Universita' degli Studi di Salerno

Dipartimento di Ingegneria Elettronica ed Ingegneria Informatica

Dottorato di Ricerca in Ingegneria dell'Informazione
X Ciclo - Nuova Serie

TESI DI DOTTORATO

Efficient Indexing and Retrieval from Large Moving Object Databases through Dynamic Spatio-Temporal Queries

CANDIDATO: **Marco LEONE**

TUTOR: **Prof. Mario VENTO**

CO-TUTOR: **Prof. Gerardo IOVANE**

COORDINATORE: **Prof. Angelo MARCELLI**

Anno Accademico 2011-2012

Abstract

Intelligent Transportation Systems have gained a great importance in the last decades given the growing need for security in many public environments, with particular attention for traffic scenarios, which are daily interested by accidents, traffic queues, highway code violations, driving in the wrong lane or on the wrong side, and so on.

In the context of camera-based traffic analysis systems, in this thesis I will present a novel indexing scheme for the design of a system for the extraction, the storage and retrieval of moving objects' trajectories from surveillance cameras.

Once spatio-temporal data have been collected, Moving Object Databases (MODs) are a widely adopted solution for the storage and indexing of data relating to moving objects. Among the various approaches proposed in literature, only a modest attention has been devoted to storing and retrieving systems able to cope with very large amount of trajectory data and sufficiently general to deal with the requirements of different application domains. This is an important and not negligible feature, especially when considering crowded real world scenarios (like highway intersections, city crossroads and important junctions). In these cases it is required that billions of trajectories must be stored and that, on this wide database, the user must be able to submit complex queries involving geometric and temporal data. One of the main limitations of such systems is the impossibility to choose at query time (i.e. exactly when the query is thought) the area of interest.

As for MODs and spatial databases, even in presence of efficient solutions from different perspectives, there is no support of indexing operations for three-dimensional data, both in commercial and freely available products. For instance, PostGIS, the well-known extension of PostgreSQL DBMS for storing spatial data, while supporting three (and even four)-dimensional data, does not support three-dimensional intersection and indexing operations.

Starting from these limitations, this dissertation will present an indexing scheme capable of reformulating any three (and theoretically N)-dimensional problem in terms of bi-dimensional subproblems, so taking advantage of existing and efficient 2D spatial indexes.

In order to optimize the indexes' efficiency, a segmentation algorithm will be introduced, performed at loading time and aimed at the reduction of the redundancy introduced by the trajectory representation.

Once data have been collected and properly stored and indexed, our system allows to efficiently solve *Dynamic Spatio-Temporal* (*DST*) queries, which are a novel type of queries allowing the choice of the query parameters at runtime.

The entire trajectory analysis approach has been tested over both synthetic and real-world data. In particular, in order to obtain better synthetic data, in terms of number of trajectories, average trajectory length and contextual relationship of trajectories and topology, a human behavior simulation model has been developed according to the Social Force Models.

Finally, query processing has been contextualized for the solution of a given application domain, the traffic flow analysis, with the formalization of the Flow- and Multi-DST queries.

Keywords:
[Trajectory Analysis, Activity Analysis, Spatio-Temporal Indexing, Trajectory Segmentation, Query Processing, Dynamic Spatio-Temporal Queries]

Acknowledgements

This four-year PhD has proven to be a great professional and personal experience, plenty of satisfactions and difficulties which have contributed to my personal growth from different perspectives. For this reasons I am sincerely grateful to all the people who, in various ways, have contributed to make this informative journey more interesting and enjoyable.

First of all I would like to express my great gratitude for having been co-advised by two admirable professors, Prof. Mario Vento and Prof. Gerardo Iovane: without their competence, their scientific advices, their patience, their helpfulness, their constant support, their natural attitude to problem solving both related to scientific and non-scientific issues, my last four years would not have been the same at all.

Special thanks go to all the members of the Machine Intelligence lab for Video, Image and Audio processing, Prof. Pasquale Foggia for his spread culture and his analytical capacity to give so productive reviews, Prof. Gennaro Percannella for his capacity to possess pronounced managerial abilities and scientific competence, Prof. Donatello Conte for the passion and the modesty with which he carries on his valuable career, and, of course, my friends and colleagues Alessia, Vincenzo and Rosario. All of them have contributed to make me spend nice moments, even when difficulties seemed to gain the upper hand: thank you all for your support, your encouragements and, most of all, your friendship.

