability is that they operate on the basis of perpendicular distances. Due to the continuous nature of moving objects, both spatial and temporal factors should be taken into account compressing their data.

In this chapter, problems of existing compression techniques for moving object application are addressed. Two spatio-temporal techniques are proposed to overcome problems for unconstrained moving objects. The quality of the methods was tested using a new and advanced error notion. The experiments confirm the superiority of the proposed methods to the existing ones. The proposed algorithms are suitable to be used both in *batch* and *on-line*. Obtained results strongly depend on the chosen threshold values. Choosing a proper threshold is not easy and is application-dependent. However, having a clear understanding of moving object behaviour helps in making these choices.

Compression techniques for unconstrained moving objects are applied on data points that are originally present. Data points that are not so informative are identified and then discarded. Another strategy has been used for compressing constrained moving objects data based on inventing new data points to add to the sequence. Consequently, two compression techniques were proposed on the basis of inclusion of data about the medium via which object is moving in one hand and speed and direction in the obtained data on the other hand.

Since, there is always a trade off between compression rate achieved and error committed (information loss), any compression technique should be chosen based on the application requirements.

Piecewise linear interpolation was used as the approximation technique. Considering that other measurements such as momentaneous speed and direction values are sometimes available, other, more advanced, interpolation techniques and consequently other error notions can be defined. This issue together with behaviour study of moving objects of different nature can be addressed in future research.

Chapter 6

Similarity notions for moving object trajectories*

Many of life's failures are people who did not realize how close they were to success when they gave up.

Thomas A. Edison (1847 – 1931)

One of the important motivations for work on (historically traced) object trajectories is the opportunity to analyze object traffic. Such analysis may lead to insights important for understanding object behaviour, whether animate or inanimate, as well as for supporting planning and other forms of decision making.

Object traffic can be defined as the aggregate of individual object movement, over some period of time. Fundamental in the analysis of object traffic, and thus, of large amounts of object trajectories, is the capability of pattern detection: which objects have moved in similar ways? It is this question, under a number of disguises, that we want to address here.

Moving objects commonly display repeated patterns of movement, the understanding of which is often highly useful for planning and management purposes. A movement may be *periodically repeated* as in animal migration, commuter traffic, or in shopping. A *movement pattern* then emerges. Movement may also be singular, and one-off. Obviously, one would like to separate patterns from one-off movements; even more, one would like to classify repeated movements in different classes. Solutions to this problem can be used for various purposes such as traffic summaries, commuter path commonalities and other questions of trajectory aggregation that take moving object querying to a level above querying individual

*This chapter is partially based on papers (1) published in the Proceedings of the 10^{th} ACM International Symposium on Advances in Geographic Information Systems (ACM-GIS 2002), Agnès Voisard and Shu-Ching Chen (Eds.), 8–9 November 2002, McLean, VA, USA, pp. 49–54, ISBN 1-58113-591-2, and (2) "Aggregation of moving object trajectories based on similarity," submitted for publication.

trajectories.

Classification requires a notion of equivalence (where the data is crisp) or similarity (where the data has a dimension of vagueness or uncertainty). Standard database aggregate functions also perform partitions, but these are essentially always based on notions of equivalence. The fundamental difference here is caused by the associativity of equality: given some partitioning criterion C, if objects a and b fall in the same partition, we have $C(a) = C(b)$. If object c is known to be in there as well, we can infer $C(a) = C(c)$. Similarity, however, is a drifting characteristic that does not have this associative nature. We have already seen in Section 4.2.1 the arguments why object trajectories are best handled under a scheme of similarity. Given a large set of object trajectories, we want to develop techniques to partition these trajectories into groups, such that each group consists of all and only those trajectories that are similar to each other, under some definition of similarity. The question remains what constitutes a proper definition of *moving object* or *trajectory similarity*.

Various problems arise when one tries to define this notion. One needs to ask when two trajectories are sufficiently similar, for the analytic purposes at hand, to be classified in the same way? The analytic purpose is an important factor as we will see below. Trajectories may be similar for all of their extent or only for a part of it; they may approximately coincide in start and end points, yet be largely dissimilar internally; they may be largely identical spatially but not temporally. Depending on analytic purpose, the emphasis put on these issues may be different. In short, it appears that a single notion of similarity does not suffice, and that we need to select the notion with the purpose in mind.

So what are the dimensions of trajectory similarity? And in which way must these be reflected in respective definitions? Some of the important issues are listed here:

- Three important factors play a role: time, space, and space/time. Two trajectories may be similar on all three factors, or just on a subset of them.
- Trajectories are two- or even three-dimensional data time series.
- Sampling rates vary within a trajectory and differ between them; likewise, neither clock readings nor positioning will be error-free: the spatio-temporal data underlying trajectories is inherently uncertain.
- Object movement is monitored for varying periods of time which leads to unequal monitoring durations and periods for different objects.

- Shifts in time and/or space should, under certain analytic schemes, not be a hindrance in co-classifying trajectories.

The majority of research conducted on similarity search is based on finding patterns similar to a given pattern. Therefore, the problems addressed are

1. how to define similarity between two pattern, here between two data sequences, and

2. how to identify the given pattern against which comparisons are made.

The simplest approach is to define the data sequence as a vector and to use Euclidean distance between such vectors as similarity measure. This approach has been widely used, see for instance [9, 24, 37, 49, 50, 68, 73, 112, 113].

A Euclidean distance technique cannot be easily applied to data sequences of variable length or, indeed in our case, with varying sampling rate, which are two typical properties of trajectories. Also, it is believed not to be effective in the presence of noise in the data, and this is common in spatial time series. On the other hand, the advantage of using Euclidean distance is that it can be used in various dimensionality reduction techniques such as Discrete Fourier Transform (DFT) [9], Single Value Decomposition (SVD) [76, 152, 71], Discrete Wavelet Transform (DWT) [19, 68], Piece-wise Aggregate Approximation (PAA) [157, 74], and Adaptive Piece-wise Constant Approximation (APCA) [73]. Nevertheless, all these techniques appear to be unsuitable for trajectories, because interdependency among the two dimensions of space in moving object positions carries valuable information, and is typically lost in dimensionality reduction, when applied on the two independently [20].

For shape-based similarity search, an interesting approach to represent a time series using the direction of the sequence at regular time intervals is presented in [110]. In this work data is represented by a discrete set of alphabet. String-matching search, then, is utilized to find similar data sequences on the alphabet set, which in turn, is mapped to the original data. In 2000, Ge and Smyth [47] presented an alternative approach for searching shape similarity of two sequences based on probabilistic matching. Their approach addresses the issue of data uncertainty, not covered in any of the previous work, in shaped-based similarity search. Yanagisawa et al. also addressed shape-based similarity search for trajectories. In this work the similarity between trajectories is defined as the spatio-temporal distance between directed discrete lines [153]. However, their defined spatio-temporal

distance can only be used in the case where compared trajectories have identical sampling rates.

A recent work that proposes a method to cluster trajectory data is by Gaffney and Smyth [46]. Their method is a model-based approach that seems to suffer from a scalability problem. Also, it assumes a model of movement, which is not easy to determine in real data sets. By extending method presented in [139], Buzan et al. proposed an approach to find hierarchal clusters of similar trajectories in the video dataset in 3D. They propose to compute the distance between two time sequences as a pair of numbers, representing the similarity between the projections of the two time sequences on the coordinate axes [18].

6.1 Similarity notions

Similarity can be defined on the basis of information that is directly or indirectly available. As mentioned in Section 2.4, we work on moving object data in its simplest format, i.e., a trajectory p is of type *seq* ($\mathbb{T} \times \mathbb{L}$), and thus is a sequence of time-stamped positions (in $\langle time, location \rangle$ format). Since similarity can be defined with different parameters, it is important to first identify what information is available from the assumed moving object data. One immediately identifies *time* and *location* information, but this is not all. Since time increases continuously and linearly, there is an order on the time data, which translates into an *order* on the position data. More hidden meaning is present in the data, for instance, the first (last) data point identifies the *origin* (*destination*) of the object. With this location origin, *time origin* is another hidden piece of information, and both are important to consider in 'shifting' the trajectory (in time as well as in space). So far, from a sequence of time-stamped positions, we have identified five direct or indirect information sources, namely: location, time, order, time origin, and location origin. Different combinations of these five properties, as we will see below, result in the definition of different similarity notions.

One perspective on trajectory similarity is via the tree structure, illustrated in Figure 6.1. It represents similarity semantics that can possibly be defined on the basis of available information. Some notions are full-fledged spatio-temporal notions, while others are degenerate cases in which some information is *intentionally* lost.

Obviously, time data plays an important role in similarity tree classification and leads to a major split, namely, between *spatial*, *temporal*, and *spatio-temporal*

similarities.

Location-origin is one factor for further splitting in the tree. When we define similarity in a *location origin-independent* way, all recorded (time-stamped) positions are spatially shifted to the origin of the embedding space, and we are dealing with relative locations instead of absolute locations. This technique is useful, however, in applications that are not the focus of this thesis and, thus, we ignore them here in our further discussion. For similar reasons, we also ignore trajectories that are reduced to purely temporal data.

Time-origin is another important split factor in the spatio-temporal similarity category. In *time origin-dependent similarity*, object trajectories need to be synchronous to be considered similar, while in *time origin-independent* similarity, object trajectories may be asynchronous, and still be considered similar. A number of subcases is identified below.

6.2 Generic solution to partitioning trajectories by similarity

The overall idea to partition a trajectory set P of size N into k classes is derived from techniques to k-mean clustering in pattern recognition [116, 89]. Our algorithm takes P and k as arguments, and is called *generic_traj_pattern_classifier*. Observe that $P : set\ \mathbb{P}$.

The algorithm operates on the basis of starting with an arbitrary partition of trajectories, which is repeatedly adjusted until a halting condition is met. Each class in a partition is represented by a prototypical (i.e., characteristic) trajectory; this information is denoted by a function $proto\_traj : [1\,..\,k] \rightarrow \mathbb{P}$, such that $proto\_traj(j)$ is the prototypical trajectory for class j.

All the trajectories in the study set are assigned to a class, and this assignation is denoted by a function $in\_class : P \rightarrow [1\,..\,k]$, such that $in\_class(p)$ denotes the class to which a trajectory p in P has been assigned.

The start partition is a randomly chosen list of k distinct, actual trajectories randomly taken from the trajectory set P. This is done via the function *random_pick*, which we do not elaborate on here. (But see Section 6.5.6).

procedure *generic_traj_pattern_classifier*(P, k)
begin
  $proto\_traj[1..k] \leftarrow random\_pick(P, k)$

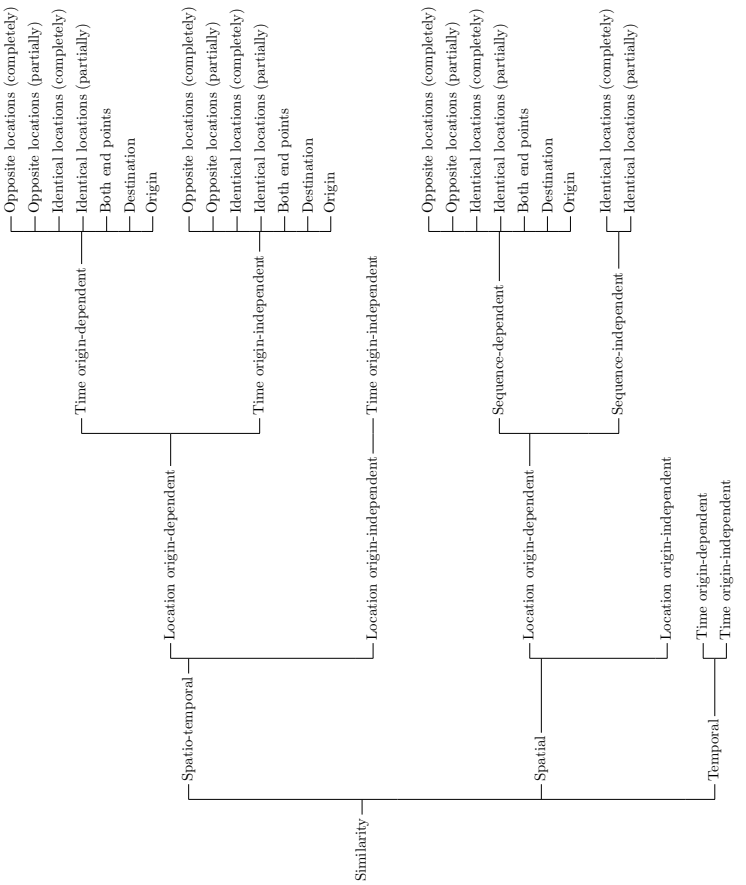


Figure 6.1: *A taxonomy of similarity notions in moving object trajectories.*

$\forall p \in P$:
  $in\_class(p) \leftarrow \mu\, j \in [1\,..\,k]$:
    $\mathbf{sim}(proto\_traj(j), p) = \min_{i \in [1..k]} \mathbf{sim}(proto\_traj(i), p)$
changes_made $\leftarrow$ true
while *changes_made*
do
  changes_made $\leftarrow$ false
  $\forall j \in [1\,..\,k] : proto\_traj(j) \leftarrow \mathbf{avg\_traj}(j)$
  $\forall p \in P$:
    $l \leftarrow \mu\, j \in [1\,..\,k] : \mathbf{sim}(proto\_traj(j), p) = \min_{i \in [1..k]} \mathbf{sim}(proto\_traj(i), p)$
    if $l \neq in\_class(p)$
    then
      $in\_class(p) \leftarrow l$
      changes_made $\leftarrow$ true
    fi
od
end

The generic algorithm cyclically determines a prototypical trajectory for a class as the average over the trajectories assigned to the class. It subsequently determines whether trajectories are still assigned to the same class. If not, re-assignation takes place and another cycle is executed.

The notation $\mu\, x \in X : P(x)$ stands for the unique value for variable $x \in X$ for which $P(x)$ holds. It is here applied to determine, for a given trajectory p, the unique class number that has the prototype trajectory closest to p. (In cases that multiple such indices exist, we arbitrarily pick one of these.) The expression uses a generic similarity function **sim** between two trajectories. It can be viewed as a generalized distance function, with $\mathbf{sim}(p, q) = 0$ denoting full similarity of trajectories p and q.

The second generic function in the algorithm is **avg_traj**. It is used to determine an average trajectory for a set of trajectories, here indicated by a class index.

Proper combined choices for **sim** and **avg_traj** allow the re-use of the algorithm for different definitions of trajectory similarity, hence, for different purposes. Several cases will be discussed below.

6.3 Similarity measures for trajectories

What we set out to do in this section is the development of an unbiased measure that expresses how similar two trajectories are. Such a measure will be an important tool, as we have seen in Section 6.2, to trajectory clustering, and thus, to trajectory grouping.

Similarity of trajectories falls into three different classes:

Temporal similarity in which trajectories are viewed as time intervals, indicating trajectory existence only;

Spatial similarity in which trajectories are viewed as directed multi-lines only;

Spatio-temporal similarity in which the temporal and spatial dimension are both considered when comparing trajectories.

All three classes are useful for trajectory analysis, and in usage contexts of exploration should probably be used in combination. Spatial similarity allows to test for likeness in shape; temporal similarity allows to test for synchronicity. In this section, we focus especially on spatio-temporal measures of similarity, for reasons similar to those that were discussed in Section 5.3.1.

Any of our similarity measures is essentially an extended average distance measure, expressing what is, on average over the time spanning both trajectory durations, the distance between the (moving) objects. This measure is an extension of the formula we derived in Section 5.3.2, specifically Formula 5.3. We first review and comment on that formula.

The formula defines $\alpha(p, a)$ as the average synchronous error between a(n original) trajectory p and a(n approximation) trajectory a. This case is somewhat specific, since by the approximation method involved, which only removes data points from p to obtain a eventually, we know that

- the two trajectories span the same time interval, and
- any time stamp in a also occurs in p.

These conditions are not met, obviously, for two *arbitrary* trajectories p and a that are otherwise unrelated.

Since we would like to use Formula 5.3 still, we need to scrutinize its use for more general purposes, and this is what this section attempts to do. We will in fact develop two such measures of similarity, one of which is based on time

interval intersection, and another that is based on time interval union. These two measures will address both problems mentioned above. They will be *full spatio-temporal measures*, as they respect the time stamps registered.

Another class of similarity measures that we will consider subsequently are degraded spatio-temporal measures, as we are intentionally shifting or scaling time axes with them.

6.3.1 Supporting trajectory operators

To describe an operator for temporal interval intersection or temporal interval union of two trajectories, we need to define six simple operators first.

Temporal projection The first is the *temporal projection* of a trajectory, denoted by π_t. This operator is of type $\mathbb{P} \rightarrow seq\ \mathbb{T}$. Its definition is:

$$\forall p : \mathbb{P} \mid \pi_t(p) = \big[p[1]_t, \ldots, p[len(p)]_t\big].$$

This states that the temporal projection of a trajectory p results in the (ordered) list of its time stamps.

Temporal interleaving The second operator is a *temporal interleaving* operator, denoted by ι_t. It is of type $seq\ \mathbb{T} \times seq\ \mathbb{T} \rightarrow seq\ \mathbb{T}$, and this infix operator is defined as follows:

$$\forall a, b : seq\ \mathbb{T} \mid a\ \iota_t\ b = c,$$

where c meets the following two conditions:

- $\text{ran}\ c = \text{ran}\ a \cup \text{ran}\ b$, and
- $\forall\, 0 < i < len(c) \ \big|\ c[i] < c[i+1]\,.$

The first condition states that all members of the lists a and b occur in the list c, and that c has no other elements. (A list can be conveniently viewed as a function from its index domain to its member domain; ran denotes the range of a function, so here it is the set of actual list members.) The second condition enforces the correct sequence amongst members of the list of time stamps.

Temporal enrichment The third operator in this discussion is the *temporal enrichment* of a trajectory, denoted as an infix operator ρ_t. It is of type $\mathbb{P} \times seq\ \mathbb{T} \rightarrow \mathbb{P}$, and operates on a trajectory p and a time stamp series a, and results in a new trajectory q, with the following characteristics:

- All original time-stamped positions of p are retained in q,
- For every time stamp t in a, a position $\mathrm{loc}(p,t)$ (on p) is determined with the method defined below, and the time-stamped position $(t, \mathrm{loc}(p,t))$ is included in q, in proper sequence. The position $\mathrm{loc}(p,t)$ is determined as follows:

$$\mathrm{loc}(p,t) = \begin{cases} p[1]_{\mathrm{loc}} & \text{if } t < p[1]_t, \\ p[i]_{\mathrm{loc}} + \frac{t - p[i]_t}{p[i+1]_t - p[i]_t}(p[i+1]_{\mathrm{loc}} - p[i]_{\mathrm{loc}}) & \text{if } \exists i | p[i]_t \leq t < p[i+1]_t, \\ p[len(p)]_{\mathrm{loc}} & \text{if } t \geq p[len(p)]_t, \end{cases}$$

- No duplicate time-stamped positions are included in q.

The effect of this definition is that $(p\ \rho_t\ a)$ is a proper trajectory that is a temporally densified version of trajectory p. Moreover, if the time range of a stretched beyond (before and/or after) that of p, the result is similarly stretched, conservatively in the position, meaning that the object is considered to be stationary in its start (respectively, end) position.

Temporal restriction The fourth operator here is the *temporal restriction*, or time slice, operator on trajectories. We denote it by τ_t; this operator is of type $\mathbb{P} \times \mathbb{T} \times \mathbb{T} \rightarrow \mathbb{P}$. Informally, its semantics is such that $\tau_t(p, t_1, t_2)$ is a trajectory identical to p within the time interval $[t_1, t_2]$, but with all other time-stamped locations lost. This typically means we obtain a trajectory with a reduced spatial extent. The result of this operator is potentially an empty trajectory, when the time interval falls outside of p's lifespan. The temporal restriction operator τ_t has the following characteristics, when invoked for $\tau_t(p, t_1, t_2)$:

- if $t_2 < t_1$, the result is undefined.
- if $t_2 < p[1]_t$ or $p[len(p)]_t < t_1$, the result is the empty trajectory [ ].
- otherwise, let $q = p\ \rho_t\ [t_1, t_2]$ and the result is

$$q\big[(\mu\ i : \mathbb{N} \mid q[i]_t = t_1),\ (\mu\ i : \mathbb{N} \mid q[i]_t = t_2)\big],$$

i.e., the subsequence of q over the indicated time slice. See our notational conventions in Section 2.4.1.

The above operators do not extend or restrict existing trajectories, leaving at least some of their data points intact. The two operators below affect the time axis of their trajectory argument, and change the time stamps of all the trajectory's data points.

Temporal shifting Another operator useful for our discussion is the time shift operator δ_t. When used as $p\ \delta_t\ d$, it expresses a trajectory obtained from trajectory p by adding a constant time lag d to all of p's time stamps. Its definition is simple:

$$\forall p : \mathbb{P} \ \mid\ p\,\delta_t\,d = [\ (p[1]_t + d, p[1]_{\text{loc}}), \ldots, (p[len(p)]_t + d, p[len(p)]_{\text{loc}})\ ].$$

Temporal scaling A final operator on trajectories is one that allows to scale the underlying time axis. When we write $p\phi_t c$, we denote a trajectory obtained from p by scaling with constant factor c the temporal distances between p's time stamps. The definition of this operator uses p's first time stamp as temporal origin:

$$\begin{aligned}\forall p : \mathbb{P} \ \mid\ p\,\phi_t\,d = [\ & (p[1]_t + c(p[1]_t - p[1]_t), p[1]_{\text{loc}}),\\ & (p[1]_t + c(p[2]_t - p[1]_t), p[2]_{\text{loc}}), \ldots,\\ & (p[1]_t + c(p[len(p)]_t - p[1]_t), p[len(p)]_{\text{loc}})\ ].\end{aligned}$$

6.3.2 Interval intersection trajectory similarity

One perspective on comparing trajectories is to look at the time interval during which both objects' positions have been time-stamped, and compare them over that interval. This leads us to the *interval intersection trajectory similarity* (*iits*) of two trajectories p and q.

We define $iits(p, q)$ as the average (spatio-temporal) distance of the objects, over the period of their co-existence. If the two did not co-exist at all, we pick a prohibitively high value K for $iits(p, q)$, indicating that the two trajectories are entirely dissimilar, under this measure.

In the definition, we use the following short-hands:

$$\begin{aligned} s &= \max\big(p[1]_t, q[1]_t\big) \\ e &= \min\big(p[\mathrm{len}(p)]_t, q[\mathrm{len}(q)]_t\big) \\ p' &= \tau_t(p, s, e) \\ q' &= \tau_t(q, s, e) \end{aligned}$$

Here, s and e are defined such that $[s, e]$ is the time interval of co-existence, if it exists at all. The trajectories p' and q' are versions of p and q, temporally restricted to the interval.

We can now define *iits* as follows:

$$\forall p, q : \mathbb{P} \mid \mathit{iits}(p, q) = \begin{cases} K & \text{if } s > e \\ \alpha(p' \, \rho_t \, \pi_t(q'), q') & \text{otherwise} \end{cases}$$

The function α is as given by Equation 5.3. The similarity measure is defined to equal the average (spatio-temporal) distance during the interval of co-existence. The attentive reader may observe that q' as the second argument to α above is slightly unexpected: for reasons of symmetry, one may have expected to see $q' \, \rho_t \, \pi_t(p')$ as the second argument. This would actually have been correct as well, but since α is not a symmetric function, it suffices to temporally enrich only the left hand argument (with time stamps from the right hand argument). A simple and a realistic semantics of interval intersection of two trajectories are illustrated in Figures 6.2 and Figures 6.3.

6.3.3 Interval union trajectory similarity

Another perspective on trajectory similarity compares two trajectories p and q for the (smallest) time interval that covers both trajectories existence intervals. This will lead us here to the similarity notion of *interval union trajectory similarity*, denoted as *iuts* of p and q.

To obtain an appropriate definition, we need to make assumptions on the whereabouts of an object outside of its trajectory's interval of existence. We will assume conservatively that an object is at its starting position any time before its first time stamp, and analogously, that it is at its final position any time after its last time stamp.

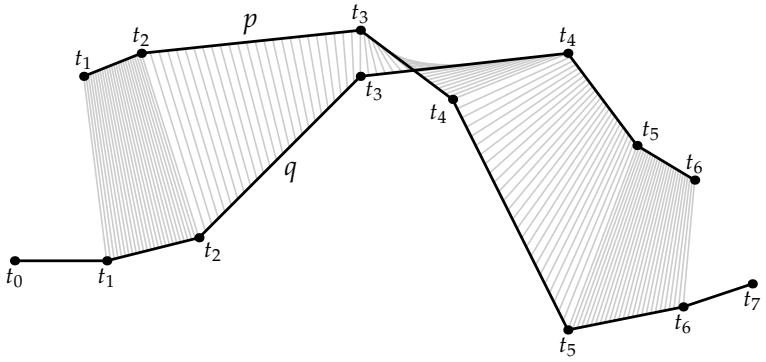


Figure 6.2: *Interval intersection semantics for trajectories p and q. The first has as interval $[t_1, t_7]$, the second $[t_0, t_6]$. The similarity measure $iits(p, q)$ is therefore determined for the interval $[t_1, t_6]$. Intuitively, it equals the average length of the distance chords, of which a number is also shown. This is a simple, atypical, example as within the intersection interval, p and q's time stamps are identical. Time stamps t_i and t_{i+1} are always one time unit apart; distance chords have been drawn at 50 per time unit, so their density is indicative of relative object speed (denser means slower).*

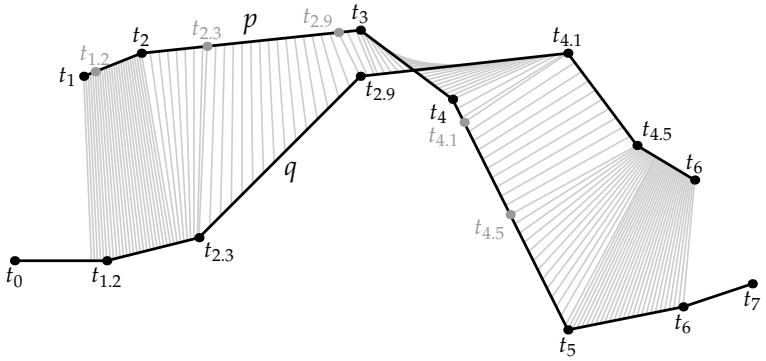


Figure 6.3: *A more realistic example of interval intersection semantics for trajectories p and q, in which many of q's time stamps do not coincide with those of p. Along the p trajectory are also indicated in grey the time stamps from q with which p has been enriched, according to the definition of interval intersection semantics.*

We can proceed as before, and use the following short-hands:

$$\begin{aligned} s &= \min\big(p[1]_t, q[1]_t\big) \\ e &= \max\big(p[\mathrm{len}(p)]_t, q[\mathrm{len}(q)]_t\big) \\ q' &= q\ \rho_t\ [s,e] \end{aligned}$$

Here, s and e are defined such that $[s,e]$ is the minimal time interval that covers both trajectories' existence. The trajectory q' is a version of q, temporally enriched by this interval. There is no need for p' in the case of *iuts*.

We can now define *iuts* as follows:

$$\forall p, q : \mathbb{IP} \mid \mathit{iuts}(p,q) = \alpha\big(p\ \rho_t\ \pi_t(q'), q'\big)$$

Both *iits* and *iuts* measures are full spatio-temporal similarity measures for trajectory comparison. They respect the time-stamped positions completely. In the next sections, we look at a number of measures in which we allow ourselves to discard some (temporal) information. A simple and a realistic semantics of interval union of two trajectories are illustrated in Figures 6.4 and Figures 6.5.

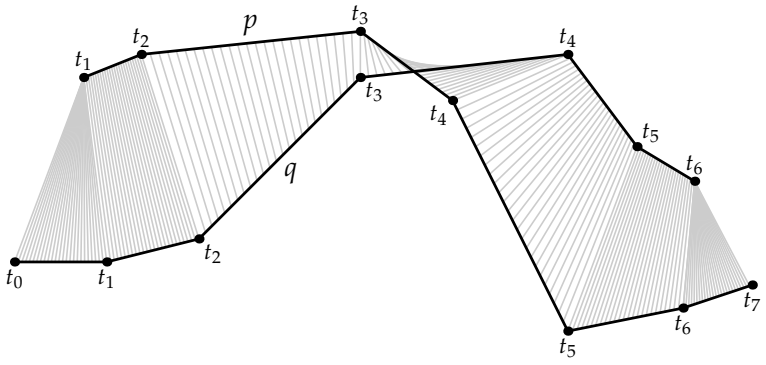


Figure 6.4: *Interval union semantics for the trajectories p and q of Figure 6.2. Trajectory p has as interval $[t_1, t_7]$, q has $[t_0, t_6]$. The similarity measure $\mathit{iuts}(p,q)$ is therefore determined for the interval $[t_0, t_7]$. This measure differs from iits in that the the object is considered to be stationary at start (end) point before (after) the time interval of its trajectory, and that position is used in the measure as well. In this example, the difference between iits and iuts is only slight, because there is substantial temporal overlap between p and q.*

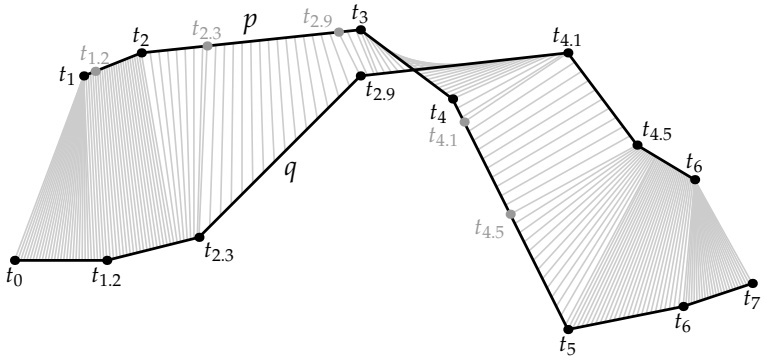


Figure 6.5: *A more realistic example of interval union semantics for trajectories p and q of Figure 6.3.*

6.3.4 Time-shifted trajectory similarity

There exist applications of trajectory comparison in which the absolute values of time stamps are less important but the, relative, travel time is still relevant. One such class of applications will attempt to group trajectories with similar spatial extent, similar duration, but different absolute timing, allowing to identify objects that show the same movement, modulo a somewhat constant time difference. For such cases, one may want to compare two trajectories by first optimizing in one way or other on their temporal overlap, followed by a search for similarity. In combination, we call these techniques time-shifted similarity computations, as they imply a shift along the time axis of at least one, and usually one, of the two trajectories.

Various approaches of time shifting seem to be possible; we mention five of them, of which we define three constructively.

Start-shifted trajectory similarity In start-shifted similarity (*ssts*), we map the start time of one moving object q onto that of the other, p, and correct for this shift in all time stamps of q. This allows us to pretend the two objects started moving at the same time, and compare their trajectories on the basis of that premise. This technique is appropriate for searches of moving objects with similar origins.

The technique can take two definitions, *ssts_ii* and *ssts_iu*, depending on whether one wants to use interval intersection or interval union semantics to the time-shifted trajectories. For *ssts_ii* similarity, the definition is :

$$\forall p, q : \mathbb{P} \mid ssts\_ii(p,q) = iits\big(p, q\ \delta_t\ (p[1]_t - q[1]_t)\big)$$

Observe that due to the time shift provided, one will not end up in the case of *ssts_ii* similarity equalling big K. The use of this measure leads to comparisons of the two trajectories only for the interval of the trajectory with the shorter duration.

Alternatively, one may opt for using interval union semantics, leading to *ssts_iu*. Its definition is entirely analogous to the one given above for *ssts_ii*, with *iits* being replaced by *iuts*. The use of the *ssts_iu* measure leads to comparisons of the two trajectories for the interval of the trajectory with the longer duration. Figures 6.6 and 6.7 illustrate the semantic of start-shifted similarities for two trajectories in interval intersection and interval union, respectively.

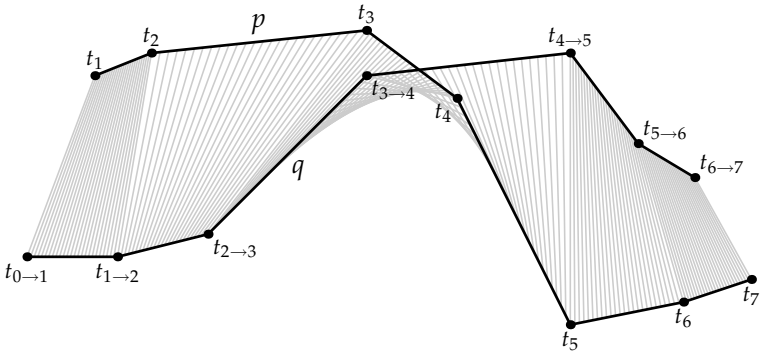


Figure 6.6: *Start-shifted comparison of the trajectories p and q of Figure 6.2. In this case, the interval and union semantics coincide, as p and q have identical interval durations.*

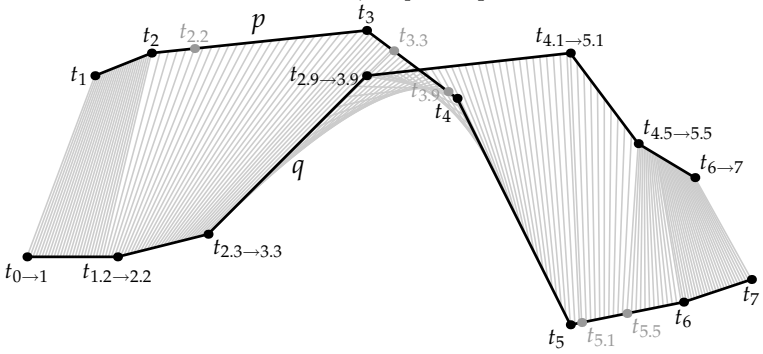


Figure 6.7: *Start-shifted comparison of the trajectories p and q of Figure 6.3.*

End-shifted trajectory similarity There is little new under the sun with this measure, as it is so akin to start-shifted similarity. Its application, however, is somewhat different as it emphasizes more the similarity in arrival at destination.

Again, we can apply interval intersection or interval union semantics. We provide here the definition of the first, being *ests_ii*, and leave the latter, *ests_iu*, for the reader.

$$\forall p, q : \mathbb{IP} \mid ests\_ii(p,q) = iits\big(p, q\, \delta_t\, (p[len(p)]_t - q[len(q)]_t)\big)$$

Mid-shifted trajectory similarity Although applications of this measure are slightly less easy to perceive, one might also consider a temporal shift of one of the trajectories' midpoints to that of the other. The technique is essentially identical to those introduced above, but now the formula for time lag d is more involved. To shift q's midpoint to that of p, one should set d to

$$\frac{p[1]_t + p[len(p)]_t - q[1]_t - q[len(q)]_t}{2}.$$

As before, interval intersection or interval union semantics can be applied.

Cycle-shifted trajectory similarity In situations in which regular, periodic movements by objects are suspected, one may want to perform time shifting of a cyclic nature. An obvious example is the analysis of commuter traffic, the data of which is accumulated over multiple days. The formulas for cycle-shifted trajectory analysis are easily written up, using the mod operator on the time domain. They do, however, rely on an a priori assumption of the cycle duration, and they do not allow to identify the cycle duration itself, other than by means of trial-and-error.

Cycle-shifting can be combined with other similarity techniques such as interval intersection and interval union semantics.

Optimally shifted trajectory similarity A final, both interesting and challenging, idea for temporal shifting, is to determine the time lag d for shifting q in such a way that the similarity between p and the optimally time-shifted q is maximal. At present, we do not know of a computational technique that would determine d unequivocally. No obvious brute-force technique is readily available either, and we haven't invested in heuristics-based techniques yet.

Time-shifted similarity looks for object trajectories that are similar in space and in time duration, but are allowed to be dissimilar in absolute clock time.

Above, we have defined a number of measures that do just this. Below, we relax one more characteristic of moving object trajectories, namely time duration. The resulting measure is still spatio-temporal, though of a rather liberal kind.

6.3.5 Time-scaled trajectory similarity

In *time-scaled trajectory similarity*, which in our notation is denoted by *tsts*, we look at trajectories even more liberally, and discard, possibly systematic, average speed differences. This can be achieved in various ways, but a choice between them does not affect the outcome of the measure we define here. Our technique first scales q's time axis so that the resulting trajectory has the same duration as that of p, and subsequently start-shifts that trajectory to trajectory p. Based on the above information, the measure is defined as:

$$\forall p, q : \mathbb{P} \mid tsts(p,q) = ssts\_ii\big(p, q\ \phi_t\ \tfrac{p[len(p)]_t - p[1]_t}{q[len(q)]_t - q[1]_t}\big)$$

This version of the definition uses interval intersection semantics. Reasoning by analogy, one would think that a version using union semantics would also exist. It does indeed, but its semantics is identical to that of the version above. The reason is that time scaling removes the difference in trajectory duration. For similar reasons, we need not look at definition versions with end-shifted trajectory measures.

6.4 Averaging over trajectories

In Section 6.2, we discussed a generic algorithm for partitioning a set of trajectories on the basis of similarity. It is generic because it makes use of two generic (unspecified) functions: **sim** and **avg_traj**. In the long Section 6.3, we provided various possibilities to instantiate the generic function **sim**.

In this section, we are providing possible instantiations of the other generic function, **avg_traj**, the function that determines an average trajectory for a collection of trajectories. In the algorithm of Section 6.2, this function had a single natural number j as argument, which indicated a class, hence a set, of trajectories. Without loss of generality, we will assume in this section that the single argument of any instantiation of **avg_traj** is in fact a set of trajectories P. Observe that $P : set\ \mathbb{P}$.

Below, we look at a rigorous approach to trajectory averaging. Much of this section intentionally has an organization that mirrors that of Section 6.3. The rea-

son is that the similarity measures defined there must be paired with appropriate averaging techniques; hence the one-one correspondence.

6.5 Averaging with computational rigour

The problem we address in this section is the computation of an 'average trajectory' from a given set P of trajectories. In line with the discussions above, there exist two natural techniques to do this. They are natural because they fit with our two temporal interval semantics: intersection and union.

A technicality of averaging over sets is always that its members cannot be identical. One may opt to go one of two ways: instead of working with sets to work with lists, or to assume that set members are distinguished by some identifier. This is essentially almost the same idea. In the paragraphs below, we assume set members are distinguished.

6.5.1 Some additional notation

To define our operators of trajectory averaging properly, we require a few more notational conventions and definitions. They are in fact generalized versions of the support operators of Section 6.3.1.

Generalized temporal projection Where $\pi_t(p)$ is the temporal projection of trajectory p to its list of time stamps, if P is a set of trajectories, we denote by $\pi_t(P)$ the set of lists of time stamps obtained from members of P:

$$\forall P : set\ \mathbb{P} \mid \pi_t(P) = \left\{\ \pi_t(p)\ \middle|\ p \in P\ \right\}.$$

Generalized temporal interleaving Where $a\ \iota_t\ b$ denotes the temporal interleaving of time stamp sequences a and b, its generalized version ι_t is of type $set\ seq\ \mathbb{T} \rightarrow seq\ \mathbb{T}$, and is defined as follows:

$$\forall A : set\ seq\ \mathbb{T} \mid\ \iota_t\ (A) = c,$$

where c meets the following two conditions:

- $\text{ran}\ c = \cup_{a \in A}\ \text{ran}\ a$, and
- $\forall\, 0 < i < len(c)\ \middle|\ c[i] < c[i+1]$.

Generalized temporal restriction We have seen that $\tau_t(p, s, e)$ is of type $\mathbb{P}$ and that it denotes the temporal restriction of trajectory p to time interval $[s, e]$. Its definition involves temporal enrichment with $[s, e]$, if those time instants were not already present in p.