Least but certainly most important thanks go to my family, the only one who will always support me, no matter what choices I will make, what places I will be in, what atmosphere will be above this earth: my mum, my father, my brother and Nunzia have been, are and will always be my reference.

Contents

List of Figures

Introduction

Everyday life has changed singificantly in the last decades; our daily habits are becoming increasingly dependent from connectivity, thanks to the proliferation of smart devices capable of various useful functions, above all the real-time collection of information. This way we are able to retrieve information about the nearest points of interest, starting from our current position, or the estimated time needed to reach a particular address. This has been possible thanks to the presence of sensors of position (Global Positioning System, GPS) and on line cartographic maps: in each time instant it is possible to consult these maps and retrieve any kind of information. Just to give an idea of how these devices will influence our lives in the near future, Berg Insight, a well-known IT company operating in the telecom industry, in its Market Research [Lim09] forecasts that global shipments of GPS-enabled GSM/WCDMA handsets will reach 940 million units in 2015, WLAN handsets shipments will reach 900 million units in 2015 and shipments of NFC-enabled handset will increase from less than two million units in 2010 to 400 million units in 2015.

Due to the spread of these devices it is possible, in a very convenient and inexpensive manner, to collect the data generated by the GPS about various moving objects, like humans, animals, vehicles, etc; in addition, the availability of other sensor-tracking techniques like GSM, radar, WiFi and RFID, contribute to capture huge amounts of trajectory data, so increasing the available functions and giving raise to brand new functionalities.

Motivations The above consideration clear how technology has favored the born of a new branch of applications, location-based services (LBS), which are a general class of computer program-level services used to include specific controls for location and time data as control features in computer programs. The key element of LBS is the information, which is obtained from mobile devices through the mobile network and is based on the geographic position of the device. LBS are used in a wide variety of contexts, such as health, indoor object search, entertainment, social networking, navigation assistants, and so on. One of the most interesting features provided to the user is the possibility to discover the nearest point of interest or the position of other objects in real-time; for instance, one could be interested in discovering the closest open pharmacy, the shortest way to reach a given address or even the whereabouts of a friend of him. The use of other technologies, like RFID, can aid in increasing already existing services, like parcel tracking systems, so providing additional features for mobile commerce.

Location-Based Services Despite having been first introduced for synchronization purposes based on the position, Location-Based Services have evolved to complex tools for control and policy systems in computers. The wide variety of application contexts, ranging from traffic flow control systems to smart weapons, make LBS one of the most used application-layer decision framework in computing today.

Location Based Services can be seen as the convergence of three different technologies in one single device: mobile internet access, positioning and rich user interfaces. All these technologies have only been made available to a large audience in the last years; until the late nineties, in fact, the devices only supported voice and SMS and were characterized by a small number of user interface capabilities. It was the introduction of WAP and internet access in mobile phone that favored the expansion of the Location Based Services.

LBS History The very first LBS-equipped mobile device was probably the Palm VII, produced by 3Com in 1999; some of the provided funtionalities were the weather application from The Weather Channel and the TrafficTouch application, which used the ZIP code-level positioning information.

A few years later, in 2001, TeliaSonera in Sweden, EMT in Estonia,swisscom in Switzerland, Vodafone in Germany, Orange in Portugal and Pelephone in Israel launched the first LBS services, which included FriendFinder, yellow pages, emergency call location, TV game, etc.

The first commercial LBS service in Japan, launched by DoCoMo, was based on triangulation for pre-GPS handsets, while the first mobile phones equipped with GPS were launched by KDDI in 2001.

In May 2002 the US companies go2 and AT&T launched the first mobile LBS local search application: users were able to determine their location and obtain the position of the nearest facilities of their interest, for example stores, restaurants, etc, ordered according to their proximity to the user's position.

LBS Application Contexts The concomitant use of GPS, GSM, radar, WiFi and RFID technologies allows the use of smart mobile devices for various applications in many different scenarios, from traffic monitoring and environmental management, to satellite navigation and geo-social networks. Some of these scenarios are, for instance:

1. tracking of animals trajectories for behavioral analysis of wild life. Animals have been implanted GPS chips in order to keep track of their displacements.

2. location-based services for mobile phones in social networks, such as Google Latitude, Foursquare, Facebook Place, Gowalla, Twitter, etc.

3. improvement of e-business quality of service thanks to the installation of RFID.