We can generalize temporal restriction to a set of trajectories P in a straightforward way:

$$\tau_t(P, s, e) = \{\ \tau_t(p_i, s, e)\ \mid\ p_i \in P\ \}$$

Centroid For positional averaging, we also require a function

$$\gamma : (set\ \mathbb{P}) \times \mathbb{T} \to \mathbb{L}$$

that determines the centroid from a set of paths and a time stamp. It is defined as follows:

$$\forall P : set\ \mathbb{P}, t : \mathbb{T}\ \mid\ \gamma(P, t) = \left(avg_{(p \in P)}\, \mathrm{loc}(p, t).x, avg_{(p \in P)}\, \mathrm{loc}(p, t).y\right).$$

This is a partial definition only, since $\mathrm{loc}(p, t)$ is not always defined, but we will ensure that it is applied only in cases defined.

6.5.2 Averaging over interval intersections

The principal idea of interval intersection averaging over a set of trajectories $P = \{\ p_i\ \}$ is to determine an average trajectory only for the temporal interval in which *all* trajectories p_i co-exist. That interval is easily determined as $[s, e]$, where

$$\begin{aligned} s &= \max_{p_i \in P} \left(p_i[1]_t\right), \\ e &= \min_{p_i \in P} \left(p_i[\mathrm{len}(p_i)]_t\right), \\ P' &= \tau_t(P, s, e), \text{ and} \\ I &= \iota_t(\pi_t(P')). \end{aligned}$$

We are including in this short-hand list also P', which is the trajectory set obtained from p by generalized temporal restriction to $[s, e]$, and I, which is the interleaving of all time stamps obtained from trajectories in P'.

Assuming for now that this $[s, e]$ is a valid temporal interval. The averaging operator is *avg_ii* and it is of type $set\ \mathbb{P} \to \mathbb{P}$. Its definition is:

$$\forall P : set\ \mathbb{P} \mid avg\_ii(P) = \left[\ \big(I[1], \gamma(P', I[1])\big), \ldots, \big(I[len(I)], \gamma(P', I[len(I)])\big)\ \right].$$

Some comments are in order here. The resulting average trajectory has as many time-stamped positions as there are time stamps in the interleaving sequence I, thus, in the trajectories of P after they have been restricted to the intersection interval. For each time stamp u in the interleaving, the formula dictates to determine for each trajectory $p \in P'$ its position at that time, which are then averaged over. Averaging with interval intersection semantics is illustrated in Figure 6.8.

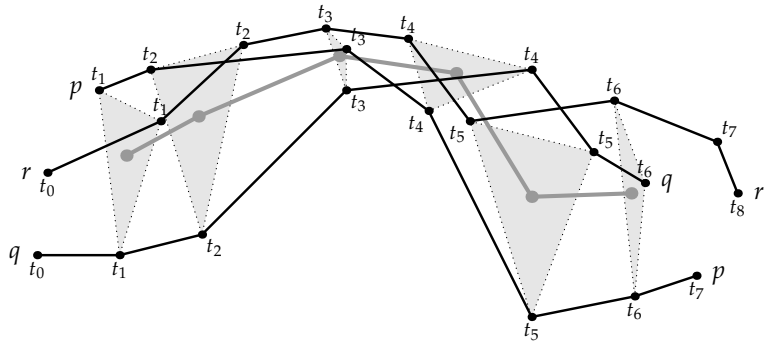


Figure 6.8: *Averaging with interval intersection semantics. Here, three trajectories p, q and r, respectively on the intervals* $[t_1, t_7]$, $[t_0, t_6]$ *and* $[t_0, t_8]$, *are averaged over. The resulting average trajectory is shown in grey: grey dots indicate average positions for* t_1 *up to* t_6.

6.5.3 Averaging over interval unions

There is substantial analogy if one wants to perform an average over the generalized union of the intervals of a set of trajectories, instead of over their generalized intersection. We define

$$\begin{aligned} s &= \min_{p_i \in P} \big(p_i[1]_t\big), \text{ and} \\ e &= \max_{p_i \in P} \big(p_i[\mathrm{len}(p_i)]_t\big), \end{aligned}$$

and can leave the other parts of the definition of *avg_ii* intact to obtain the definition of *avg_iu*. Averaging with interval union semantics is illustrated in Figure 6.9.

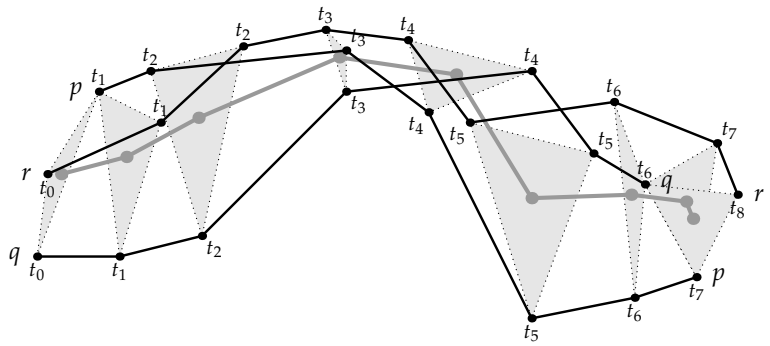


Figure 6.9: *Averaging with interval union semantics. Here, three trajectories p, q and r, respectively on the intervals $[t_1, t_7]$, $[t_0, t_6]$ and $[t_0, t_8]$, are averaged over. The resulting average trajectory is shown in grey: grey dots indicate average positions for t_0 up to t_8.*

6.5.4 Averaging over time-shifted trajectories

Time-shifted trajectory averaging, like the related similarity measures, can be based on interval intersection or interval union semantics. This is true for start-shifted and end-shifted techniques, as well as any other. A technicality that arises with time shifting, is that in the (binary) similarity measure, we typically shifted the second trajectory to the first. In averaging, we operate on sets of trajectories, so there is no 'first trajectory' or chosen trajectory to shift to. Luckily, the semantics of averaging and time shifting are such that it does not matter where we shift to, as long as we shift all trajectories in the same way. We will systematically start- and end-shift to $t = 0$.

Start-shifting The definitions of trajectory averaging, *avg_ss_ii* and *avg_ss_iu* respectively, are simple applications of the functions defined above:

$$\forall P : set\ \mathbb{P} \mid avg\_ss\_ii(P) = avg\_ii\big(\{\ p\ \delta_t\ (-p[1]_t) \mid p \in P\ \}\big)\,,$$

and for the definition of *avg_ss_iu* the used function *avg_ii* is simply replaced by *avg_iu*.

End-shifting Again, there is pure analogy here, as we shift each trajectory now by its last time stamp, not its first:

$$\forall P : set\ \mathbb{P} \mid avg\_es\_ii(P) = avg\_ii\big(\{\ p\ \delta_t\ (-p[len(p)]_t) \mid p \in P\ \}\big)\,.$$

The definition of *avg_es_iu* again uses *avg_iu*.

6.5.5 Averaging over time-scaled trajectories

Recall that time-scaling effectively results in start-shifting and temporally scaling the involved trajectories to the same interval duration. We already covered the issue of start-shifting a complete set of trajectories to a common temporal origin. An equivalent action must be taken for obtaining a common interval duration. Luckily, the semantics of scaling, similarity and averaging are such that the duration of choice (say C) is immaterial for the average trajectory determined, modulo its interval duration, which will equal C, obviously.

This leads us to the following:

$$\boxed{\forall P : set\ \mathbb{P} \ \mid \ avg\_ts(P) = avg\_ii\big(\{\ (p\phi_t \tfrac{C}{p[len(p)]_t - p[1]_t})\delta_t(-p[1]_t) \mid p \in P\ \}\big)}$$

The definition is based on an arbitrary choice for constant C. Time-scaled trajectory averaging involves, for each trajectory p in the argument set P, a scaling of p's duration to C, achieved by the ϕ_t operator above, and then a time shift to the temporal origin, achieved by δ_t.

6.5.6 Discussion of applicability

We discuss applications of the various measures and averaging techniques in Section 6.7. But some comments can be made at this stage from an analysis of the developed techniques and formulas.

The measures and functions above were developed from a perspective of supporting the analysis of large sets of moving object trajectories. Such applications typically do not pose real-time requirements, and this is appropriate, given their computational complexity. It is expected that the usefulness of the developed techniques will be most prominent in off-line analytic situations.

A thorough analysis of computational complexity of the algorithms involved was not carried out. It is well-known from the field of data mining that the popular k-means clustering technique, of which our algorithm *generic_traj_pattern_classifier* on page 121 is a special form, has a polynomial complexity $O(Nkai)$, where N is the number of objects to be partitioned, k is the number of clusters to be identified, a is the number of object attributes to be considered, and i is the number of iterations needed for the algorithm to converge [15].

The problem with moving object clustering is that a is variable: our trajectories have a variable number of (sampling) data points, each of which has three characteristics: time stamp and position. Thus, a is invariably high, because the trajectory sequence will typically be long. In fact, the factor a is most prominent

in the definitions of our similarity measures and averaging functions. The time complexity of the latter require deeper study, but an initial, conservative estimate is that most of our support operators are $O(a)$—temporal restriction appears to be implementable in $O(\log a)$—and our similarity measures appear to be typically $O(a^2)$ because of the application of function α. The averaging functions appear to be $O(Na^2)$.

Another factor of influence is the number of iterations required to reach convergence. It is not known a priori; some approaches have tried to optimize it by proper bootstrapping [16]. We have not looked into such techniques, but it is obvious that spatial indices may help to dramatically improve bootstrap values, as they will allow a proper spatial spread of the initial trajectories. Depending on combined choice of similarity measure and averaging algorithm, different spatial spreads are required.

A different and fundamental problem, which arises from our definitions, hides in the partiality of our function (*avg_ii*, see Section 6.5.2) for averaging trajectories with interval intersection semantics. The temporal interval $[s, e]$ may not exist, and thus, an average trajectory may be impossible to determine. This is particularly awkward in the unconstrained case in which we do not have a priori knowledge about the set of trajectories that is the argument to the function. In more constrained cases, like application of the function on a set of time-shifted or time-scaled trajectories, the problem is decidedly less grave, as the interval will exist in such cases.

Caution must be taken therefore to apply *avg_ii* on unconstrained trajectory sets. A computational solution, though more a fix, would be to define the result of the application of the function, in case of a non-existing intersection interval, to be a degenerate single data point trajectory that is spatially remote of any data point included in the analysed trajectories. However, such a 'solution' seems to be potentially detrimental to the overall convergence of our generic algorithm. Indeed, one obvious case where application of *avg_ii* in *generic_traj_pattern_classifier* will cause problems is if the trajectory set p does not contain at least k trajectories that in combination span the complete time range of all trajectories combined. A proper assignation of all trajectories to k classes is then unsatisfiable.

6.6 Optimization issues

Experiments with the above algorithmics are underway at the moment of writing, and will be used for a forthcoming paper. They are based on a simulation application for urban traffic, which generates moving object trajectory test data. These experiments are meant to improve our understanding of the practical consequences of the techniques proposed here.

In the near future, attention will be paid to optimizing the techniques, especially when the data is being operated on in a (spatial) database context. We believe that the requirements for proper trajectory support on a SDBMS, in the light of this chapter, are (a) low level support for sequence operators, and (b) support for higher-dimensional spatial indexing, specifically applicable to trajectories. The first is needed to implement our support operators, which are heavily used in the functions above. The second may help in filtering trajectories for the class assignation step of our generic algorithm: this may dramatically improve the clustering technique.

The computationally rigourous approach that is proposed above may lead to solutions with high computational complexity. We have also considered the application of heuristics to:

- bootstrapping initial trajectories (see the *random_pick* operator in our overall algorithm,
- (given a trajectory) determining the most similar trajectory amongst k known trajectories, by using spatiotemporal bounding box characterizations, and
- faster determination of a representative (average) trajectory for a class of trajectories, involving directed hulls.

The general problem with such approaches is that the introduction of heuristic rules may lead to a loss of convergence of the overall algorithm.

6.7 Applications

The ability to satisfactorily partition trajectories take from a large set is beneficial to numerous applications. Throughout our work, we have had urban traffic as an important example and focus. Here we briefly discuss how each of our similarity measures, defined earlier in Section 6.3, can find application in this setting. It needs to emphasized, however, that moving objects are not only agents in urban

traffic, and that other applications of the techniques here proposed do exist. One may think of analysis of migration patterns in wild animals, shopping behaviour in supermarkets, and other locations such as airports where people congregate.

Interval intersection trajectory similarity Interval intersection semantics emphasize 'co-existence in co-presence' and allows to identify objects that shared part of their path synchronously for part of their travel time.

Traffic bottleneck analysis and *rush hour analysis* are obvious applications of this similarity measure, since people who take similar paths at the same time are likely facing traffic jams. *Trend analysis* can also be mentioned as another application of this similarity measure.

Interval union trajectory similarity Interval union semantics take a more complete temporal view of trajectories, and are thus more suited to be applied in analysis cases looking for overall trajectory similarity. It will allow to conduct origin/destination studies more properly.

Commuter traffic analysis, specifically distinguishing day-shift commuters from night-shift commuters, is an application of this similarity measure.

Time-shifted trajectory similarity With time shifts, we relax the condition of synchronicity of the objects involved. As a consequence, we can aggregate traffic over time, and this leads to proper support of studying the use of space by traffic categories. Time shifts without time scaling still do justice to object categories with different average speeds, allowing for instance to tell the cars from the pedestrians. Start-shifted and end-shifted similarity have different semantics with obvious application differences.

Transportation means analysis is an obvious application for this similarity measure, as it aims at identifying how people travel and distinguishes between various transportation means, i.e., bicycle, car, or bus. We can also mention an application for *car pooling* analysis here.

Time-scaled trajectory similarity When we apply time scaling, we are implicitly using time shifting as well, and so the arguments of the previous section apply here. But with time scaling, we are removing any factor of average speed, and thus we aggregate at a more coarse grain level. By example, we will no longer distinguish cars from pedestrians: their difference in speed is no longer an obstacle for co-clustering. The main emphasis is on the list of visited locations.

Sharing transportation means is an example application that will benefit from this type of similarity measure. One interest in this application is clearly on identifying people who can provide a ride to another. Various scenarios can be defined. *General traffic flow analysis* is another application of this type of similarity measure.

6.8 Lessons learnt and plans ahead

Trajectories may be similar for all of their extent or only for a part of it. They may also approximately coincide in start and end points, yet be largely dissimilar internally; they may be largely identical spatially but not temporally. Therefore, a single notion of similarity does not suffice, and that we need to select the notion with various purposes in mind.

In this chapter various similarity notions were defined, which in turn were used for classification and partitioning trajectories. To achieve this, a generic solution was proposed.

Defining an average trajectory is an important requirement for the generic solution to be fully operational. Therefore, averaging with computational rigour was discussed.

Various optimization issues to enhance the performance of the generic solution, as well as the applicability of the solution were explained.

Chapter 7

What was done and what to do next

It is wiser to find out than to suppose.

Mark Twain (1835 – 1910)

Thanks to recent advances in positioning technologies, which have made small, cheap, and reasonably accurate receivers publicly available, it has become possible to collect data about whereabouts of moving objects easily, i.e., objects whose positions continuously change. Availability of data together with performance enhancement of computing devices, growth of the Internet and global wireless network infrastructure, and rapid advances in hardware technology of personal mobile devices have introduced new scenarios in the GIS and database field collectively called mobility.

Despite of all the previous work on GIS and databases, situations in which the whereabouts of objects are constantly monitored and stored for future analysis are an important class of problems that present-day database users will find hard to tackle satisfactorily with their systems. This stems from the fact that the current databases are at their best good in handling static situations, a property which definitely is not met by the mobility concept. The concept of mobility brings whole new sets of requirements, challenges, and applications; its potential is endless.

7.1 Brief summary of this thesis

Continuous progress in science and technology has made existence of mobility scenarios possible and has facilitated proving its related services. The applications that benefit from the mobility concept are manifold. However, there is a long way to go to actually design and implement these applications, due to a variety of challenges involved. This necessitates research in the field of moving objects.

Chapter 1 describes the motivations behind this research and discusses important problems of the moving object world. The chapter also explains that in addition to general problematic issues of moving objects that hold for all related applications, there are some other issues that are application-dependent. Obviously, the more problems are solved provides the more successful applications can be built. It is, however, not possible to address all the identified difficulties of the moving object domain in a single work. Therefore, the chapter reports on four fundamental issues, namely, handling uncertainty for moving object data, faithful trajectory representation, trajectory compression techniques, and similarity measures for trajectories, as central items to be addressed.

The evolution of data models that incorporate space and time are described in Chapter 2. It provides a time line on how various attempts in modeling space and time independently failed to model continuous change and how long it took to be realized that these two concepts should be integrated and corporate in one single data model. The capabilities and problems of each integrated data model are pointed out as well. Also, the chapter gives an overview of various data acquisition methods and techniques used to collect data about moving objects. The importance of this discussion is that each of these methods provides both different data types and data accuracy. A proper understanding about moving object application and its requirements is a must in selecting the right data acquisition methods and devices. The rest of the chapter sets up the framework, in which the research is conducted by explaining fundamental assumptions that were made, tools (i.e., data model and data acquisition methods) that were utilized, and like wise, notations.

Chapter 3 addresses data quality and accuracy of the data that is required in moving object applications. Although in any application uncertainty may arise from the very first step of data modelling, and remain until the last operation of the decision-making process, most erroneous results are due to error in the raw data. Furthermore, since errors in raw data can be substantial and that they may be propagated by further processes, it should be reduced to the best possible extent. Having satellite-based positioning in mind, as the data acquisition technology, the chapter describes the currently available strategies to enhance accuracy of the data. Although these techniques effectively reduce the error, result may still not still be satisfactory for some applications. Car and airplane navigation applications are just two applications here. As the chapter explains, at the best accuracy offered by satellite-based positioning systems at present may result in mapping the data to

the wrong side of the road, or even not on the road at all. When technology fails to further improve of the data, more intelligent techniques are needed to achieve this. The chapter reports on our proposed snapping technique, which works through the integration of measurements with additional spatial information contained in the GIS. The idea behind this techniques is to replace each location data obtained from a receiver with its 'most probable' location on the network. Distance from the network and accessability of network segments visited prove to be the key to successfully snap the registered data to the underlying network. The experimental results show significant reduction of error associated with raw data.

Objects movement is a continuous phenomenon. However, the corresponding data is acquired in discrete ways. Therefore, the problem of unavailability of data between two consecutive registered data points exists. Since most, if not all, moving object applications work around central questions of *where*, *when*, and *what*, which involve location, time and object, respectively, to answer these questions a faithful determination of moving object trajectories is a requirement. To do so, linear and spline interpolation, the two most well-known interpolation techniques, have often been used. However, as Chapter 4 points out, as far as realistic representations of moving objects are concerned, these methods have failed to perform well. By analyzing the behaviour of these techniques, the chapter reports on the identified problems. It finds that the common problem in current techniques is to use a single function to describe the complete movement, while faithful representation requires a clear distinction between different patterns of movement (states of the trajectory). Furthermore, movement in each of these states should be described by a different function which best suites that portion of the object's movement. To achieve this, the chapter introduces the concept of 'break points' as points along the trajectory in which an abrupt change in movement occurs. Consequently, a method for extracting break points based on data sequence grouping is presented. Data points that prove to be similar in speed or acceleration are added to a defined sequence. The chapter pays special attention to the fact that regardless of how much error reduction has taken place, data may still be erroneous. Therefore, the definition of similarity of speed and acceleration is based on uncertainty in the data, which in turn is defined using error propagation laws. On the other hand, the chapter explains that applications dealing with network-constrained objects have frequently made use of pure interpolation techniques, without paying attention to the underlying network, which has had direct effects on where object can go. By identifying all these elements and showing the incapabilities of current

methods, the ultimate goal of this chapter is an alternative approach to overcome the mentioned problems. Experimental results do not only back up the claim of a more realistic representation of object trajectories, but also provide useful ideas for compression techniques of network-constrained moving object trajectories.