4. Real-time traffic analysis and streets congestion avoidance through installation of GPS devices on vehicles.

All the above application contexts deal with the possession of a sequence of spatio-temporal data concerning the position of an entity, may be it an individual, an animal, a vehicle, or anything else. The increasing presence of trajectories in our world spurs to think to applications more and more complex which, also thanks to the full interconnectivity reached in these years, makes the range of possible applications based on the trajectories almost infinite.

Video Surveillance The primary target of a Governmental Institution is to guarantee the security of its citizens against the threats; this is accomplished in many ways, according to the culture and the history of a Country. The investments in the security field have gained a strategic importance after the events of September 11th 2001 in New York, March 11th 2004 in Madrid and July 7th 2005 in London, which involved the targeting of fundamental infrastructures for the financial and the transportation field, which have always played a crucial role in our everyday life.

Since this breakpoint technology has been enslaved to security requirements and, as a matter of fact, law enforcement and police can count on various instruments which make them able to ease and enhance their investigations and surveillance tasks. For example, access control has been strengthened through the use of innovative metal detector and body scanner, which allow to quickly detect the presence of arms or suspicious objects transported by individuals; moreover, biometric systems permit personal identification and access control, making it far more difficult for unauthorized personnel to gain access to restricted areas. In addition, the contextual use of different technologies and data coming from different sensing devices has enriched the effectiveness of security technologies; this is the case, for instance, of the ambitious project Blue Crush [Vla12], which is being experimented in Memphis: through the use of proper *data mining* procedures a huge amount of data is extracted from the historical archives and

the reports of the police departments and analyzed so to detect, with good approximation, the typology of violation and the area involved which are more likely to happen in the aftermath.

Thanks to reduced installation costs and high efficiency, surveillance systems are being widely used as monitoring instruments; moreover, the research advance in *computer vision* has contributed to the born of a new generation of surveillance systems, able to overcome to the previous limitations also thanks to the use of *behavior recognition*.

Figure 1 Cameras are everywhere in our daily life.

Apart from individual security and integrity, videosurveillance has also great interest from the environmental perspective, which is connected with various aspects of our daily life:

- Production and commercial industry, in which the safety of the personnel and the customers represents a prerogative in relation to the crime acts like theft and robbery.

- Intelligence and Military fields, which need environmental control as a protective instrument towards sensitive and reserved areas and information with respect to terrorism and espionage.

- Urban and metropolitan transports, which include the security of airports, metropolitan and railway stations, in which

the great daily flow of passengers involves various risks for the users and the working personnel.

- Preservation of the historical-cultural heritage from urban vandalism acts.
- Institutional and financial branches of public administration, whose strategic importance makes them a target for those who aim to disturb the civil peace and the sensitivity of the community; this is the case of the 9-11 events, for instance.
- Cultural, sport and religious events, like festival, concerts, meetings and so on.
- Public areas of meeting or assembly for the citizenry.

The increasing need for security in public and private areas, the significant reduction of the costs of the acquisition peripherals and the progress in the techniques of digital image and signal processing have strongly contributed to the spread of *computer vision* techniques in monitoring and surveillance applications, also through the development of remote controlled systems for private areas in the brand new field of Home Automation.

Surveillance systems can be historically distinguished according to the operational mode: Registration and Passive. Registration systems convey the flows of the cameras in a monitoring central station for the storage; passive systems, besides storing the captured video sequences also display this flows in real-time on video benches to be monitored by human operators, who accomplish to properly intervene in case of security alerts or anomalous behaviors. The technical characteristics of both types of systems make them unsuitable for crime prevention and on-line surveillance of complex areas, therefore passive monitoring systems are usually used for post-facto analysis. A registration surveillance system often acts as a dissuasive device, so permitting to reduce the percentage of some typologies of crimes; a passive system, in principle, allows to promptly detect suspicious behaviors or crime acts.

Traffic Flow Analysis Among the various application contexts of computer vision and monitoring systems, a strong interest is devoted to traffic scenarios, which are daily interested by accidents, traffic queues, highway code violations, driving in the wrong lane or on the wrong side, and so on. These events characterize our daily life and, as a consequence, proper strategies must be implemented in order to correctly manage the risks connected with them.