Chapter 5 brings the attention to the fact that large quantities of moving objects that are equipped with position-enabled devices will soon start to generate huge volumes of data. Storing all this data is neither possible, nor wise. Various complications exist for storage, transmission, indexing, and query processing. One solution to the problem is using compression techniques to discard not so informative data points, while preserving the major characteristics of the original trajectory. The chapter explains why existing compression techniques commonly used for line generalization in the GIS domain or for time-series data in the data mining field are not suitable for compressing moving object trajectories. They are good for short one-dimensional time series and in absence of noise; but these are properties not valid for moving objects. On the other hand, the error notion used to decide on keeping data points in current compression methods works mainly on the basis of distances between registered data points and compressed representation of the trajectory. However, as the chapter explains, this is not appropriate. Due to the continuous nature of object movement, both spatial and temporal factors should be taken into account in compressing moving object data. Our proposed compression techniques successfully achieve the following goals *(i)* obtaining a lasting reduction in data size, *(ii)* obtaining a data series that still allows various computations at acceptable (low) complexity, and *(iii)* obtaining a data series with known, small margins of error, which are parametrically adjustable. The chapter also reports on a new error notion based on distance between each original data point and its representation on the compressed trajectory. While compression techniques for unconstrained moving objects are applied on data points that are originally present, another strategy is used for compressing network-constrained moving object data based on inventing new data points to add to the sequence. The reported experimental results back up the proposed approaches.

Moving objects often follow the same routine at particular times, e.g. commuters, clients in shopping malls. Chapter 6 reports on methods to discover these repeated patterns. The significance of such discovery is to allow grouping objects with similar patterns [139] and finding their common spatio-temporal properties and consequently providing them with appropriate services. On the other hand, this discovery helps to identify possible problems that are common to grouped

objects, e.g., traffic bottlenecks, and to plan for remedies. However, this similarity search is by no means an easy task due to different sampling rates, inherent imprecision of object location, different trajectory lengths, shifts in location and/or time components, and partial similarities involved [140, 20]. As the chapter explains, similarity can be defined on the basis of information that is directly or indirectly available. Therefore, to cover all possible cases, 'information reduction operators' are defined. By applying such operators, one level of information is ignored and a similarity notion is defined based on available information. This procedure continues until no more information can be ignored by applying the operators. The defined similarity notions are, in turn, used to group and classify moving objects.

7.2 Main contributions of the thesis

The main contribution of this thesis are as follows:

- It proposes an uncertainty handling method, which proves to reduce the error associated with the registered (raw) network-constrained moving object data by 36.45% on average.
- It proposes a trajectory determination technique, which results in more realistic determination of trajectories.
- It proposes spatio-temporal trajectory compression techniques to compress moving object data by achieving significant compression rates and low error committed.
- It defines a new spatio-temporal error notion to be used in compression techniques, as a measure describing how good the compression technique is.
- It defines different similarity notions for moving object trajectories.
- It proposes averaging method for object trajectories.
- It proposes grouping technique to classify and group moving objects with similar characteristics.

7.3 Plans ahead

Further study in the moving object domain can be carried out in two directions: *(i)* on general issues of moving objects that still need to be investigated, *(ii)* on the

aspects that were dealt with in this research and still need further improvement and development. Some of these plans are as follows:

- Provision of moving object data

  Although third parties, such as telephony companies, at the moment have time-stamped location data about their customers, obtaining such data has proved to be difficult due to the privacy issues involved. On the other hand, the attempts to simulate moving object data have not been so successful, as they are simple at best and far from realistic. One issue, needing serious attention, is to provide realistic moving object data as a means to test the proposed techniques in this domain. This can be done either by solving privacy issues, or by appropriate simulations that consider all factors playing a role in object movement.

- Investigation of effects of inclusion of more data types in moving object data stream

  This research was carried out on basis of the assumption of availability of time-stamped locations, only. If other data types such as direction or speed were occasionally needed, they were computed based on the position data available. However, since some data acquisition techniques can directly provide these (and even more) data, the possibility of extension of current techniques to accommodate these additional data should be investigated.

- Extending pattern recognition techniques to find object profiles and possible mis-behaviours

  The context-aware applications that have recently received serious attention, work on the basis of finding interests of their users and providing them with their services accordingly. By extending the similarity search techniques reported in this research, the repeated behaviour of the users can be identified and in turn be used to make/improve their profiles. The role of classification of objects for detecting possible mis-behavours should also be investigated.

- Inclusion and identification of possible errors in GIS

  One of the main assumptions in this research is that the data is erroneous, but not the the underlying network or the GIS. In future research, possible errors in the GIS should also be taken into account. One should note that one way to find out about such possible error is to use faithful representation of moving object trajectories.

- Real-time integration of information from different sources

  Since many moving object applications, like tourist information, advertising, traffic monitoring, require integration of information from different sources in real-time, techniques that work well for one source of information should be extended to support the kinds of dynamic, continuously evolving positional data from multiple sources that are found in mobility scenarios.

- Re-evaluation of queries in real-time applications

  In applications that work on the basis of providing real-time answers to moving object queries, mobility of objects directly influences validity of query results. In these applications, the system not only should find the 'best' (instead of 'all') answers, but also should frequently re-evaluate, re-transmit and re-display the results. Due to object mobility if these processes occur at inappropriate time intervals, the object may loose some of the desired opportunities. Defining an appropriate frequency for such processes is a necessity, and it needs further investigation.

- Indexing

  Considering the huge volume of moving object data and its multi-dimensionality, and the need for fast processes in real-time applications, efficient indexing methods are highly required. Further research in this direction should aim at providing spatio-temporal indexing techniques that can tackle current indexing problems in moving object applications.

Bibliography

[1] EGNOS: European satellite navigation system, http://www.galileo-industries.net.

[2] Geodesy, Space Geodesy Analysis Centre, GPS after Selective Availability (SA), http://www.auslig.gov.au/geodesy/gps/gps_sa.htm.

[3] Institute of Transportation Engineering (ITE), http://www.ite.org/geo/ch11text5.doc.

[4] NIST: Spatial data transfer standard (FIPS 173). Technical report, National Institute of Standards and Technology, Department of Commerce, Washington DC, USA, 1992.

[5] R. A. Abbaspour, M. R. Delavar, and R. Batouli. The issue of uncertainty propagation in spatial decision making. In K. Virrantaus and H. Tveite, editors, *Proceedings of the* 9^{th} *Scandinavian Research Conference on Geographical Information Science (SsanGIS'03)*, pages 57–65, Espoo, Finland, 4–6 June 2003. Department of Surveying, Helsinki University of Technology.

[6] M. Abdelguerfi, J. Givaudan, K. Shaw, and R. Ladner. The 2-3TR-tree, a trajectory-oriented index structure for fully evolving valid-time spatio-temporal datasets. In *Proceedings of the* 10^{th} *International Symposium on Advances in Geographic Information Systems (ACMGIS'02)*, pages 29–34, McLean, Virginia, USA, 8–9 November 2002. ACM Press.

[7] T. Abraham and J. F. Roddick. Survey of spatio-temporal databases. *GeoInformatica*, 3(1):61–99, 1999.

[8] P. K. Agarwal, L. J. Guibas, H. Edelsbrunner, J. Erickson, M. Isard, S. Har-Peled, J. Hershberger, C. Jensen, L. Kavraki, P. Koehl, M. Lin, D. Manocha, D. Metaxas, B. Mirtich, D. Mount, S. Muthukrishnan, D. Pai, E. Sacks, J. Snoeyink, S. Suri, and O. Wolfson. Algorithmic issues in modeling motion. *ACM Computing Surveys (CSUR)*, 34(4):550–572, 2002.

[9] R. Agrawal, C. Faloutsos, and A. Swami. Efficient similarity search in sequence databases. In D. B. Lomet, editor, *Proceedings of the 4th International Conference on Foundations of Data Organization and Algorithms (FODO'93)*, volume 730 of *Lecture Notes in Computer Science (LNCS)*, pages 69–86, Chicago, Illinois, USA, 13–15 October 1993. Springer Verlag.

[10] E. André, G. Bosch, G. Herzog, and T. Rist. Characterizing trajectories of moving objects using natural language path descriptions. In *Proceedings of the 7th International Conference on Artificial Intelligence (ECAI'86)*, pages 1–8, 1986.

[11] G. Ariav. An overview of TQuel. *ACM Transaction on Database Systems*, 11(4):499–527, 1986.

[12] M. P. Armstrong. Temporality in spatial databases. In *Proceedings of the GIS/LIS: Accessing the World*, volume 2, pages 880–889, Falls Church, Virginia, USA, 1988. American Society for Photogrammetry and Remote Sensing.

[13] A. Beller, T. Giblin, K. V. Le, S. Litz, T. Kittel, and D. Schimel. A temporal GIS prototype for global change research. In *Proceedings of the 1991 GIS/LIS: Accessing the World*, volume 2, pages 752–765, Atlanta, Georgia, USA, 1991. American Society for Photogrammetry and Remote Sensing.

[14] B. Berry. Approaches to regional analysis: A synthesis. *Association of American Geographers*, 54:2–11, 1964.

[15] B. Boutsinas and T. Gnardellis. On distributing the clustering process. *Pattern Recognition Letters*, 23(8):999–1008, 2002.

[16] P. S. Bradley and U. M. Fayyad. Refining initial points for K-Means clustering. In *Proceedings of the* 15th *International Conference on Machine Learning*, pages 91–99. Morgan Kaufmann, 1998.

[17] J. Brodeur, Y. Bédard, and M.-J. Proulx. Modelling geospatial application databases using UML-based repositories aligned with international standards in geomatics. In *Proceedings of the 8th International Symposium on Advances in Geographic Information Systems (ACMGIS'00)*, pages 39–46, Washington, D.C, USA, 6–11 November 2000. ACM Press.

[18] D. Buzan, S. Sclaroff, and G. Kollios. Extraction and clustering of motion trajectories in video. In *Proceedings of the 17th International Conference on Pattern Recognition (ICPR'04)*, volume 2, pages 521–524, Cambridge, UK, 23–26 August 2004. IEEE Computer Society.

[19] K.-P. Chan and W.-C. Fu. Efficient time series matching by wavelets. In *Proceedings of the 15th International Conference on Data Engineering (ICDE'99)*, pages 126–133, Sydney, Australia, 23-26 March 1999. IEEE Computer Society.

[20] L. Chen, M. T. Özsu, and V. Oria. Symbolic representation and retrieval of moving object trajectories. In *Proceedings of the 6th ACM SIGMM International Workshop on Multimedia Information Retrieval (MIR'04)*, pages 1–8, New York, NY, USA, 15–16 October 2004.

[21] Y. Chen, F. Rao, X. Yu, and D. Liu. CAMEL: A moving object database approach for intelligent location aware services. In M.-S. Chen, P. K. Chrysanthis, M. Sloman, and A. B. Zaslavsky, editors, *Proccedings of the 4th IEEE International Conference on Mobile Data Management (MDM'03)*, pages 331–334, Melbourne, Australia, 21–24 January 2003. Springer-Verlag.

[22] J. Chomicki and P. Z. Revesz. Constraint-based interoperability of spatiotemporal databases. In M. Scholl and A. Voisard, editors, *Proceedings of the 5th International Symposium on Advances in Spatial Databases (SSD'97)*, volume 1262 of *Lecture Notes in Computer Science (LNCS)*, pages 142–161, Berlin, Germany, 15–18 July 1997. Springer-Verlag.

[23] J. Chomicki and P. Z. Revesz. A geometric framework for specifying spatiotemporal objects. In *Proceedings of the 6th International Workshop on Temporal Representation and Reasoning (TIME'99)*, pages 41–46, Orlando, Florida, USA, 1–2 May 1999. IEEE Computer Society.

[24] K. K. W. Chu and M. H. Wong. Fast time-series searching with scaling and shifting. In V. Vianu and C. Papadimitriou, editors, *Proceedings of the 18th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS'99)*, pages 237–248, Philadelphia, Philadelphia, USA, 31 May – 2 June 1999. ACM Press.

[25] C. Claramunt and M. Thériault. *Recent Advances in Temporal Databases*,

chapter Managing Time in GIS: An Event-Oriented Approach, pages 23–42. Springer-Verlag, 1995.

[26] J. Clifford, A. Crocker, and A. Tuzhilin. *Temporal Databases: Theory, Design and Implementation*, chapter On the Completeness of Query Languages for Grouped and Ungrouped Historical Data Models, pages 496–533. The Benjamin/Cummings Publishing Company, 1993.

[27] M. Cooper. Antennas get smart. *Scientific American*, 283(7):48–55, July 2003.

[28] R. A. de By, editor. *Principles of Geographic Information Systems : An Introductory Textbook*. ITC Educational Textbook Series. ITC, second edition, 2001. ISBN 90-6164-200-0.

[29] M. N. Demers. *Fundamentals of Geographic Information System*. John Wiley & Sons Inc., 1997.

[30] M. Deng and F. Zhang. Spatial temporal queries and triggers for managing moving objects. In Y. Manolopoulos and P. Návrat, editors, *Proceedings of the 6th East-European Conference on Advances in Databases and Information Systems (ADBIS'02)*, volume 2435 of *Lecture Notes in Computer Science (LNCS)*, pages 88–97, Bratislava, Slovakia, 8–11 September 2002. Springer-Verlag.

[31] D. H. Douglas and T. K. Peucker. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *The Canadian Cartographer*, 10(2):112–122, 1973.

[32] M. J. Egenhofer and K. K. Al-Taha. Reasoning about gradual changes of topological relationships. In *Proceedings of the International Conference on GIS - From Space to Territory: Theories and Methods of Spatio-Temporal in Geographic Space*, pages 196–219. Springer-Verlag, 1992.

[33] M. Erwig, R. H. Güting, M. Schneider, and M. Vazirgiannis. Abstract and discrete modeling of spatio-temporal data types. In R. Laurini, K. Makki, and N. Pissinou, editors, *Proceedings of the 6th International Symposium on Advances in Geographic Information Systems (ACMGIS'98)*, pages 131–136, Washington, DC, USA, 6–7 November 1998. ACM Press.

[34] M. Erwig, R. H. Güting, M. Schneider, and M. Vazirgiannis. Spatio-temporal data types: An approach to modeling and querying moving objects in databases. *GeoInformatica*, 3(3):269–296, 1999.

[35] M. Erwig and M. Schneider. Developments in spatio-temporal query languages. In T. J. M. Bench-Capon, G. Soda, and A. M. Tjoa, editors, *Proceedings of the* 10^{th} *Workshop on Database and Expert Systems Applications (DEXA'99)*, pages 441–449, Florence, Italy, 1–3 September 1999. IEEE Computer Society.

[36] M. Erwig, M. Schneider, and R. H. Güting. Temporal and spatio-temporal data models and their expressive power. Technical Report CH-97-10, Chorochronos Research Project, December 1997.

[37] C. Faloutsos and K.-I. Lin. FastMap: A fast algorithm for indexing, datamining and visualization of traditional and multimedia datasets. In V. Vianu and C. Papadimitriou, editors, *Proceedings of 1995 ACM SIGMOD International Conference on Management of Data (SIGMOD'95)*, pages 163–174, San Jose, California, USA, 23–25 May 1999. ACM Press.

[38] J. D. Foley, A. van Dam, S. K. Feiner, and J. F. Hughes. *Computer Graphics: Principles and Practice.* Addison-Wesley Publishing Company, second edition, 1990.

[39] L. Forlizzi, R. H. Güting, E. Nardelli, and M. Schneider. A data model and data structures for moving objects databases. In W. Chen, J. F. Naughton, and P. A. Bernstein, editors, *Proceedings of 2000 ACM SIGMOD International Conference on Management of Data (SIGMOD'00)*, pages 319–330, Dallas, Texas, USA, 16–18 May 2000. ACM Press.

[40] M. Foss and G. J. Geier. *Understanding GPS: Principles and Applications*, chapter Integration of GPS With Other Sensors, pages 385–437. Artech House, INC., 1996. ISBN 0-89006-793-7.

[41] A. Friis-Christensen, N. Tryfona, and C. S. Jensen. Requirements and research issues in geographic data modeling. In W. G. Aref, editor, *Proceedings of the* 9^{th} *International Symposium on Advances in Geographic Information Systems (ACMGIS'01)*, pages 2–8, Atlanta, Georgia, USA, 9–10 November 2001. ACM Press.

[42] D. M. Gabbay and P. McBrien. Temporal logic & historical databases. In G. M. Lohman, A. Sernadas, and R. Camps, editors, *Proceedings of the 17th International Conference on Very Large Data Bases (VLDB'91)*, pages 423–430, Barcelona, Catalonia, Spain, 3–6 September 1991. Morgan Kaufmann.

[43] S. K. Gadia and S. S. Nair. *Temporal Databases: A Prelude to Parametric Data*, chapter Temporal Databases: Theory, Design and Implementation, pages 28–66. The Benjamin/Cummings Publishing Company.

[44] S. K. Gadia and J. H. Vaishnav. A query language for a homogeneous temporal database. In *Proceedings of the 4th International Symposium on Principles of Database Systems (PODS'85)*, pages 51–56, Portland, Oregon, USA, 23–27 March 1985. ACM Press.

[45] S. K. Gadia and C.-S. Yeung. A generalized model for a relational temporal database. In H. Boral and P. Larson, editors, *Proceedings of the 1988 International Symposium on Management of Data (SIGMOD'88)*, pages 251–259, Chicago, Illinois, USA, 1–3 June 1988. ACM Press.

[46] S. Gaffney and P. Smyth. Trajectory clustering with mixtures of regression models. In U. Fayyad, S. Chaudhuri, and D. Madigan, editors, *Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'99)*, pages 63–72, San Diego, California, USA, 15–18 August 1999. ACM Press.

[47] X. Ge and P. Smyth. Deformable Markov model templates for time-series pattern matching. In R. Ng, editor, *Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'00)*, pages 81–90, Boston, Massachuset, USA, 20–23 August 2000. ACM Press.

[48] A. Gelb. *Applied Optimal Estimation*. MIT Press, May 1974. ISBN 0-262-57048-3.

[49] A. Gionis, P. Indyk, and R. Motwani. Similarity search in high dimensions via hashing. In M. P. Atkinson, M. E. Orlowska, P. Valduriez, S. B. Zdonik, and M. L. Brodie, editors, *Proceedings of the 25th International Conference on Very Large Data Bases (VLDB'99)*, pages 518–529, Edinburgh, Scotland, UK, 7–10 September 1999. Morgan Kaufmann.

[50] D. Q. Goldin and P. C. Kanellakis. On similarity queries for time-series data: Constraint specification and implementation. In U. Montanari and F. Rossi, editors, *Proceedings of the First International Conference on Principles and Practice of Constraint Programming (CP'95)*, volume 976 of *Lecture Notes in Computer Science (LNCS)*, pages 137–153, Cassis, France, 19–22 September 1995. Springer-Verlag.

[51] S. Grumbach, P. Rigaux, and L. Segoufin. The DEDALE system for complex spatial queries. In L. M. Haas and A. Tiwary, editors, *Proceedings of 1998 ACM SIGMOD International Conference on Management of Data (SIGMOD'98)*, pages 213–224, Seattle, Washington, USA, 2–4 June 1998. ACM Press.

[52] S. Grumbach, P. Rigaux, and L. Segoufin. Spatio-temporal data handling with constraints. In R. Laurini, K. Makki, and N. Pissinou, editors, *Proceedings of the 6^{th} International Symposium on Advances in Geographic Information Systems (ACMGIS'98)*, pages 106–11, Washington, DC, USA, 6–7 November 1998. ACM Press.

[53] R. H. Güting. An introduction to spatial database systems. *The International Journal on Very Large Data Bases (VLDB), Special Issue on Spatial Database Systems*, 3(4):357–399, 1994.

[54] R. H. Güting, M. H. Böhlen, M. Erwig, C. S. Jensen, N. A. Lorentzos, M. Schneider, and M. Vazirgiannis. A foundation for representing and querying moving objects. *ACM Transactions on Database Systems*, 25(1):1–42, 2000.

[55] R. H. Güting and M. Schneider. Realms: A foundation for spatial data types in database systems. In D. J. Abel and B. C. Ooi, editors, *Proceedings of the 3^{th} International Symposium on Advances in Spatial Databases (SSD'93)*, volume 692 of *Lecture Notes in Computer Science (LNCS)*, pages 14–35, Singapore, 23–25 June 1993. Springer-Verlag.

[56] R. H. Güting and M. Schneider. Realm-based spatial data types: The ROSE algebra. *The VLDB Journal*, 4(2):243–286, 1995.

[57] P. Haggett, A. D. Cliff, and A. Fery. *Locational Models*. Halsted Press New York, 1977.

[58] J. Hershberger and J. Snoeyink. Speeding up the Douglas-Peucker line-simplification algorithm. In *Proceedings of the* 5^{th} *International Symposium on Spatial Data Handling (SDH'92)*, volume 2, pages 134–143, Charleston, South Carolina, USA, 3–7 August 1992. University of South Carolina, Humanities and Social Sciences Computing Lab.