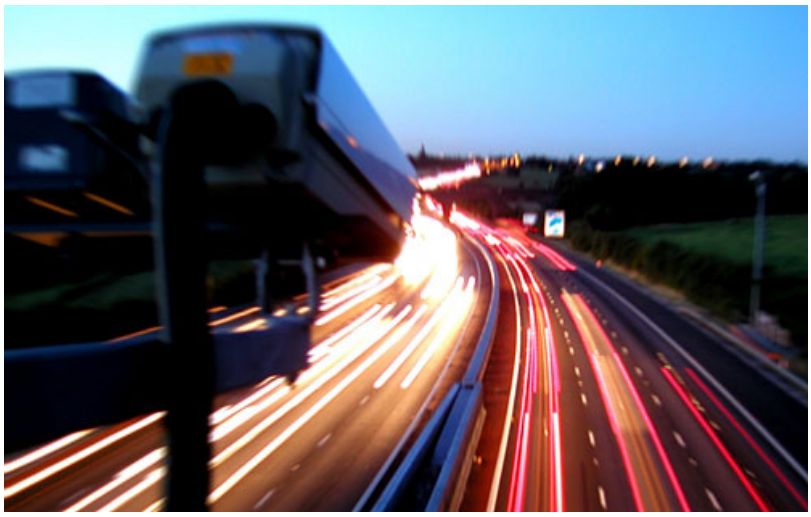


Figure 2 Highways are more and more monitored with cameras.

The technological progresses have deeply modified the way traffic control is performed, so easing the work from many perspectives; by the way, the scientific research is still ongoing when dealing with the real-time analysis of the video streams. Apart from the various different sensors which have been introduced to enhance the traffic control systems, cameras represent a suitable solution for their relative low cost of maintenance and the possibility of installing them virtually everywhere; in fact, our highways, public squares and parking areas are being more and more endowed by security cameras. This leads to the storage of a huge amount of data, which can be profitably used to increase the security of our infrastructures.

Limitations and well-known problems Given the presence of a growing number of security cameras, one of the greatest limitations for control and management systems is the storage of these data. It must be noticed that one single scene can be controlled by different cameras, each of which collects at least 25 frames every second; depending on the context in which an activity is performed, a sequence of interest can last from a few seconds to some minutes; in each frame various objects can be interested by the activity of interest; this implies that the amount of data to be stored is significant and storage and indexing procedures must be adopted.

Figure 3 An instance of surveillance operator monitoring a number of areas.

When thinking of the security of a building, a bank or an airport, we could easily imagine a human operator sitting in front of a video-wall composed of many monitors, each of which displays the view of a different camera (see figure 3 for example). As it is impossible to imagine to have a human operator for each of the cameras, it is likewise difficult for a human being to devote the same attention to a great number of monitors for a long time period. Recent studies, in fact, conducted by the US National Institute of Justice, have revealed that the attention threshold of

a human operator collapses below an acceptable value after only 20 minutes; in addition, ASIS, a very important organization for security professionals, has confirmed that an operator can monitor a number of monitors between 9 and 12 for no more than 15 minutes[GF09]. These observations imply a minor capacity of detection and intervention, which is further reduced in case of complex and/or crowded scenes, for the presence of a higher number of zones and objects.

The above consideration lead to the employment of systems capable of detecting, in a semi-automatic fashion, the events of interest, so that to ease the task of the human operator, which will be in charge of taking the final decision. In the Video-Surveillance context, for instance, have been recently introduced *Active Surveillance Systems*, which include a computer analysis module which supports the human operator and, in case of suspicious events, it points out its attention only to the interested area. This is possible through the use of computer vision techniques, which allow to give a semantic interpretation to the depicted scene. These systems, however, will not be charged of the complete surveillance process, as the task of evaluating the collected data and solving possible ambiguities will always be played by humans. To better understand this concept, think about a suitcase left unattended in the middle of an airport hall: this situation could be possibly dangerous, depending on the path of the person who as left it, the number of people present and their identity, the time it will remain unattended and, of course, its content. All these factors cannot be taken into consideration by au automatic system due to its incapacity of abstraction and contextualization.

Chapter 1

Human Behavior Analysis

The automatic detection of motion patterns in video sequences has gained great importance due to many fundamental reasons, the first one of which is the huge number of its potential applications. The technological progress has made it available more and more sophisticated acquisition peripherals. While the quality of the acquired images has definitely improved, permitting good quality compromises at low/affordable costs, the requirements for video analysis have also increased, so that automatic/supervised video images understanding still remains an open problem. This is attributable to the inherent complexity of these class of problems, most of which are ill-posed: just think about the classic case of detecting the pose and the motion of a complex articulated and self-occluding non-rigid object from video images.