[59] K. Hornsby and M. J. Egenhofer. Modeling moving objects over multiple granularities. *Annals of Mathematics and Artificial Intelligence*, 36(1-2):177–194, 2002.

[60] J. Ioup, M. L. Gendron, and M. C. Lohrenz. Vector map data compression with wavelets. *Journal of Navigation*, 53(3):437–449, 2000.

[61] M. Jasinski. The compression of complexity measures for cartographic lines. Technical report 90–1, National Center for Geographic Information and Analysis, Department of Geography. State University of New York at Buffalo, New York, USA, 1990.

[62] G. F. Jenks. Lines, computers, and human frailties. *Annuals of the Association of American Geographers*, 71(1):1–10, 1981.

[63] G. F. Jenks. Linear simplification: How far can we go? Paper presented to the Tenth Annual Meeting, Canadian Cartographic Association, 1985.

[64] C. S. Jensen. Research challenges in location-enabled M-services. In *Proceedings of the* 3^{th} *International Conference on Mobile Data Management (MDM'02)*, pages 3–7, Singapore, 8–11 January 2002. IEEE Computer Society.

[65] C. S. Jensen, A. Friis-Christensen, T. B. Pedersen, D. Pfoser, S. Šaltenis, and N. Tryfona. Location-based services – a database perspective. In *Proceedings of the* 8^{th} *Scandinavian Research Conference on Geographical Information Science (ScanGIS'01)*, pages 22–25, Ås, Norway, 22–25 June 2001.

[66] C. S. Jensen, A. Kligys, T. B. Pedersen, and I. Timko. Multidimensional data modeling for location-based services. In *Proceedings of the* 10^{th} *International Symposium on Advances in Geographic Information Systems (ACMGIS'02)*, pages 55–61, McLean, Virginia, USA, 8–9 November 2002. ACM Press.

[67] C. S. Jensen and R. T. Snodgrass. Temporal specialization. In F. Golshani, editor, *Proceedings of the* 8^{th} *International Conference on Data Engineering*

(ICDE'92), pages 594–603, Tempe, Arizona, USA, 3–7 February 1992. IEEE Computer Society.

[68] T. Kahveci and A. K. Singh. Variable length queries for time series data. In *Proceedings of the 17th International Conference on Data Engineering (ICDE'01)*, pages 273–282, Heidelberg, Germany, 2–6 April 2001. IEEE Computer Society.

[69] R. E. Kalman. A new approach to linear filtering and prediction problems. *ASME Journal of Basic Engineering*, 82(D):35–45, 1960.

[70] P. C. Kanellakis, G. M. Kuper, and P. Z. Revesz. Constraint query languages (preliminary report). In D. J. Rosenkrantz and Y. Sagiv, editors, *Proceedings of the 9th International Symposium on Principles of Database Systems (PODS'90)*, pages 299–313, Nashville, Tennessee, USA, 2–4 April 1990. ACM Press.

[71] K. V. R. Kanth, D. Agrawal, and A. K. Singh. Dimensionality reduction for similarity searching in dynamic databases. In L. M. Haas and A. Tiwary, editors, *Proceedings of 1998 ACM SIGMOD International Conference on Management of Data (SIGMOD'98)*, pages 166–176, Seattle, Washington, USA, 2–4 June 1998. ACM Press.

[72] J. F. Kenny and E. S. Keeping. *Mathematics of Statistics*. Van Nostrand, 1962.

[73] E. J. Keogh, K. Chakrabarti, S. Mehrotra, and M. J. Pazzani. Locally adaptive dimensionality reduction for indexing large time series databases. In W. G. Aref, editor, *Proceedings of the 2001 International Conference on Management of Data (SIGMOD'01)*, pages 151–162, Santa Barbara, California, USA, 21–24 May 2001. IEEE Computer Society.

[74] E. J. Keogh, K. Chakrabarti, M. J. Pazzani, and S. Mehrotra. Dimensionality reduction for fast similarity search in large time series databases. *Knowledge and Information Systems*, 3(3):263–286, 2001.

[75] E. J. Keogh, S. Chu, D. Hart, and M. J. Pazzani. An online algorithm for segmenting time series. In *Proceedings of 2001 IEEE International Conference on Data Mining (ICDM'01)*, pages 289–296, Silicon Valley, California, USA, 29 November – 2 December 2001. IEEE Computer Society.

[76] F. Korn, H. V. Jagadish, and C. Faloutsos. Efficiently supporting ad hoc queries in large datasets of time sequences. In J. M. Peckman, S. Ram, and M. Franklin, editors, *Proceedings of the 1997 International Conference on Management of Data (SIGMOD'97)*, pages 289–300, Tucson, Arizona, USA, 13–15 May 1997. ACM Press.

[77] N. S.-N. Lam. Spatial interpolation methods: A review. *The American Cartographer*, 10(2):129–149, 1983.

[78] T. Lang. Rules for robot draughtsmen. *Geographical Magazine*, 42(1):50–51, 1969.

[79] M. Langford, D. Maguire, and D. J. Unwin. *Handling Geographical Information: Methodology and Potential Applications*, chapter The Areal Interpolation Problem: Estimating Population Using Remote Sensing in a GIS framework, pages 55–77. Longman Scientific & Technical, 1991.

[80] G. Langran and N. R. Chrisman. A framework for temporal geographic information. *Cartographica*, 25(3):1–14, 1988.

[81] J. Z. Li. Modeling and querying multimedia data. Technical Report TR98-05, University of Alberta, Canada, March 1998.

[82] Z. Li. An algorithm for compressing digital contour data. *The Cartographic Journal*, 25:143–146, December 1988.

[83] D. Mark, M. Egenhofer, L. Bian, K. Hornsby, P. Rogerson, and J. Vena. Spatio-temporal GIS analysis for environmental health. Technical report, University of Maine, September 1999.

[84] B. Márkus. Decision support and error handling in GIS environment. Habilitation presentation, University of Sopron, Sopron, 1997.

[85] R. B. McMaster. Statistical analysis of mathematical measures of linear simplification. *The American Cartographer*, 13(2):103–116, 1986.

[86] R. B. McMaster. Automated line generalization. *Cartographica*, 24(2):74–111, 1987.

[87] X. Meng and Z. Ding. DSTTMOD: A future trajectory based moving objects database. In V. Marík, W. Retschitzegger, and O. Stepánková, editors, *Proceedings of the 14^{th} International Conference on Database and Expert*

System Applications (DEXA'03), volume 2736 of *Lecture Notes in Computer Science*, pages 20–35, Prague, Czech Republic, 1–5 September 2003. Springer-Verlag.

[88] E. M. Mikhail and F. E. Ackermann. *Observations and Least Squares*. Thomas Y. Crowell Company, 1976.

[89] B. Mirkin. *Mathematical Classification and Clustering*. Kluwer Academic Publishers, June 1996.

[90] H. Mokhtar and J. Su. Universal trajectory queries for moving object databases. In *Proccedings of the 5th IEEE International Conference on Mobile Data Management (MDM'04)*, pages 133–145. IEEE Computer Society, 2004.

[91] H. Mokhtar, J. Su, and O. Ibarra. On moving object queries. In *Proceedings of the 21th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS'02)*, pages 188–198, Madison, Wisconsin, USA, 3–6 June 2002. ACM Press.

[92] J. Moreira, C. Ribeiro, and T. Abdessalem. Query operations for moving objects database systems. In *Proceedings of the 8th International Symposium on Advances in Geographic Information Systems (ACMGIS'00)*, pages 108–114, Washington, D.C, USA, 6–11 November 2000. ACM Press.

[93] J. Moreira, C. Ribeiro, and J.-M. Saglio. Representation and manipulation of moving points: An extended data model for location estimation. *Cartography and Geographic Information Systems*, 26(2), 1999.

[94] M. Nabil, A. H. H. Ngu, and J. Shepherd. Modelling moving objects in multimedia databases. In R. W. Topor and K. Tanaka, editors, *Proceedings of the 5th International Conference on Database Systems for Advanced Applications (DASFAA'97)*, volume 6 of *Advanced Database Research and Development Series*, pages 67–76, Melbourne, Australia, 1–4 April 1997. World Scientific.

[95] N. S. Namini. A new approach for simplification of linear vector data for Internet-based GIS applications. Master thesis, Department of Geomatics Engineering, University of Calgary, Calgary, Canada, November 2002.

[96] M. Nanni. Distances for spatio-temporal clustering. In P. Ciaccia, F. Rabitti, and G. Soda, editors, *Decimo Convegno Nazionale su Sistemi Evoluti per Basi di Dati (SEBD'02)*, pages 135–142, Portoferraio (Isola d'Elba), Italy, 19–21 June 2002.

[97] H. M. Palancioglu and K. Beard. Modeling moving objects and their movements using fuzzy logic approach. In *Proceedings of ASPRS 2001 Annual Conference: Gateway to the New Millennium (ASPRS'01)*, St. Louis, Missouri, USA, 23–27 April 2001. American Society for Photogrammetry and Remote Sensing (ASPRS).

[98] C. Parent, S. Spaccapietra, and E. Zimányi. Spatio-temporal conceptual models: Data structures + space + time. In C. B. Medeiros, editor, *Proceedings of the 7th International Symposium on Advances in Geographic Information Systems (ACMGIS'99)*, pages 26–33, Kansas City, Missouri, USA, 2–6 November 1999. ACM Press.

[99] T. B. Pedersen, C. S. Jensen, and C. E. Dyreson. A foundation for capturing and querying complex multidimensional data. *Information Systems*, 26(5):383–423, 2001.

[100] D. Peuquet. It is about time: A conceptual framework for the representation of temporal dynamics in geographic information systems. *Annuals of the Association of American Geographers*, 84:441–461, 1994.

[101] D. Peuquet and N. Duan. An event-based spatiotemporal data model (ESTDM) for temporal analysis of geographical data. *International Journal of Geographical Information Systems*, 9(1):7–24, 1995.

[102] D. Peuquet and E. Wentz. An approach for time-based analysis of spatiotemporal data. In R. G. Healy, editor, *Proceedings of the 6th International Symposium on Spatial Data Handling (SDH'94)*, volume 1 of *Advances in GIS Research*, pages 489–504, Edinburgh, Scotland, 1994. Taylor & Francis.

[103] D. Pfoser and C. S. Jensen. Capturing the uncertainty of moving-object representations. In R. H. Güting, D. Papadias, and F. H. Lochovsky, editors, *Proceedings of the 6th International Symposium on Advances in Spatial Databases (SSD'99)*, volume 1651 of *Lecture Notes in Computer Science (LNCS)*, pages 111–132, Hong Kong, China, 20–23 July 1999. Springer-Verlag.

[104] D. Pfoser and C. S. Jensen. Indexing of network constrained moving objects. In E. Hoel and P. Rigaux, editors, *Proceedings of the 11th ACM International Symposium on Advances in Geographic Information Systems Database (ACMGIS'03)*, pages 25–32, New Orleans, Louisiana, USA, 7–8 November 2003. ACM Press.

[105] C. Plazanet, J. Affholder, and E. Fritsch. The importance of geometric modeling in linear feature generalization. *Cartography and Geographic Information Systems*, 22(4):291–305, 1996.

[106] F. Porikli. Trajectory distance metric using hidden Markov model based representation. In *Proceedings of the 6th International Workshop on Performance Evaluation of Tracking and Surveillance (PETS'04)*, Prague, Czech Republic, 10 May 2004.

[107] R. Price, B. Srinivasan, and K. Ramamohanarao. Extending the unified modeling language to support spatiotemporal applications. In *Proceedings of the 32th International Conference on Technology of Object-Oriented Languages (TOOLS'99)*, volume 32, pages 163–175. IEEE Computer Society, 22–25 November 1999.

[108] R. Price, N. Tryfona, and C. S. Jensen. Extended spatiotemporal UML: Motivations, requirements, and constructs. *Journal ef Database Management. Special Issue on UML*, 11(4):14–27, 2000.

[109] M.-J. Proulx and Y. Bédard. Perceptory, Centre de Recherche en Geomatique, Universite Laval, Quebec, Canada, http://sirs.scg.ulaval.ca/perceptory, 2000.

[110] Y. Qu, C. Wang, and X. S. Wang. Supporting fast search in time series for movement patterns in multiple scales. In G. Gardarin, J. C. French, N. Pissinou, K. Makki, and L. Bouganim, editors, *Proceedings of the 1998 ACM CIKM International Conference on Information and Knowledge Management (CIKM'98)*, pages 251–258, Bethesda, Maryland, USA, 3–7 November 1998. ACM Press.

[111] H. Raafat, Z. Yang, and D. Gauthier. Relational spatial topologies for historical geographical information. *International Journal of Geographical Information Systems*, 8:163–173, 1994.

[112] D. Rafiei and A. Mendelzon. Similarity-based queries for time series data. In J. M. Peckman, S. Ram, and M. Franklin, editors, *Proceedings of the 1997 International Conference on Management of Data (SIGMOD'97)*, pages 13–25, Tucson, Arizona, USA, 13–15 May 1997. ACM Press.

[113] D. Rafiei and A. Mendelzon. Querying time series data based on similarity. *IEEE Transaction on Knowledge and Data Engineering*, 12(5):675–693, January/February 2000.

[114] A. Rahimi and T. Darrell. Bayesian network for online global pose estimation. In *Proceedings of 2002 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'02)*, volume 1, pages 427–433, Lausanne, Switzerland, 30 September–4 October 2002. IEEE Computer Society.

[115] J. Raper and D. Livingstone. Development of a geomorphological spatial model using object-oriented design. *International Journal of Geographical Information Systems*, 9(4):359–384, 1995.

[116] H. C. Romesburg. *Cluster Analysis for Researchers*. Robert E. Krieger Publishing Company, 1990.

[117] S. Šaltenis, C. S. Jensen, S. T. Leutenegger, and M. A. Lopez. Indexing the positions of continuously moving objects. In W. Chen, J. F. Naughton, and P. A. Bernstein, editors, *Proceedings of 2000 ACM SIGMOD International Conference on Management of Data (SIGMOD'00)*, pages 331–342, Dallas, Texas, USA, 16–18 May 2000. ACM Press.

[118] S. Scott-Young and A. Kealy. An intelligent navigation solution for land mobile location based services. *The Journal of Navigation*, 55:225–240, 2002.

[119] A. Segev and A. Shoshani. *Temporal Databases: A Prelude to Parametric Data*, chapter Logical Modeling of Temporal Data, pages 248–170. The Benjamin/Cummings Publishing Company.

[120] T. K. Sellis. Research issues in spatio-temporal database systems. In R. H. Güting, D. Papadias, and F. H. Lochovsky, editors, *Proceedings of the 6^{th} International Symposium on Advances in Spatial Databases (SSD'99)*, volume 1651 of *Lecture Notes in Computer Science (LNCS)*, pages 5–11, Hong Kong, China, 20–23 July 1999. Springer-Verlag.

[121] D. Sinton. The inherent structure of information as a constraint to analysis: Mapped thematic data as a case study. *First International Advanced Study Symposium on topological data structures for Geographic Information Systems*, 7, 1978.

[122] A. P. Sistla, O. Wolfson, S. Chamberlain, and S. Dao. Querying the uncertain position of moving objects. In O. Etzion, S. Jajodia, and S. M. Sripada, editors, *Proceedings of Dagstuhl seminar on Temporal Databases: Research and Practice*, volume 1399 of *Lecture Notes in Computer Science (LNCS)*, pages 310–337, Dagstuhl Castle, Germany, 23–27 June 1997. Springer-Verlag.

[123] P. Sistla, O. Wolfson, S. Chamberlain, and S. Dao. Modeling and querying moving objects. In A. Gray and P. Larson, editors, *Proceedings of the 13th International Conference on Data Engineering (ICDE'97)*, pages 422–432, Birmingham, U.K, 7–11 April 1997. IEEE Computer Society.

[124] R. T. Snodgrass and I. Ahn. A taxonomy of time in databases. In S. B. Navathe, editor, *Proceedings of the 1985 International Symposium on Management of Data (SIGMOD'85)*, volume 14 of *SIGMOD Record*, pages 236–246, Austin, Texas, USA, 28–31 May 1985. ACM Press.

[125] L. Speičys, C. S. Jensen, and A. Kligys. Computational data modeling for network-constrained moving objects. In *Proceedings of the 11th International Symposium on Advances in Geographic Information Systems (ACMGIS'03)*, pages 118–125, New Orleans, Louisiana, USA, 7–8 November 2003. ACM Press.

[126] J. Su, H. Xu, and O. Ibarra. Moving objects: Logical relationships and queries. In C. S. Jensen, M. Schneider, B. Seeger, and V. J. Tsotras, editors, *Proceedings of the 7th International Symposium on Advances in Spatial and Temporal Databases (SSTD'01)*, volume 2121 of *Lecture Notes in Computer Science (LNCS)*, pages 3–19, Redondo Beach, California, USA, 12–15 July 2001. Springer-Verlag.

[127] J. Thurston, T. K. Poiker, and J. P. Moore. *Integrated Geospatial Technologies, A Guide to GPS, GIS and Data Logging.* John Wiley & Sons, INC., 2003. ISBN 0-471-24409-0.

[128] W. R. Tobler. Numerical map generalization. In J. D. Nystuen, editor, *IMaGe Discussion Papers*, Michigan Interuniversity Community of Mathemat-

ical Geographers, chapter 8. University of Michigan, Ann Arbor, Michigan, USA, 1966.

[129] G. Trajcevski, O. Wolfson, C. Hu, H. L. F. Zhang, and N. Rishe. Managing uncertain trajectories of moving objects with DOMINO. In C. S. Jensen, K. G. Jeffery, J. Pokorný, S. Šaltenis, E. Bertino, K. Böhm, and M. Jarke, editors, *Proceedings of the 4th International Conference on Enterprise Information Systems (ICEIS'02)*, volume 2287 of *Lecture Notes in Computer Science (LNCS)*, pages 218–225. Springer-Verlag, 2002.

[130] G. Trajcevski, O. Wolfson, F. Zhang, and S. Chamberlain. The geometry of uncertainty in moving objects databases. In C. S. Jensen, K. G. Jeffery, J. Pokorný, S. Šaltenis, E. Bertino, K. Böhm, and M. Jarke, editors, *Proceedings of the 8th International Conference on Extending Database Technology (EDBT'02)*, volume 2287 of *Lecture Notes in Computer Science (LNCS)*, pages 233–250. Springer-Verlag, 2002.

[131] N. Tryfona and T. Hadzilacos. Evaluation of database modeling methods for geographical information systems. Technical Report CH-97-05, Chorochronos Research Project, December 1997.

[132] N. Tryfona and T. Hadzilacos. Logical data modelling of spatio-temporal applications: Definitions and a model. Technical Report CH-97-03, Chorochronos Research Project, December 1997.

[133] N. Tryfona and C. S. Jensen. Conceptual data modeling for spatiotemporal applications. *Geoinformatica*, 3(3):245–268, 1999.

[134] N. Tryfona and C. S. Jensen. Using abstractions for spatio-temporal conceptual modeling. In *Proceedings of the 2000 ACM Symposium on Applied Computing (SAC'00)*, volume 2, pages 313–322, Villa Olmo, Como, Italy, 19–121 March 2000.

[135] D. J. Unwin. Geographical information systems and the problem of error and uncertainty. *Progress in Human Geography*, 19(4):549–558, 1995.

[136] M. Vazirgiannis, Y. Theodoridis, and T. Sellis. Spatio-temporal composition in multimedia applications. In *Proceedings of the 1996 International Workshop on Multimedia Software Development (MMSD'96)*, pages 120–127, Berlin, Germany, 25–26 March 1996. IEEE Computer Society.

[137] M. Vazirgiannis and O. Wolfson. A spatiotemporal model and language for moving objects on road networks. In *Proceedings of the 7th International Symposium on Advances in Spatial and Temporal Databases (SSTD'01)*, volume 2121 of *Lecture Notes in Computer Science*, pages 20–35, Redondo Beach, California, USA, 12–15 July 2001. Springer-Verlag.

[138] H. Veregin and X. Dai. Minimizing positional error induced by line simplification. In *Proceedings of the 1999 International Symposium on Spatial Data Quality (ISSDQ'99)*, Hong Kong, China, 18–20 July 1999.

[139] M. Vlachos, D. Gunopulos, and G. Kollios. Robust similarity measures for mobile object trajectories. In *Proceedings of the 13th International Workshop on Database and Expert Systems Application (DEXA'02), the 5th International Workshop on Mobility in Databases and Distributed Systems (MDDS'02)*, pages 721–728, Aix-en-Provence, France, 2–6 September 2002. IEEE Computer Society.

[140] M. Vlachos, G. Kollios, and D. Gunopulos. Discovering similar multidimensional trajectories. In *Proceedings of the* 18th *International Conference on Data Engineering (ICDE'02)*, page 673684, San Jose, California, USA, 26 February –1 March 2002. IEEE Computer Society.

[141] E. W. Weisstein. Confidence interval, from mathworld–a wolfram web resource. http://mathworld.wolfram.com/confidenceinterval.html.

[142] P. A. Whigham. Hierarchies of space and time. In A. U. Frank and I. Campari, editors, *Proceedings of 1993 International Conference on Spatial Information Theory: A Theoretical Basis for GIS (COSIT'03)*, volume 716 of *Lecture Notes in Computer Science (LNCS)*, pages 190–201, Marciana Marina, Elba Island, Italy, 19–22 September 1993. Springer-Verlag.

[143] E. R. White. Assessment of line generalization algorithms using characteristic points. *The American Cartographer*, 12(1):17–27, 1985.

[144] O. Wolfson. Moving objects information management: The database challenge. In A. Halevy and A. Gal, editors, *Proceedings of the 5th Workshop on Next Generation Information Technologies and Systems (NGITS'02)*, volume 2382 of *Lecture Notes in Computer Science (LNCS)*, pages 75–89, Caesarea, Israel, 24–25 June 2002. Springer-Verlag.

[145] O. Wolfson, S. Chamberlain, S. Dao, L. Jiang, and G. Mendez. Cost and imprecision in modeling the position of moving objects. In *Proceedings of the 14^{th} International Conference on Data Engineering (ICDE'98)*, pages 588–596, Orlando, Florida, USA, 23–27 February 1998. IEEE Computer Society.

[146] O. Wolfson, A. P. Sistla, B. Xu, J. Zhou, S. Chamberlain, Y. Yesha, and N. Rishe. Tracking moving objects using database technology in DOMINO. In R. Y. Pinter and S. Tsur, editors, *Proceedings of the 4^{th} International Workshop on Next Generation Information Technologies and Systems (NGITS'99)*, volume 1649 of *Lecture Notes in Computer Science (LNCS)*, pages 112–119, Zikhron-Yaakov, Israel, 5–7 July 1999. Springer-Verlag.

[147] O. Wolfson, P. Sistla, S. Chamberlain, and Y. Yesha. Updating and querying databases that track mobile units. *Distributed and Parallel Databases*, 7(3):257–387, 1999.

[148] O. Wolfson, P. Sistla, B. Xu, J. Zhou, and S. Chamberlain. DOMINO: Databased fOr MovINg Objects tracking. In *Proceedings of 1999 ACM SIGMOD International Conference on Management of Data (SIGMOD'99)*, pages 547–549, Philadelphia, Pennsylvania, USA, 1–3 June 1999. ACM Press.

[149] O. Wolfson, B. Xu, S. Chamberlain, and L. Jiang. Moving objects databases: Issues and solutions. In M. Rafanelli and M. Jarke, editors, *Proceedings of the 10^{th} International Conference on Scientific and Statistical Database Management (SSDBM'98)*, pages 111–122, Capri, Italy, 1–3 July 1998. IEEE Computer Society.

[150] M. F. Worboys. A model for spatio-temporal information. In *Proceedings of the 5^{th} International Symposium on Spatial Data Handling (SDH'92)*, volume 2, pages 602–611, Charleston, South Carolina, USA, 3–7 August 1992. University of South Carolina, Humanities and Social Sciences Computing Lab.

[151] M. F. Worboys. A unified model for spatial and temporal information. *The Computer Journal*, 37(1):27–34, 1994.

[152] D. Wu, D. Agrawal, A. E. Abbadi, A. K. Singh, and T. R. Smith. Efficient retrieval for browsing large image databases. In *Proceedings of the 5^{th} International Conference on Information and Knowledge Management*

(CIKM'96), pages 11–18, Rockville, Maryland, USA, 12–16 November 1996. ACM Press.

[153] Y. Yanagisawa, J. ichi Akahani, and T. Satoh. Shape-based similarity query for trajectory of mobile objects. In M.-S. Chen, P. K. Chrysanthis, M. Sloman, and A. B. Zaslavsky, editors, *Proccedings of the 4^{th} IEEE International Conference on Mobile Data Management (MDM'03)*, pages 63–77, Melbourne, Australia, 21–24 January 2003. Springer-Verlag.

[154] N. J. Yattaw. Conceptualizing space and time: A classification of geographic movement. *Cartography and Geographic Information Science*, 26(2):85–98, 1999.

[155] T.-S. Yeh and B. de Cambray. Time as a geometric dimension for modeling the evolution of entities: A 3D approach. In *International Conference on Integrating GIS and Environmental Modeling*, 1993.

[156] T.-S. Yeh and B. de Cambray. Modeling highly variable spatio-temporal data. In 6^{th} *AustraliAsian Database Conference*, pages 221–230, 1995.

[157] B.-K. Yi and C. Faloutsos. Fast time sequence indexing for arbitrary L_p norms. In A. E. Abbadi, M. L. Brodie, S. Chakravarthy, U. Dayal, N. Kamel, G. Schlageter, and K.-Y. Whang, editors, *Proceedings of the 26^{th} International Conference on Very Large Data Bases (VLDB'00)*, pages 385–394, Cairo, Egypt, 10–14 September 2000. Morgan Kaufmann.

[158] M. Yuan. Temporal GIS and spatio-temporal modeling. Department of Geography, The University of Oklahoma, 1996.

[159] M. Yuan. *Geographic Information Research: Bridging the Atlantic*, chapter Modeling Semantical, Temporal, and Spatial Information in Geographic Information Systems, pages 334–347. Taylor & Francis, 1997.

[160] H. Zhu, J. Su, and O. H. Ibarra. Trajectory queries and octagons in moving object databasess. In *Proceedings of 2002 ACM CIKM International Conference on Information and Knowledge Management (CIKM'02)*, pages 413–421, McLean, Virginia, USA, 4–9 November 2002.

SIKS Dissertation Series

[1998–1] Johan van den Akker (CWI), *DEGAS - An Active, Temporal Database of Autonomous Objects*

[1998–2] Floris Wiesman (UM), *Information Retrieval by Graphically Browsing Meta-Information*

[1998–3] Ans Steuten (TUD), *A Contribution to the Linguistic Analysis of Business Conversations within the Language/Action Perspective*

[1998–4] Dennis Breuker (UM), *Memory versus Search in Games*

[1998–5] E.W.Oskamp (RUL), *Computerondersteuning bij Straftoemeting*

[1999–1] Mark Sloof (VU), *Physiology of Quality Change Modelling; Automated modelling of Quality Change of Agricultural Products*

[1999–2] Rob Potharst (EUR), *Classification Using Decision Trees and Neural Nets*

[1999–3] Don Beal (UM), *The Nature of Minimax Search*

[1999–4] Jacques Penders (UM), *The Practical Art of Moving Physical Objects*

[1999–5] Aldo de Moor (KUB), *Empowering Communities: A Method for the Legitimate User-Driven Specification of Network Information Systems*

[1999–6] Niek J.E. Wijngaards (VU), *Re-design of Compositional Systems*

[1999–7] David Spelt (UT), *Verification Support for Object Database Design*

[1999–8] Jacques H.J. Lenting (UM), *Informed Gambling: Conception and Analysis of a Multi-Agent Mechanism for Discrete Reallocation*

[2000–1] Frank Niessink (VU), *Perspectives on Improving Software Maintenance*

[2000–2] Koen Holtman (TUE), *Prototyping of CMS Storage Management*

[2000–3] Carolien M.T. Metselaar (UvA), *Sociaal-organisatorische gevolgen van kennistechnologie; een procesbenadering en actorperspectief*

[2000–4] Geert de Haan (VU), *ETAG, A Formal Model of Competence Knowledge for User Interface Design*

[2000–5] Ruud van der Pol (UM), *Knowledge-based Query Formulation in Information Retrieval*

[2000–6] Rogier van Eijk (UU), *Programming Languages for Agent Communication*

[2000–7] Niels Peek (UU), *Decision-theoretic Planning of Clinical Patient Management*

[2000–8] Veerle Coupé (EUR), *Sensitivity Analyis of Decision-Theoretic Networks*

[2000–9] Florian Waas (CWI), *Principles of Probabilistic Query Optimization*

[2000–10] Niels Nes (CWI), *Image Database Management System Design Considerations, Algorithms and Architecture*

[2000–11] Jonas Karlsson (CWI), *Scalable Distributed Data Structures for Database Management*

[2001–1] Silja Renooij (UU), *Qualitative Approaches to Quantifying Probabilistic Networks*

[2001–2] Koen Hindriks (UU), *Agent Programming Languages: Programming with Mental Models*

[2001–3] Maarten van Someren (UvA) *Learning as problem solving*

[2001–4] Evgueni Smirnov (UM), *Conjunctive and Disjunctive Version Spaces with Instance-Based Boundary Sets*

[2001–5] Jacco van Ossenbruggen (VU), *Processing Structured Hypermedia: A Matter of Style*

[2001–6] Martijn van Welie (VU), *Task-based User Interface Design*

[2001–7] Bastiaan Schonhage (VU), *Diva: Architectural Perspectives on Information Visualization*

[2001–8] Pascal van Eck (VU), *A Compositional Semantic Structure for Multi-Agent Systems Dynamics*

[2001–9] Pieter Jan ' t Hoen (RUL), *Towards Distributed Development of Large Object-Oriented Models, Views of Packages as Classes*

[2001–10] Maarten Sierhuis (UvA), *Modeling and Simulating Work Practice BRAHMS: a multiagent modeling and simulation language for work practice analysis and design*

[2001–11] Tom M. van Engers (VUA), *Knowledge Management: The Role of Mental Models in Business Systems Design*

[2002–1] Nico Lassing (VU), *Architecture-Level Modifiability Analysis*

[2002–2] Roelof van Zwol (UT), *Modelling and Searching Web-based Document Collections*

[2002–3] Henk Ernst Blok (UT), *Database Optimization Aspects for Information Retrieval*

[2002–4] Juan Roberto Castelo Valdueza (UU), *The Discrete Acyclic Digraph Markov Model in Data Mining*

[2002–5] Radu Serban (VU), *The Private Cyberspace Modeling Electronic Environments inhabited by Privacy-concerned Agents*

[2002–6] Laurens Mommers (UL), *Applied legal epistemology; Building a knowledge-based ontology of the legal domain*

[2002–7] Peter Boncz (CWI), *Monet: A Next-Generation DBMS Kernel For Query-Intensive Applications*

[2002–8] Jaap Gordijn (VU), *Value Based Requirements Engineering: Exploring Innovative E-Commerce Ideas*

[2002–9] Willem-Jan van den Heuvel (KUB), *Integrating Modern Business Applications with Objectified Legacy Systems*

[2002–10] Brian Sheppard (UM), *Towards Perfect Play of Scrabble*

[2002–11] Wouter C.A. Wijngaards (VU), *Agent Based Modelling of Dynamics: Biological and Organisational Applications*

[2002–12] Albrecht Schmidt (UvA), *Processing XML in Database Systems*

[2002–13] Hongjing Wu (TUE), *A Reference Architecture for Adaptive Hypermedia Applications*

[2002–14] Wieke de Vries (UU), *Agent Interaction: Abstract Approaches to Modelling, Programming and Verifying Multi-Agent Systems*

[2002–15] Rik Eshuis (UT), *Semantics and Verification of UML Activity Diagrams for Workflow Modelling*

[2002–16] Pieter van Langen (VU), *The Anatomy of Design: Foundations, Models and Applications*

[2002–17] Stefan Manegold (UvA), *Understanding, Modeling, and Improving Main-Memory Database Performance*

[2003–1] Heiner Stuckenschmidt (VU), *Ontology-Based Information Sharing in Weakly Structured Environments*

[2003–2] Jan Broersen (VU), *Modal Action Logics for Reasoning About Reactive Systems*

[2003–3] Martijn Schuemie (TUD), *Human-Computer Interaction and Presence in Virtual Reality Exposure Therapy*

[2003–4] Milan Petkovic (UT), *Content-Based Video Retrieval Supported by Database Technology*

[2003–5] Jos Lehmann (UvA), *Causation in Artificial Intelligence and Law - A modelling approach*

[2003–6] Boris van Schooten (UT), *Development and specification of virtual environments*

[2003–7] Machiel Jansen (UvA), *Formal Explorations of Knowledge Intensive Tasks*

[2003–8] Yongping Ran (UM), *Repair Based Scheduling*

[2003–9] Rens Kortmann (UM), *The resolution of visually guided behaviour*

[2003–10] Andreas Lincke (UvT), *Electronic Business Negotiation: Some Experimental Studies on the Interaction Between Medium, Innovation Context and Culture*

[2003–11] Simon Keizer (UT), *Reasoning under Uncertainty in Natural Language Dialogue using Bayesian Networks*

[2003–12] Roeland Ordelman (UT), *Dutch Speech Recognition in Multimedia Information Retrieval*

[2003–13] Jeroen Donkers (UM), *Nosce Hostem - Searching with Opponent Models*

[2003–14] Stijn Hoppenbrouwers (KUN), *Freezing Language: Conceptualisation Processes across ICT-Supported Organisations*

[2003–15] Mathijs de Weerdt (TUD), *Plan Merging in Multi-Agent Systems*

[2003–16] Menzo Windhouwer (CWI), *Feature Grammar Systems - Incremental Maintenance of Indexes to Digital Media Warehouses*

[2003–17] David Jansen (UT), *Extensions of Statecharts with Probability, Time, and Stochastic Timing*

[2003–18] Levente Kocsis (UM), *Learning Search Decisions*

[2004–1] Virginia Dignum (UU), *A Model for Organizational Interaction: Based on Agents, Founded in Logic*

[2004–2] Lai Xu (UvT), *Monitoring Multi-party Contracts for E-business*

[2004–3] Perry Groot (VU), *A Theoretical and Empirical Analysis of Approximation in Symbolic Problem Solving*

[2004–4] Chris van Aart (UvA), *Organizational Principles for Multi-Agent Architectures*

[2004–5] Viara Popova (EUR), *Knowledge Discovery and Monotonicity*

[2004–6] Bart-Jan Hommes (TUD), *The Evaluation of Business Process Modeling Techniques*

[2004–7] Elise Boltjes (UM), *Voorbeeldig Onderwijs; Voorbeeldgestuurd Onderwijs, Een Opstap Naar Abstract Denken, Vooral Voor Meisjes*

[2004–8] Joop Verbeek(UM), *Politie en de Nieuwe Internationale Informatiemarkt, Grensregionale Politile Ggevensuitwisseling en Digitale Expertise*

[2004–9] Martin Caminada (VU), *For the Sake of the Argument; Explorations into Argument-based Reasoning*

[2004–10] Suzanne Kabel (UvA), *Knowledge-rich Indexing of Learning-objects*

[2004–11] Michel Klein (VU), *Change Management for Distributed Ontologies*

[2004–12] The Duy Bui (UT), *Creating Emotions and Facial Expressions for Embodied Agents*

[2004–13] Wojciech Jamroga (UT), *Using Multiple Models of Reality: On Agents who Know how to Play*

[2004–14] Paul Harrenstein (UU), *Logic in Conflict. Logical Explorations in Strategic Equilibrium*

[2004–15] Arno Knobbe (UU), *Multi-Relational Data Mining*

[2004–16] Federico Divina (VU), *Hybrid Genetic Relational Search for Inductive Learning*

[2004–17] Mark Winands (UM), *Informed Search in Complex Games*

[2004–18] Vania Bessa Machado (UvA), *Supporting the Construction of Qualitative Knowledge Models*

[2004–19] Thijs Westerveld (UT), *Using Generative Probabilistic Models for Multimedia Retrieval*

[2004–20] Madelon Evers (Nyenrode), *Learning from Design: Facilitating Multidisciplinary Design Teams*

[2005–1] Floor Verdenius (UvA), *Methodological Aspects of Designing Induction-Based Applications*

[2005–2] Erik van der Werf (UM), *AI techniques for the game of Go*

[2005–3] Franc Grootjen (RUN), *A Pragmatic Approach to the Conceptualisation of Language*

[2005–4] Nirvana Meratnia (UT), *Towards Database Support for Moving Object Data*

ITC Dissertations

1. **Akinyede, Joseph O.**, 1990, *Highway Cost Modelling and Route Selection Using a Geotechnical Information System*, Delft University of Technology

2. **Pan, Ping He**, 1990, *A Spatial Structure Theory in Machine Vision and Applications to Structural and Textural Analysis of Remotely Sensed Images*, University of Twente, 90-9003757-8

3. **Bocco Verdinelli, Gerardo H. R.**, 1990, *Gully Erosion Analysis Using Remote Sensing and Geographic Information Systems: A Case Study in Central Mexico*, Universiteit van Amsterdam

4. **Sharif, Massoud**, 1991, *Composite Sampling Optimization for DTM in the Context of GIS*, Wageningen Agricultural University

5. **Drummond, Jane E.**, 1991, *Determining and Processing Quality Parameters in Geographic Information Systems*, University of Newcastle

6. **Groten, Susanne**, 1991, *Satellite Monitoring of Agro-ecosystems in the Sahel*, Westfälische Wilhelms-Universität

7. **Sharifi, Ali**, 1991, *Development of an Appropriate Resource Information System to Support Agricultural Management at Farm Enterprise Level*, Wageningen Agricultural University, 90-6164-074-1

8. **van der Zee, Dick**, 1991, *Recreation Studied from Above: Air Photo Interpretation as Input into Land Evaluation for Recreation*, Wageningen Agricultural University, 90-6164-075-X

9. **Mannaerts, Chris**, 1991, *Assessment of the Transferability of Laboratory Rainfall-runoff and Rainfall—Soil Loss Relationships to Field and Catchment Scales: A Study in the Cape Verde Islands*, University of Ghent, 90-6164-085-7

10. **Wang, Ze Shen**, 1991, *An Expert System for Cartographic Symbol Design*, Utrecht University, 90-3930-333-9

11. **Zhou, Yunxuan**, 1991, *Application of Radon Transforms to the Processing of Airborne Geophysical Data*, Delft University of Technology, 90-6164-081-4

12. **de Zuviría, Martín**, 1992, *Mapping Agro-topoclimates by Integrating Topographic, Meteorological and Land Ecological Data in a Geographic Information System: A Case Study of the Lom Sak Area, North Central Thailand*, Universiteit van Amsterdam, 90-6164-077-6

13. **van Westen, Cees J.**, 1993, *Application of Geographic Information Systems to Landslide Hazard Zonation*, Delft University of Technology, 90-6164-078-4

14. **Shi, Wenzhong**, 1994, *Modelling Positional and Thematic Uncertainties in Integration of Remote Sensing and Geographic Information Systems*, Universität Osnabrück, 90-6164-099-7

15. **Javelosa R.**, 1994, *Active Quaternary Environments in the Philippine Mobile Belt*, Utrecht University, 90-6164-086-5

16. **Lo, King-Chang**, 1994, *High Quality Automatic DEM, Digital Elevation Model Generation from Multiple Imagery*, University of Twente, 90-9006-526-1

17. **Wokabi, S. M.**, 1994, *Quantified Land Evaluation for Maize Yield Gap Analysis at Three Sites on the Eastern Slope of Mt. Kenya*, University Ghent, 90-6164-102-0

18. **Rodríguez Parisca, O. S.**, 1995, *Land Use Conflicts and Planning Strategies in Urban Fringes: A Case Study of Western Caracas, Venezuela*, University of Ghent

19. **van der Meer, Freek D.**, 1995, *Imaging Spectrometry & the Ronda Peridotites*, Wageningen Agricultural University, 90-5485-385-9