Another factor that has contributed in giving greater importance to motion analysis in video sequences is the massive presence of cameras (mostly security cameras) installed in public areas, which means we have lots of video streams acquired from real urban scene, but we still are not able to fully interpret them. When talking about automatic interpretation of the scene, we refer to the possibility of having algorithms (usually Artificial Intelligence Algorithms) capable of recognizing the occurrence of particular patterns in difficult conditions. If we consider, as an example, hand gesture recognition, we refer to the capability of detecting

and recognizing hand movements, associate them to the corresponding person and giving them the corresponding semantic significance. While the first two operations can be accomplished with any of the available methods in literature, most of which are statistical, the semantic interpretation of patterns still represents a challenging task.

The interpretation task becomes more interesting but even more difficult when handling motion patterns referring to human people: decision-theoretic (statistical) approaches provide good facilities to correctly detect and recognize human movements, but we still need methods to associate them a semantic interpretation.

1.1 Image Analysis

When referring to a system which is aimed at the detection and recognition of human movements in video images, we can generally refer to it as a Human Motion Capture and Analysis System. This is a very general purpose system, which can be applied to many application contexts, that we can divide into three macro-categories: Video-surveillance, Control and Analysis [MHK06]. *Control applications* concern the recognition of gestures, pose and facial expression, aimed at improving the communication between human users and computers (HCI, Human Computer Interfaces): this is the case of gesture-driven controls. Virtual Reality, Teleconferencing, Character animation and motion synthesis applications are also part of this category. *Analysis* category deals with human body parts segmentation and body structure analysis for diagnosis purposes; in this category we can mention medical diagnosis and treatment support, biomedical studies of athletes (sports), personalized training systems and, more recently, car industry applications (like sleeping driver detection and pedestrian detection). The *Surveillance* macro-category is probably the most interesting, studied and difficult one: unlike the case of Control and Analysis, the image sequences are not taken from a controlled environment, but usually refer to outdoor scenes, in which we can have

crowded scenarios with many occurrences of well-known problems like shading, waving trees, occlusions, collisions, lighting changes and so on. Newer applications address Human Behavior Analysis, the detection of abnormal or critical situations connected to human actions in crowded areas: in this cases human motion capture is, of course, an inescapable basis but, in addition, semantic interpretation is needed.

Recently it has been possible to address natural outdoor scenes and operate on long video sequences in which are present multiple (occluded) people, thanks to the introduction of more advanced segmentation algorithms and, as mentioned before, the cheapness of high quality hardware.

A brand new area that is attracting interest is Facial Expression Recognition. The first attempts of facial recognition were made long time ago, and results in this field have improved a lot since that time; nowadays researchers would like to achieve a new goal: the recognition of human feelings through facial expression recognition. This new interest has also caused some ethical objections concerning privacy and a open discussion is still in progress. The attempt to recognize human feelings can be attributed to a new branch called Affective Computing, thanks to the work of Rosalind W. Picard at MIT Media Laboratory.

1.2 Object and Motion Detection

In order to accomplish the difficult task of recognizing human actions, some preprocessing steps, related to the detection of the human figure and its motion, are of course needed; these preprocessing steps can be generalized to the detection and identification of a given target aimed at keeping trace of its movements.

The *Object Detection* phase permits to distinguish the moving objects, belonging to the foreground, from the background within the input frames; once the set of pixels belonging to a moving object (called "blob") has been detected, it will be possible to keep trace of the blob's movements in time (tracking).

The operation of separating the objects of interest from the background can be subdivided into two different levels:

- The *Pixel Processing* Level is a low level providing binary information about the estimation of the background; this is possible thanks to the application of different algorithms which evaluate the membership of each pixel to the background or to a given object. The noise in the image, induced by the acquisition system, can be weakened in this phase through filtering and calibration process.

- The *Frame Processing* Level represents the higher step in the detection phase; it is devoted to the refinement of the results obtained at the previous level, which means that the binary image is analyzed by taking into account the relationships between the pixel belonging to the background and/or the objects. The final aim of this process is *blob segmentation*, which means to obtain a classification of the pixels in relation to their belonging to different objects.

Video segmentation represents a key operation within content-based video analysis, multimedia content description and intelligent signal analysis; therefore it is fundamental that the results obtained in this phase are less noisy as possible in order to avoid wrong pixel classification. In this regard a *false positive* is a pixel which has been erroneously labeled as belonging to an object, while a *false negative* is a pixel which has been erroneously labeled as background. Some of the phenomena which can lead to this kind of errors, are:

- *Waving Trees*, that is events of relevant motion which do not have interest from the application perspective (for instance, motion of the trees' fronds);

- *Camouflage* of objects with the background due to their chromatic characteristics;

- *Light changes*, that is each generic variation of the lighting conditions of the depicted scene;

- *Foreground aperture*, which denoted the possible difficulty in detecting the moving objects which possess uniform color.