20. **Kufoniyi, Olajide**, 1995, *Spatial Coincidence: Automated Database Updating and Data Consistency in Vector GIS*, Wageningen Agricultural University, 90-6164-105-5

21. **Zambezi, P.**, 1995, *Geochemistry of the Nkombwa Hill Carbonatite Complex of Isoka District, North-east Zambia, with Special Emphasis on Economic Minerals*, Vrije Universiteit Amsterdam

22. **Woldai, Tsehaie**, 1995, *The Application of Remote Sensing to the Study of the Geology and Structure of the Carboniferous in the Calañas Area, Pyrite Belt, South-west Spain*, Open University, United Kingdom

23. **Verweij, Pita A.**, 1995, *Spatial and Temporal Modelling of Vegetation Patterns: Burning and Grazing in the Páramo of Los Nevados National Park, Colombia*, Universiteit van Amsterdam, 90-6164-109-8

24. **Pohl, Christine**, 1996, *Geometric Aspects of Multisensor Image Fusion for Topographic Map Updating in the Humid Tropics*, Universität Hannover, 90-6164-121-7

25. **Bin, Jiang**, 1996, *Fuzzy Overlay Analysis and Visualization in Geographic Information Systemes*, Utrecht University, 90-6266-128-9

26. **Metternicht, Graciela I.**, 1996, *Detecting and Monitoring Land Degradation Features and Processes in the Cochabamba Valleys, Bolivia. A Synergistic Approach*, University of Ghent, 90-6164-118-7

27. **Chu Thai Hoanh**, 1996, *Development of a Computerized Aid to Integrated Land Use Planning (CAILUP) at Regional Level in Irrigated Areas: A Case Study for the Quan Lo Phung Hiep region in the Mekong Delta, Vietnam*, Wageningen Agricultural University, 90-6164-120-9

28. **Roshannejad, A.**, 1996, *The Management of Spatio-Temporal Data in a National Geographic Information System*,University of Twente, 90-9009-284-6

29. **Terlien, Mark T. J.**, 1996, *Modelling Spatial and Temporal Variations in Rainfall-triggered Landslides: The Integration of Hydrologic Models, Slope Stability Models and GIS for the Hazard Zonation of Rainfall-triggered Landslides with Examples from Manizales, Colombia*, Utrecht University, 90-6164-115-2

30. **Mahavir, J.**, 1996, *Modelling Settlement Patterns for Metropolitan Regions: Inputs from Remote Sensing*, Utrecht University, 90-6164-117-9

31. **Al-Amir, Sahar**, 1996, *Modern Spatial Planning Practice as Supported by the Multi-applicable Tools of Remote Sensing and GIS: The Syrian Case*, Utrecht University, 90-6164-116-0

32. **Pilouk, M.**, 1996, *Integrated Modelling for 3D GIS*, University of Twente, 90-6164-122-5

33. **Duan, Zengshan**, 1996, *Optimization Modelling of a River-Aquifer System with Technical Interventions: A Case Study for the Huangshui River and the Coastal Aquifer, Shandong, China*, Vrije Universiteit Amsterdam, 90-6164-123-3

34. **de Man, W. H. E.**, 1996, *Surveys: Informatie als Norm: Een Verkenning van de Institutionalisering van Dorp-surveys in Thailand en op de Filippijnen*, University of Twente, 90-9009-775-9

35. **Vekerdy, Zoltan**, 1996, *GIS-based Hydrological Modelling of Alluvial Regions: Using the Example of the Kisaföld, Hungary*, Lorand Eotvos University of Sciences, 90-6164-119-5

36. **Gomes Pereira, Luisa M.**, 1996, *A Robust and Adaptive Matching Procedure for Automatic Modelling of Terrain Relief*, Delft University of Technology, 90-407-1385-5

37. **Fandiño Lozano, M. T.**, 1996, *A Framework of Ecological Evaluation oriented at the Establishment and Management of Protected Areas: A Case Study of the Santuario de Iguaque, Colombia*, Universiteit van Amsterdam, 90-6164-129-2

38. **Toxopeus, Bert**, 1996, *ISM: An Interactive Spatial and Temporal Modelling System as a Tool in Ecosystem Management: With Two Case Studies: Cibodas Biosphere Reserve, West Java Indonesia: Amboseli Biosphere Reserve, Kajiado District, Central Southern Kenya*, Universiteit van Amsterdam, 90-6164-126-8

39. **Wang, Yiman**, 1997, *Satellite SAR Imagery for Topographic Mapping of Tidal Flat Areas in the Dutch Wadden Sea*, Universiteit van Amsterdam, 90-6164-131-4

40. **Saldana Lopez, Asun**, 1997, *Complexity of Soils and Soilscape Patterns on the Southern Slopes of the Ayllon Range Central Spain: a GIS Assisted Modelling Approach*, Universiteit van Amsterdam, 90-6164-133-0

41. **Ceccarelli, T.**, 1997, *Towards a Planning Support System for Communal Areas in the Zambezi Valley, Zimbabwe; A Multi-criteria Evaluation Linking Farm Household Analysis, Land Evaluation and Geographic Information Systems*, Utrecht University, 90-6164-135-7

42. **Peng, Wanning**, 1997, *Automated Generalization in GIS*, Wageningen Agricultural University, 90-6164-134-9

43. **Mendoza Lawas, M. C.**, 1997, *The Resource Users' Knowledge, the Neglected Input in Land Resource Management: The Case of the Kankanaey Farmers in Benguet, Philippines*, Utrecht University, 90-6164-137-3

44. **Bijker, Wietske**, 1997, *Radar for Rain Forest: A Monitoring System for Land Cover Change in the Colombian Amazon*, Wageningen Agricultural University, 90-6164-139-X

45. **Farshad, Abbas**, 1997, *Analysis of Integrated Soil and Water Management Practices within Different Agricultural Systems under Semi-arid Conditions of Iran and Evaluation of their Sustainability*, University of Ghent, 90-6164-142-X

46. **Orlic, B.**, 1997, *Predicting Subsurface Conditions for Geotechnical Modelling*, Delft University of Technology, 90-6164-140-3

47. **Bishr, Yaser**, 1997, *Semantic Aspects of Interoperable GIS*, Wageningen Agricultural University, 90-6164-141-1

48. **Zhang, Xiangmin**, 1998, *Coal fires in Northwest China: Detection, Monitoring and Prediction Using Remote Sensing Data*, Delft University of Technology, 90-6164-144-6

49. **Gens, Rudiger**, 1998, *Quality Assessment of SAR Interferometric Data*, University of Hannover, 90-6164-155-1

50. **Turkstra, Jan**, 1998, *Urban Development and Geographical Information: Spatial and Temporal Patterns of Urban Development and Land Values Using Integrated Geo-data, Villaviciencio, Colombia*, Utrecht University, 90-6164-147-0

51. **Cassells, Craig James Steven**, 1998, *Thermal Modelling of Underground Coal Fires in Northern China*, University of Dundee

52. **Naseri, M. Y.**, 1998, *Monitoring Soil Salinization, Iran*, Ghent University, 90-6164-195-0

53. **Gorte, Ben G. H.**, 1998, *Probabilistic Segmentation of Remotely Sensed Images*, Wageningen Agricultural University, 90-6164-157-8

54. **Ayenew, Tenalem**, 1998, *The Hydrological System of the Lake District Basin, Central Main Ethiopian Rift*, Universiteit van Amsterdam, 90-6164-158-6

55. **Wang, Donggen**, 1998, *Conjoint Approaches to Developing Activity-Based Models*, Technical University of Eindhoven, 90-6864-551-7

56. **Bastidas de Calderon, María**, 1998, *Environmental Fragility and Vulnerability of Amazonian Landscapes and Ecosystems in the Middle Orinoco River Basin, Venezuela*, University of Ghent

57. **Moameni, A.**, 1999, *Soil Quality Changes under Long-term Wheat Cultivation in the Marvdasht Plain, South-central Iran*, University of Ghent

58. **van Groenigen, J. W.**, 1999, *Constrained Optimisation of Spatial Sampling: A Geostatistical Approach*, Wageningen Agricultural University, 90-6164-156-X

59. **Cheng, Tao**, 1999, *A Process-oriented Data Model for Fuzzy Spatial Objects*, Wageningen Agricultural University, 90-6164-164-0

60. **Wolski, Piotr**, 1999, *Application of Reservoir Modelling to Hydrotopes Identified by Remote Sensing*, Vrije Universiteit Amsterdam, 90-6164-165-9

61. **Acharya, B.**, 1999, *Forest Biodiversity Assessment: A Spatial Analysis of Tree Species Diversity in Nepal*, Leiden University, 90-6164-168-3

62. **Abkar, Ali Akbar**, 1999, *Likelihood-based Segmentation and Classification of Remotely Sensed Images*, University of Twente, 90-6164-169-1

63. **Yanuariadi, Tetra**, 1999, *Sustainable Land Allocation: GIS-based Decision Support for Industrial Forest Plantation Development in Indonesia*, Wageningen University, 90-5808-082-X

64. **Abu Bakr, Mohamed**, 1999, *An Integrated Agro-Economic and Agro-Ecological Framework for Land Use Planning and Policy Analysis*, Wageningen University, 90-6164-170-5

65. **Eleveld, Marieke A.**, 1999, *Exploring Coastal Morphodynamics of Ameland (The Netherlands) with Remote Sensing Monitoring Techniques and Dynamic Modelling in GIS*, Universiteit van Amsterdam, 90-6461-166-7

66. **Hong, Yang**, 1999, *Imaging Spectrometry for Hydrocarbon Microseepage*, Delft University of Technology, 90-6164-172-1

67. **Mainam, Félix**, 1999, *Modelling Soil Erodibility in the Semiarid Zone of Cameroon*, University of Ghent, 90-6164-179-9

68. **Bakr, Mahmoud I.**, 2000, *A Stochastic Inverse-Management Approach to Groundwater Quality*, Delft University of Technology, 90-6164-176-4

69. **Zlatanova, Siyka**, 2000, *3D GIS for Urban Development*, Graz University of Technology, 90-6164-178-0

70. **Ottichilo, Wilber K.**, 2000, *Wildlife Dynamics: An Analysis of Change in the Masai Mara Ecosystem*, Wageningen University, 90-5808-197-4

71. **Kaymakci, Nuri**, 2000, *Tectono-stratigraphical Evolution of the Cankori Basin (Central Anatolia, Turkey)*, Utrecht University, 90-6164-181-0

72. **Gonzalez, Rhodora**, 2000, *Platforms and Terraces: Bridging Participation and GIS in Joint-learning for Watershed Management with the Ifugaos of the Philippines*, Wageningen University, 90-5808-246-6

73. **Schetselaar, Ernst**, 2000, *Integrated Analyses of Granite-gneiss Terrain from Field and Multisource Remotely Sensed Data. A Case Study from the Canadian Shield*, University of Delft, 90-6164-180-2

74. **Mesgari, M. Saadi**, 2000, *Topological Cell-Tuple Structure for Three-Dimensional Spatial Data*, University of Twente, 90-3651-511-4

75. **de Bie, Cees A. J. M.**, 2000, *Comparative Performance Analysis of Agro-Ecosystems*, Wageningen University, 90-5808-253-9

76. **Khaemba, Wilson M.**, 2000, *Spatial Statistics for Natural Resource Management*, Wageningen University, 90-5808-280-6

77. **Shrestha, Dhruba**, 2000, *Aspects of Erosion and Sedimentation in the Nepalese Himalaya: Highland-lowland Relations*, Ghent University, 90-6164-189-6

78. **Asadi Haroni, Hooshang**, 2000, *The Zarshuran Gold Deposit Model Applied in a Mineral Exploration GIS in Iran*, Delft University of Technology, 90-6164-185-3

79. **Raza, Ale**, 2000, *Object-Oriented Temporal GIS for Urban Applications*, University of Twente, 90-3651-540-8

80. **Farah, Hussein O.**, 2001, *Estimation of Regional Evaporation under Different Weather Conditions from Satellite and Meteorological Data. A Case Study in the Naivasha Basin, Kenya*, Wageningen University, 90-5808-331-4

81. **Zheng, Ding**, 2001, *A Neuro-Fuzzy Approach to Linguistic Knowledge Acquisition and Assessment in Spatial Decision Making*, University of Vechta, 90-6164-190-X

82. **Sahu, B. K.**, 2001, *Aeromagnetics of Continental Areas Flanking the Indian Ocean; with Implications for Geological Correlation and Gondwana Reassembly*, University of Capetown, South Africa

83. **Alfestawi, Yahia Ahmed M.**, 2001, *The Structural, Paleogeographical and Hydrocarbon Systems Analysis of the Ghadamis and Murzuq Basins, West Libya, with Emphasis on Their Relation to the Intervening Al Qarqaf Arch*, Delft Technical University, 90-6164-198-5

84. **Liu, Xuehua**, 2001, *Mapping and Modelling the Habitat of Giant Pandas in Foping Nature Reserve, China*, Wageningen University, 90-5808-496-5

85. **Oindo, Boniface Oluoch**, 2001, *Spatial Patterns of Species Diversity in Kenya*, Wageningen University, 90-5808-495-7

86. **Carranza, Emmanuel John M.**, 2002, *Geologically-constrained Mineral Potential Mapping: Examples from the Philippines*, Technical University of Delft, 90-6164-203-5

87. **Rugege, Denis**, 2002, *Regional Analysis of Maize-based Land Use Systems for Early Warning Applications*, Wageningen University, 90-5808-584-8

88. **Liu, Yaolin**, 2002, *Categorical Database Generalization in GIS*, Wageningen University, 90-5808-648-8

89. **Ogao, Patrick**, 2002, *Scientific Visualization*, Utrecht University, 90-6164-206-X

90. **Abadi, Abdulbaset Musbah**, 2002, *Tectonics of the Sirt Basin: Inferences from Tectonic Subsidence Analysis, Stress Inversion and Gravity Modeling*, Vrije Universiteit Amsterdam, 90-6164-205-1

91. **Geneletti, Davide**, 2002, *Ecological Evaluation for Environmental Impact Assessment*, Vrije Universiteit Amsterdam, 90-6809-337-1

92. **Sedogo, Laurent D.**, 2002, *Integration of Local Participatory and Regional Planning for Resources Management Using Remote Sensing and GIS*, Wageningen University, 90-5808-751-4

93. **Montoya, Ana Lorena**, 2002, *Urban Disaster Management: A Case Study of Earthquake Risk Assesment in Cartago, Costa Rica*, Utrecht University, 90-6164-2086

94. **Mobin-ud Din, Ahmad**, 2002, *Estimation of Net Groundwater Use in Irrigated River Basins Using Geo-information Techniques: A Case Study in Rechna Doab, Pakistan*, Wageningen University, 90-5808-761-1

95. **Said, Mohammed Yahya**, 2003, *Multiscale Perspectives of Species Richness in East Africa*, Wageningen University, 90-5808-794-8

96. **Schmidt, Karen S.**, 2003, *Hyperspectral Remote Sensing of Vegetation Species Distribution in a Saltmarsh*, Wageningen University, 90-5808-830-8

97. **López Binnqüist, Citlalli**, 2003, *The Endurance of Mexican Amate Paper: Exploring Additional dimensions to the Sustainable Development Concept*, University of Twente, 90-3651-900-4

98. **Huang, Zhengdong**, 2003, *Data Integration for Urban Transport Planning*, Utrecht University, 90-6164-211-6

99. **Cheng, Jianquan**, 2003, *Modelling Spatial and Temporal Urban Growth*, Utrecht University, 90-6164-212-4

100. **Campos dos Santos, José Laurindo**, 2003, *A Biodiversity Information System in an Open Data-Metadatabase Architecture*, University of Twente, 90-6164-214-0

101. **Hengl, Tomislav**, 2003, *Pedometric Mapping: Bridging the Gaps Between Conventional and Pedometric Approaches*, Wageningen University

102. **Barrera Bassols, Narciso**, 2003, *Symbolism, Knowledge and management of Soil and Land Resources in Indigenous Communities: Ethnopedology at Global, Regional and Local Scales*, University of Ghent

103. **Zhan, Qingming**, 2003, *A Hierarchical Object-based Approach for Urban Land-use Classification from Remote Sensing Data*, Wageningen University, 90-5808-917-7

104. **Daag, Arturo Santos**, 2003, *Modelling the Erosion of the Pyroclastic Flow Deposits and the Occurrences of Lahars at Mt. Pinatubo, Philipines*, Utrecht University, 90-6164-218-3

105. **Bacic, Ivan Luiz Zilli**, 2003, *Demand Driven Land Evaluation: With Case Studies in Santa Catarina, Brazil*, Wageningen University, 90-5808-902-9

106. **Murwira, Amon**, 2003, *Scale matters! A New Approach to Quantify Spatial Heterogeneity for Predicting the Distribution of Wildlife*, Wageningen University

107. **Mazvimavi, Dominic**, 2003, *Estimation of Flow Characteristics of Ungauged Catchments: A Case Study in Zimbabwe*, Wageningen University, 90-5808-950-9

108. **Tang, Xinming**, 2004, *Spatial Object Modeling in Fuzzy Topological Spaces: With Applications to Land Cover Change*, University of Twente, 90-6164-2205

109. **Kariuki, Patrick C.**, 2004, *Spectroscopy to Measure the Swelling Potential of Expansive Soils*, University of Delft, 90-6164-221-3

110. **Morales Guarin, Javier M.**, 2004, *Model-driven Design of Geo-information Services*, Twente University, 90-6164-222-1

111. **Mutanga, Onisimo**, 2004, *Hyperspectral Remote Sensing of Tropical Grass Quality and Quantity*, Wageningen Agricultural University, 90-5808-981-9

112. **Sliuzas, Richard V.**, 2004, *Managing Informal Settlements : A Study Using Geo-Information in Dar es Salaam, Tanzania*, Utrecht University, 90-6164-223-X

113. **Lucieer, Arko**, 2004, *Uncertainties in Segmentation and Their Visualisation*, Utrecht University, 90-6164-225-6

114. **Corsi, Fabio**, 2004, *Applications of Existing Biodiversity Information : Capacity to Support Decision-making*, Wageningen Agricultural University, 90-8504-090-6

115. **Tuladhar, Arbind M.**, 2004, *Parcel-based Geo-Information System: Concepts and Guidelines*, Delft Technical University, 90-6164-224-8

116. **van Elzakker, Corn'e P. G. M**, 2004, *The Use of Maps in the Exploration of Geographic Data*, Utrecht University, 90-6809-365-7

117. **Nidumolu, Uday Bhaskar**, 2004, *Integrating Geo-information Models with Participatory Approaches: Applications in Land Use Analysis*, Wageningen Agricultural University, 90-8504-138-4

118. **Koua, Etien L.**, 2005, *Computational and Visual Support for Exploratory Geovisualization and Knowledge Costruction*, Utrecht University, 90-6164-229-9

119. **Blok, Connie A.**, 2005, *Dynamic Visualization Variables in Animation to Support Monitoring of Spatial Phenomena*, Utrecht University, 90-6809-367-3

120. **Meratnia, Nirvana**, 2005, *Towards Database Support for Moving Object Data*, University of Twente, 90-365-2152-1

Abbreviations

ADR	Adaptive Dead-Reckoning
ADT	Abstract Data Type
APCA	Adaptive Piece-wise Constant Approximation
ASPRS	American Society for Photogrammetry and Remote Sensing
CWI	Centrum voor Wiskunde en Informatica
DBMS	DataBase Management System
DFT	Discrete Fourier Transform
DGPS	Differential Global Positioning System
DR	Dead-Reckoning
DTDR	Disconnection deTecting Dead-Reckoning
DTW	Dynamic Time Warping
DWT	Discrete Wavelet Transform
ER	Entity Relationship
ESTDM	Event-based Spatio-temporal Data Model
EUR	Erasmus Universiteit Amsterdam
FTL	Future Temporal Logic
GIS	Geographic Information System
GPS	Global Positioning System
ITE	Institute of Transportation Engineering
KUB	Katholieke Universiteit Brabant
LCSS	Longest Common SubSequence
LNCS	Lecture Notes in Computer Science
MADS	Modelling of Application Data with Spatio-temporal features
MOST	Moving Objects Spatio-Temporal
NIST	National Institute of Standards and Technology
PAA	Piece-wise Aggregate Approximation
PDA	Personal Digital Assistant
RMSE	Root Mean Square Error
RUL	Universiteit Leiden
SA	Selective Availability
SDR	Speed Dead-Reckoning

SDT	Spatial Data Type
SPS	Standard Positioning Service
STER	SpatioTemporal ER
STRM	Spatio-Temporal Relational Model
SVD	Single Value Decomposition
TMS	Temporal Map Set
TUD	Technische Universiteit Delft
TUE	Technische Universiteit Eindhoven
UM	Universiteit Maastricht
UML	Unified Modelling Language
UT	Universiteit Twente
UU	Universiteit Utrecht
UvA	Universiteit van Amsterdam
UvT	Universiteit van Tilburg
VLDB	Very Large DataBase
VU	Vrije Universiteit Amsterdam

Summary

Recently performance of portable computing devices has been greatly improved and at the same time enormous efforts have been directed towards miniaturization and personalization of wireless gadgets. All these efforts have opened new horizons for applications dealing with moving objects. Soon, it is expected that all the wireless personal devices start to generate an unprecedented data stream of time-stamped positions. At the same time, while these advances make it possible to introduce services that previously were unthought of, new serious challenges are ahead. That is because existing databases, which are one of the most important components for providing such services, are mainly good in static spatial data handling, while the need for space- and time- dependent data handling in the new spatio-temporal applications is rapidly increasing. The concept of mobility brings up a new set of requirements.