In order to process the pixels contained in the input images two different methods can be used:

- *Derivative* algorithms;
- *Background comparison* algorithms.

1.2.1 Algorithms based on the difference between frames (derivative)

Derivative algorithms allow the detection of moving objects within a video sequence by only analyzing the differences between subsequent frames. Therefore their output only highlights those objects which have undergone a significant variation in intensity values. In particular, it is possible to distinguish between *single difference* and *double difference* algorithms, which analyze, respectively, the last two or three frames of the video sequence. Though single difference algorithms are based on a procedural logic which can be easily implemented, their results are often influenced by disturbance phenomena like foreground aperture, ghosting, and so on. The adoption of double difference algorithms can overcome the previous limitations through the following operations:

1. repeated application of the single difference algorithm on two consecutive frame couples;
2. binarization with threshold;
3. AND operation pixel-by-pixel between the two generated binary images.

As the above described procedure includes simple operations, this class of algorithms has a low computational cost and a substantial insensitivity to gradual lighting changes, which represents their strength.

1.2.2 Algorithms based on the comparison with a background model

The basic operation of the algorithms in this category is the comparison of the current frame with a reference background; their output, differently from the derivative algorithms, indicates their complete independence from the object's motion and usually solves some well-known problems (like foreground aperture). One crucial point of these algorithms is the frequent update of the reference background, which is an expensive operations from a computation point of view; in fact, a generic background comparison algorithm is constituted by a background subtraction module, which aims at extracting the objects of interest from the current image through the comparison with the background model and a background update module, which is in charge of frequently updating the background model starting from the output of the previous iteration.

Within this category, we can further distinguish between *statistical algorithms*, which generate statistics for each pixel in order to give some interpretation of the scene dynamics according to the chosen statistical model, and *reference image* algorithms, which use as a reference the image background without any object in it.

1.3 Object Tracking

The concept of Tracking in visual analysis of human motion involves two inherent notions: figure-ground segmentation, which is the process of separating the objects of interest from the rest of the image (background) and temporal correspondences, which is the process of associating the detected figure in the current frame with the same figure in the previous frames.

Starting from the results obtained from the previous (Motion Detection) phase, the *Object Tracking* phase first encodes the frames flow in order to consider the blocks of pixels as moving objects' blobs, where the blob can be seen as a collection of object's pixels. The main operation consists in finding in each sub-

sequent frame the blob generated by the same object and creating an association between these blobs; the capability of the system of correctly and efficiently creating this association will denote the efficiency of the entire tracking process.

From an operational perspective, different motion regions can correspond to different moving objects; for this reason, it is of fundamental importance to correctly classify these objects. The object classification process within the tracking phase can be performed through two different modalities: recognition-based and motion-based.

Recognition-based techniques consist in the recognition of the object within the sequence of consecutive images and the extraction of its position. The main advantage of these algorithms is their three-dimensional application and the possibility to estimate the object's rotation and translation, while their main disadvantage is the inability to track objects which differ from the ones searched; this limitation entails a dependence of the performance from the recognition method in terms of computational complexity.

Motion-based techniques, on the contrary, aim to detect the object according to the estimation of its motion; a further level of classification can be adopted:

- region-based;
- feature-based;
- active contour-based;
- probabilistic approach;
- hybrid approach.

Region-based approaches aim to associate each object with a blob by using measurements of correlation between blobs in consecutive frames. These methods prove to be particularly efficient in case of object "adequately distant" form each other; the proximity factor, in fact, could strongly influence the results giving raise to occlusion phenomena.

Feature-based techniques focus the attention to some given object's points belonging to its morphological structure (features); this allows to overcome the occlusion problems of region-based approaches, as features tend to remain constant even in presence of occlusions or lighting changes. It must be noticed, however, that a crucial point is the choice of the set of features: there is a trade-off in the number of the features, as a small number would constitute a scarce representative set, with consequent loss of robustness, while an higher number of features would entail a higher computational cost.

Active contour-based approaches shoot for building a representation of the object's contour which is constantly updated; these methods are still affected by occlusions, although in minor form.

Probabilistic approaches are based on a stochastic representation of the object; the object's behavior is represented through a set of states and transitions with given likelihood. These methods permit to understand aspects whose modeling is very difficult; by the way, for each transition we must avoid the "escape to infinity" problem, that is the progressive move away from the correct state of the object.

The employment of a *hybrid approach* permit to exploit the advantages of different object tracking methods and to moderately overcome their limitations.

1.4 Activity Recognition

The field of activity recognition is relatively old but still immature; the approaches which have been presented strongly depend on the application context (surveillance, medical studies and rehabilitation, animation, etc.). A first difference can be spotted between holistic approaches, that take into account the entire human body, and local approaches, that only consider particular body parts. A more accurate subdivision of the approaches is:

- scene interpretation, which attempts to interpret the entire scene without identifying particular objects or humans;

- holistic recognition, in which the entire human body is applied for recognition;

- action primitives and grammars, where an action hierarchy gives rise to a semantic description of a scene.

When aiming at describing the actions performed by one or more objects in the scene, it is useful to construct a hierarchical structure of the behaviors; an action hierarchy is constituted by: Action primitives or motor primitives, used for atomic entities, out of which actions are built; Actions, sequences of action primitives that are, in turn, decomposed in activities; Activities, larger scale events that typically depend on the context of environment, objects or interacting humans. This approach is fundamental in syntactic approaches.

1.4.1 Scene Interpretation

In reasonable situations, where the objects are small enough to be represented as points on a 2D plane, scene interpretation is about attempting to learn and recognize activities by the simple observation of the objects motion, without necessarily knowing their identity. Various approaches have been presented, among which:

- Stauffer et al. [SG00a], who presented a system for the full scene interpretation aimed to the detection of unusual situations. The system extracts some features as 2D position, speed, size and binary silhouette; subsequently, using co-occurrence statistics and a binary tree structure for two probability mass functions, some simple scene activities to be detected are defined, like pedestrian and car motion.

- In [ETK+03] a swimming pool surveillance system is presented: for each of the tracked objects the system extracts features relating to speed, posture, submersion index, an activity index and a splash index; these features are then fed

into a multivariate polynomial network in order to detect water crisis events.

- An interesting approach is presented in [BI05a] aiming to the detection of irregularities in a scene by extracting small images and video patches, which are then used as local descriptors; in an inference process, the system searches for patches with similar geometric configuration and appearance properties, efficiently inferring the even imperceptible but significant local changes in behavior.

- Some approaches based on the use of shapes, like the work in [CC03],[VRCC03], in which the activity trajectory is modeled by non-rigid shapes and a dynamic model characterizes the variations in the shape structure.

1.4.2 Holistic Recognition

The identity recognition of a human has been widely discussed; some approaches are based on gait recognition, while some others concern the recognition of simple actions such as running or walking. Almost all methods are holistic: the entire silhouette or contour is considered without detecting individual body parts. Among the human-based recognition of identity systems the following papers are worth mentioning:

- In [WTNH03] Wang et al. present a system that performs the following actions: computation of an individual's silhouette by contour sampling, computation of the distance between each contour point and its center of gravity, PCA processing on the unwrapped contour and finally trajectories comparison. In spite of its simplicity, the system gives good results on outdoor data and is computationally efficient.

- In [FNPB03] Foster et al. present an approach for the automatic gait recognition; the system uses binary masking functions to measure area as a time varying signal from a sequence of silhouettes extracted from a walking subject.

Fisher analysis is applied and finally the k-nearest neighbor classifier is used for classification.

- In [KSR+04] an HMM is used to model the dynamics of individual gait: the HMM is trained for each individual in the database; five representative silhouettes are used as the hidden state for which transition probabilities and observation likelihoods are trained. During the recognition process, the HMM with the largest probability identifies the individual.

Another category of holistic approaches is based on the use of dynamics to recognize what individuals are doing rather than who they are. An interesting approach in [RR05] attempts to understand actions by building a hierarchical system based on reasoning with belief networks and HMMs at the highest level, while at the lowest level it uses features such as position and velocity as action descriptors (it outputs qualitative information such as walking left-to-right on the sidewalk).

A pioneering idea is that of temporal templates: a representation based on Motion Energy Images (MEI) and Motion History Images (MHI). The MEI is a binary cumulative motion image. The MHI is an enhancement of the MEI where the pixel intensities are a function of the motion history at that pixel. Matching temporal templates is based on Hu moments. Bradski et al. [BD02] pick up the idea of MHI and develop timed MHI (tMHI) for motion segmentation. tMHI allow determination of the normal optical flow: motion is segmented according to object boundaries and motion orientation; Hu moments are applied to the binary silhouette to recognize the pose. This approach has given rise to new works based on it.