Moving objects, as one of the main players in mobility scenarios, are objects of which locations continuously changes. Despite of all work on databases, the situation in which moving objects are constantly monitored and their whereabouts are stored for future analysis forms an important class of problems that present-day database users find hard to tackle satisfactorily. This is because databases have not very well accommodated such data in the past, as their design paradigm was always one of 'snapshot representation'. Their present support for spatial time series is at best rudimentary. Moving object related challenges are manifold; data modelling, data structures and operations, indexing methods and query processing techniques that can handle continuous change in location of these objects, as well as their large amount of data, are just a few to mention.

To narrow down moving object challenges, the focus of this thesis is on four issues, namely, uncertainty handling for moving object data, faithful trajectory representation, trajectory compression techniques, and similarity measures for trajectories.

As far as moving point objects are concerned, three different types of movement

can be identified: *(i)* free movement in 2D or 3D space, *(ii)* restricted movement in 2D or 3D space, and *(iii)* restricted movement on 2D or 3D networks. Topics addressed in this thesis are mainly about the first and third case, and are restricted to 2D.

Since the error associated with (spatial) data tends to be propagated in later stages of analysis, error should be reduced to the best possible extent. Advances in positioning technology already offer early techniques to do so, resulting in 10–15 m accuracy at best. Although this may be already a good accuracy for some moving object applications, there are applications that require a higher degree of accuracy. For unconstrained moving objects, there is not much more to be done. However, for network-constrained objects, this thesis offers more intelligent methods to lower the error even further.

Moving objects are continuous phenomena. However, data about them is acquired and stored in a discrete way. Therefore, to have a full and complete understanding of the behaviour of these objects and their trajectories, techniques are needed to provide data when no observation is available. Two well-known and commonly used techniques to do so are linear and spline interpolations. By analyzing the behaviour of these two methods and showing their incapabilities, this thesis proposes an alternative approach to faithfully represent trajectories.

Dealing with moving objects, means dealing with a huge volume of data. Storing all this data is neither possible nor wise. The amount of data introduces numerous challenges, e.g., storage, indexing, and transmission overhead. Current compression techniques are not suitable for moving objects. They are either good for short, one dimensional time-series and in absence of noise, or for line generalization. The former does not hold for moving object data. The latter also has the problem of ignoring one important component of object data, i.e., time. This thesis introduces spatio-temporal techniques to overcome these problems.

Moving objects often have repeated patterns of movement. Identifying these patterns is a great help for planning and management purposes. However, considering the multi-dimensionality of respective data and its large volume, identifying these patterns is not an easy task. On the other hand, due to inherent imprecision of the data, these patterns may not be identical but similar. Therefore, defining similarity notions is a must. These notions can be defined at various levels, due to different data types, either explicitly or implicitly, available. Considering these issues, this thesis first defines similarity notions at different levels according to data availability and then proposes techniques to find similar movement patterns.

This identification is used to group and classify the objects.

Although, at first these four topics may seem independent, there is a strong relationship between them. A successful uncertainty handling method results in a more realistic trajectory representation, while this in turn results in more accurate pattern detection and classification. Reaching a compressed data set while avoiding serious information loss benefits the speed up of all the processes.

Experiments carried out support the proposed methods. Results show considerable improvements.

Samenvatting

De prestaties van draagbare computers en hand-helds zijn in de laatste jaren enorm verbeterd, terwijl deze apparaten ook aanmerkelijk kleiner en beter draadloos vernetwerkt zijn geworden, en beter afstembaar op persoonlijke wensen zijn gemaakt. Dit heeft geleid tot mogelijkheden van mobiliteitsscenario's, uitmondend in de verwachting dat zulke persoon-specifieke apparatuur binnen afzienbare tijd een niet eerder opgetreden hoeveelheid gegevens, met name reeksen van tijdsgebonden lokaties, zal genereren. Dit biedt weliswaar ongekende mogelijkheden voor datagebaseerde dienstverlening, maar serieuze obstakels dienen daarvoor uit de weg geruimd te worden. De reden is dat de hiervoor belangrijke databases wel met statische ruimtelijke gegevens overweg kunnen, maar niet goed met dynamische ruimtelijke gegevens. In het kort: gegevens rond mobiliteit stellen andere eisen van beheer en bevraging.

'Moving objects', een belangrijk onderdeel van mobiliteitsscenario's, zijn objekten met voortdurend wijzigende lokatie. Hun continue monitoring om redenen van analyse bepaalt een belangrijke klasse van problemen die met huidige databasetechnologie lastig te adresseren valt. Databases zijn slecht uitgerust om de eraan ten grondslag liggende gegevenssoorten te bewerken, aangezien ze historisch meer gericht zijn geweest op statische gegevens. Ondersteuning van tijdreeksgegevens is op z'n best rudimentair te noemen. Gegevens rond 'moving objects' bieden talloze uitdagingen: voortdurend wijzigende lokatie en hoog gegevensvolume zijn ttypsiche eigenschappen waarmee modellering, datastrukturering en -operaties, indiceringstechnieken en querybewerking te maken hebben.

In dit proefschrift worden een viertal thema's rond 'moving objects' nader onderzocht: behandeling van onzekerheid in 'moving object' gegevens, waarheidsgetrouwe representatie van trajektorieën van objekten, compressietechnieken van 'moving object' gegevens, en maten van gelijkvormigheid in 'moving object' trajektorieën.

Waar het 'moving point objects' betreft, kunnen drie soorten beweging worden

onderscheiden: *(i)* vrije beweging in twee- of drie-dimensionele ruimte, *(ii)* beperkte beweging in twee- of drie-dimensionele ruimte, en *(iii)* beperkte beweging over twee- of drie-dimensionele ruimtelijke netwerken. In dit proefschrift wordt vooral naar het eerste en derde geval gekeken, in een twee-dimensionele omgeving.

Aangezien fouten in ruimtelijke gegevens typisch doorwerken in latere bewerkingen, dienen zij geminimaliseerd te worden in de brongegevens. Verbeterde positioneringstechnieken bieden daartoe al mogelijkheden, leidend tot een nauwkeurigheid in de ordegrootte van 10–15 m. Dit is goed genoeg voor sommige 'moving object' toepassingen, maar niet voor andere die een hogere nauwkeurigheid vereisen. In omstandigheden waarin de objekten gedacht worden zich vrij te bewegen valt aan de nauwkeurigheid niet veel meer te verbeteren, bij beweging over een netwerk echter wel. Het proefschrift beschrijft enkele handige methoden.

'Moving objects' zijn continue fenomenen. Het monitoringsproces genereert echter typisch discrete gegevens. Voor een volledig begrip van het gedrag van de objekten zijn derhalve technieken nodig om gegevens van tijd en lokatie af te leiden als die niet rechtstreeks beschikbaar zijn. Twee bekende technieken zijn lineaire en spline-interpolatie. Middels een analyse van hun gedrag komen enkele van hun beperkingen aan het licht; het proefschrift stelt een alternatieve techniek voor om tot waarheidsgetrouwe representatie te komen.

Intrinsiek aan het beheer van 'moving object' gegevens is het hoge gegevensvolume. De opslag van domweg alle gegevens is nauwelijks mogelijk en is ook niet verstandig. Talloze technische uitdagingen gaan ermee gepaard. Bestaande compressietechnieken zijn niet erg geschikt voor 'moving object' gegevens. Zij zijn toepasbaar op korte, een-dimensionele tijdreeksen zonder ruis op de gegevens, of, indien men geometrische lijngeneralisatie wenst toe te passen. Het eerste geldt niet voor 'moving object' gegevens. Het tweede negeert de belangrijke temporele karakteristiek ervan. Het proefschrift beschrijft spatiotemporele technieken om aan deze bezwaren tegemoet te komen.

'Moving objects' vertonen regelmatig vergelijkbare bewegingen en zich herhalende bewegingspatronen. De herkenning hiervan is van groot belang in planning en beheer. Gezien het multi-dimensionele karakter en het grote volume van de gegevens, is het herkennen van deze patronen geen sinecure. Daar komt bij dat, vanwege intrinsieke onnauwkeurigheid, gelijkende bewegingspatronen niet identiek hoeven zijn. Een of meerdere noties van gelijkvormigheid zijn derhalve noodzakelijk. Zij kunnen worden gedefinieerd op meerdere niveaus, vanwege verschillen in gegevenstype, en zowel impliciet als expliciet. Dit wordt in dit proefschrift ge-

daan, waarna diverse technieken worden beschreven om gelijkvormige bewegingspatronen te ontdekken. Zulke overeenkomsten worden vervolgens gebruikt om tot klassifikatie en aggregatie van 'moving objects' te komen.

Op het eerste gezicht lijken de vier behandelde thema's onafhankelijk, maar er bestaat een duidelijk verband. Een geslaagde behandeling van onzekerheid in de gegevens leidt tot een realistischer representatie, en op zijn beurt tot een verbeterde patroonherkenning. Het comprimeren van de gegevens, mist zonder verlies van essentiële informatie, verbetert op zijn beurt de tijdscomplexiteit van de diverse rekenprocessen.

Uitgevoerde experimenten ondersteunen de beweringen omtrent de voorgestelde technieken. De resultaten laten opvallende verbeteringen zien.

Persian summary

چکیده:

درطلیعه هزاره سوم، بهبود روزافزون فناوری‌های محاسباتی از یک‌سو و پیشرفت‌های سریع در کوچک سازی و شخصی‌سازی تجهیزات الکترونیکی (که به نوبه خود منجر به کاهش شدید قیمت این دستگاه‌ها گردیده است) از سوی دیگر، افق‌های جدیدی را پیش روی برنامه‌های کاربردی مربوط به پدیده‌های متحرک گشوده است. چنانکه انتظار می رود که به‌زودی تمامی این دستگاه‌ها انبوهی از داده‌های جغرافیایی مکان دار و زمان دار را تولید کنند. اگرچه پیشرفت‌هایی از این دست امکان ارائه سرویس‌هایی که قبلا حتی در مخیله نمی گنجید را فراهم آورده، ولی درعین حال چالش های متعدد جدیدی را نیزدر پیش رو قرار داده است. چرا که پایگاه‌های داده‌ای به عنوان یکی از مهم‌ترین مولفه‌هایی که وجود چنین سرویس‌هایی را محقق می سازند، تنها برای داده‌های جغرافیایی ساکن و غیر متحرک مناسب هستند، وحال آنکه نیاز به بهره‌گیری از داده‌های جغرافیایی متحرک روز به روز در حال افزایش است، ومفهوم قابلیت تحرک در سامانه‌های جغرافیایی مجموعه ای از الزامات و نیازهای جدید را در پی دارد.

پدیده‌های متحرک *(Moving Objects)* به عنوان مهم‌ترین نقش آفرینان سناریوی حرکت، پدیده‌هایی هستند که مکانشان بطور مستمر و دایم در حال تغییر است. علیرغم تمام مساعی به عمل آمده در زمینه پایگاه‌های داده‌ای، مشاهده و دنبال کردن مداوم پدیده‌های متحرک و ذخیره‌سازی داده‌های مربوط به آنها، مشکلات فراوانی را برای کاربران فعلی پایگاه‌های داده‌ای فراهم آورده است وازآنجا که در حال حاضرپایگاه های داده ای امکانات

مورد نیاز برای کار بر روی چنین داده‌هایی را ندارند، این خود به یکی از عوامل مهم ناخشنودی و دشواری کاربران امروزی چنین سیستم هایی بدل شده است. این امر ناشی از آن است که تا پیش از این پایگاه‌های داده‌ای تنها با هدف مدیریت داده های ساکن و یا حداکثر برش‌های مقطعی زمان‌دار از اطلاعات مکان‌دار طراحی شده بودند. در نتیجه قابلیت‌ها و پشتیبانی امروزی پایگاه‌های داده ای برای سری‌های زمانی داده های مکان‌دار درنهایت بسیارابتدائی است. چالش های مرتبط با پدیده‌های متحرک برای پایگاه‌های داده‌ای امروزی بسیار متنوع و متعدد است؛ مدل سازی داده‌ای (Data Modelling)، ساختار داده‌ها و عملگرهای داده‌ای (Data Structures and Operators) روش‌های نمایه‌سازی (Indexing) و تکنیکهای پرس وجو(Query Processing) بطوری که توانایی مدیریت حجم انبوه داده‌های جغرافیایی که مستمراً در حال تغییر هستند را داشته باشند، تنها نمونه هایی از مشکلات مطرح شده اند.

به علت تعدد مشکلات و چالش های مطروحه، تلاشهای ما در این پایان نامه، تنها برروی چهار مورد متمرکز است که عبارتند از: "کنترل خطاهای همراه داده‌های مربوط به پدیده‌های متحرک"، "تعیین دقیق مسیر پیموده شده توسط پدیده های متحرک"، "فشرده سازی داده‌های مربوط به پدیده‌های متحرک" و "جستجوی الگوهای مشابه در مسیرهای پیموده شده توسط پدیده‌های متحرک".

برای پدیده‌های متحرک سه نوع حرکت قابل تشخیص است. (**الف**) حرکت آزاد در فضای دو یا سه بعدی، (**ب**) حرکت محدود در فضای دو یا سه بعدی، و (**ج**) حرکت محدود بر روی شبکه‌های دو یا سه بعدی، که تنها در این کتاب موارد (الف) و (ج) آنهم در فضای دوبعدی مورد بحث قرار می گیرند.

از آنجا که امکان افزایش خطاهای همراه داده در مراحل متعدد تحلیل وجود دارد، این خطاها باید در مراحل مقدماتی به بهترین وجهی کاهش یابند. روشهای موجود در سیستم‌های مکان‌یابی (Positioning System) در حال حاضر واجد سازو کارهایی هستند که در بهترین حالت امکان تامین داده‌های مکان‌دار با حداکثر پنج متر خطا را دارند. اگرچه چنین خطایی می‌تواند مورد قبول بسیاری از برنامه‌های کاربردی (به ویژه سیستم‌های

تفسیر اطلاعات ماهواره‌ای و سنجش از دور) قرار گیرد، ولی برنامه‌های کاربردی متعدد دیگری وجود دارند که در آنها چنین خطایی هنوز بزرگ انگاشته شده و در نتیجه قابل قبول نمی باشد. متاسفانه برای پدیده‌های متحرکی که بصورت آزادانه حرکت می‌کنند در حال حاضر اقدام دیگری در جهت کاهش خطا نمی‌توان انجام داد، ولی برای پدیده‌هایی که برروی شبکه حرکت می‌کنند در این پایان‌نامه روش‌های هوشمندانه‌ای پیشنهاد شده‌اند که موجب کاهش خطای همراه داده می‌گردند.

پدیده‌های متحرک پدیده‌هایی پیوسته هستند و حال آنکه داده‌های مربوط به آنها به صورت ناپیوسته جمع‌آوری و ذخیره می‌شوند. در نتیجه برای درک کامل رفتار این پدیده‌ها و همچنین تعریف دقیق مسیر پیموده شده توسط آنها، می‌بایست روش‌هایی بکار گرفته شوند که قادر به تامین داده‌ها یی که از ابتدا جمع‌آوری و ذخیره نشده‌اند، باشند. معروف‌ترین و متداول‌ترین روش‌ها دراین راستا Spline و Linear Interpolation می‌باشند. با تحلیل رفتار این دو تکنیک و مشخص نمودن نقاط ضعف آنها، این کتاب روش جایگزینی جهت تعیین هرچه دقیق‌تر و واقعی تر مسیر پیموده شده توسط پدیده‌های متحرک را پیشنهاد می کند.

بحث در مورد پدیده‌های متحرک در حقیقت بحث در مورد روش‌های مدیریت حجم انبوهی از داده‌ها است. ذخیره‌سازی تمامی این داده ها نه امکان پذیر است و نه عاقلانه. چنین حجم عظیمی از داده هزینه‌های عمده‌ای را در امر ذخیره سازی، نمایه‌سازی و انتقال داده‌ها بهمراه دارد. روش‌های فشرده‌سازی داده‌ها در حال حاضر برای پدیده‌های متحرک کاربری ندارند، چرا که تنها برای رشته‌های داده‌ای کوتاه یک بعدی زمان‌دار و بدون خطا مناسب می‌باشند و داده‌های مربوط به پدیده‌های متحرک واجد چنین ویژگی‌هایی نیستند. از سوی دیگر یکی از پایه‌های مهم روش‌هایی که امروزه در سیستم‌های اطلاعات جغرافیایی به منظور فشرده‌سازی داده‌ها استفاده می‌گردند، صرف نظر کردن از بخش مهمی از داده‌ها، یعنی داده زمان‌دار است. این کتاب روش‌های جدیدی را به منظور حل مشکلات کنونی روش‌های فشرده‌سازی داده‌های پدیده‌های متحرک و با در نظر گرفتن هر دو عامل زمان و مکان ارائه می کند.

پدیده‌های متحرک عموما الگوهای حرکتی تکراری دارند. تشخیص این الگوها کمک عمده‌ای در برنامه‌ریزی و مدیریت محسوب می‌شوند. اما با توجه به چند بعدی بودن و حجم عظیم داده‌های مربوط به پدیده‌های متحرک، تشخیص این الگوها کار ساده ای نیست. ازسوی دیگر خطای موجود درداده‌ها موجب می‌شود که الگوهای حرکتی تکراری بجای تظاهر یکسان و همتا، تظاهری مشابه داشته باشند. نتیجتا تعریف مفهوم و شاخص‌های "شباهت" الزامی است. به جهت انواع گوناگون داده ها، که بصورت مستقیم و یا غیر مستقیم در دسترس می‌باشند، می توان "شباهت" و مشخصه‌های آن را در سطوح مختلف تعریف نمود. از این رو این کتاب با توجه به انواع داده‌های موجود، ابتدا به تعریف "شباهت" در سطوح مختلف می‌پردازد وسپس تکنیک‌هایی را جهت یافتن الگوهای مشابه پیشنهاد می‌کند. تشخیص این الگوها به نوبه خود یاریگر دسته‌بندی و طبقه‌بندی پدیده‌های متحرک خواهد بود.

اگرچه در ابتدا ممکن است که چهار مبحث مطروحه در این کتاب مستقل از هم بنظر برسند، ولی ارتباطی تنگاتنگ بین آنها وجود دارد. زیرا یک روش موفق کاهش خطا، منجر به تعیین دقیق‌تر و واقعی‌تر مسیر پیموده شده توسط پدیده مورد نظر می‌گردد، و این امر به نوبه خود به تشخیص صحیح الگوهای تکراری و طبقه بندی مناسب پدیده‌های متحرک کمک می کند. ضمن آنکه فشرده‌سازی داده‌ها بدون از دست دادن اطلاعات عمده وحیاتی نهفته در آنها، تاثیر عمده‌ای در تسریع تمامی پردازش‌ها و فرآیندهای مذکور دارد.

نتایج شبیه‌سازی‌ها، تجربیات و آزمایشات انجام شده تمامی روش‌های پیشنهادی را تایید می‌کنند.