Instead of using spatio-temporal volumes, a large number of papers takes into account the sequences of silhouettes. Yu et al. in [YmSxSL05] present an approach that: extracts silhouettes, whose contours are unwrapped and processed by PCA; successively, a three-layer feed forward network is used to distinguish "walking", "running" and "other" on the basis of the trajectories in eigenspace. In [CCK02] Cheng et al. attempt to distinguish

walking from running on the basis of sport event video data. The data come from real-life programs. They compute a dense motion field; foreground segmentation is performed based on color and motion. Within the foreground region, the mean motion magnitude between frames is computed over time, followed by an analysis in frequency space to compute a characteristic frequency. A Gaussian classifier is finally used for classification.

Recognition based on body parts is about trying to recognize actions on the basis of their dynamics and settings. Wang et al. [WNTH04] present an approach where contours are extracted and a mean contour is computed to represent the static contour information. Dynamic information is extracted by using a detailed model composed of 14 rigid body parts, each one of which is represented by a truncated cone. Particle filtering is used to compute the likelihood of a pose given an input image. For the classification task, a nearest neighbor classifier is used.

Sheikh et al. [SSS05] argue that the three most important sources of variability in the task of recognizing actions come from variations in viewpoint, execution rate, and anthropometry of the actors. Based on this, they state that the variability associated with the execution of an action can be closely approximated by a linear combination of action bases in joint spatio-temporal space.

1.4.3 Action Primitives and Grammars

There is strong neurobiological evidence that human actions and activities are directly connected to the motor control of the human body. The neurobiological representation for visually perceived, learned, and recognized actions appears to be the same as the one used to drive the motor control of the body. These findings have gained considerable attention from the robotics community.

In Imitation Learning the goal is to develop a robot system that is able to relate perceived actions to its own motor control in order to learn and to later recognize and perform the demonstrated actions. Consequently, there is ongoing research to identify a set of motor primitives that allow:

1. representation of the visually perceived action;

2. motor control for imitation.

As a consequence, this gives rise to the idea of interpreting and recognizing activities in a video scene through a hierarchy of primitives, simple actions and activities. Many approaches are aimed at decoupling actions into action primitives and interpreting actions as compositions on the alphabet of these action primitives; once the primitives have been detected, an iterative approach is used to find the sequence of primitives for a novel action.

Modeling of activities on a semantic level has been addressed by Park and Aggarwal [PA04b]. The system they describe has 3 abstraction levels: at the first level human body parts are detected using a Bayesian network; at the second level DBNs (Dynamic Bayesian Networks) are used to model the actions of a single person; finally, at the highest level, the results from the second level are used to identify the interactions between individuals.

Ivanov and Bobick [IB00] propose the use of stochastic parsing for a semantic representation of an action. They argue that, for some activities where it comes to semantic or temporal ambiguities or insufficient data, stochastic approaches may be insufficient to model complex actions and activities. Therefore they suggest to decouple the actions into primitive components and use a stochastic parser for recognition, picking up a work by Stolcke [WSH04] on syntactic parsing in speech recognition and enhancing this work for activity recognition in video streams.

In [YY05] Yu and Yang present an approach in which they use neural networks to find primitives. They apply self-organizing maps (SOMs, Kohonen's feature maps) for the training images clustering based on shape feature data. After having trained the SOMs, a label is generated for each input image in order to convert the input image sequence into a sequence of labels. A subsequent clustering algorithm allows to find repeatedly appearing substructures in these label sequences. These substructures are finally interpreted as motion primitives.

A very interesting approach is presented by Lv and Nevatia

in [LN06], in which the authors are interested in recognizing and segmenting full-body human action. Lv and Nevatia propose to decompose the large joint space into a set feature space where each feature corresponds either to a single joint or to a combinations of related joints. The recognition of each class action according to the features is achieved through the use of HMMs and, finally, an AdaBoost scheme is employed to detect and recognize these features.

An important differentiation must be done for the methods addressing the interpretation of actions explicitly considered as regarding humans; in this approaches, a large attention is devoted to rather simple actions such as walking, running and simple hand gesturing: a good understanding of these simple actions is necessary before they can be combined into more complex ones.

According to the considerations exposed and the methods presented in this chapter, it is clear that *Human Behavior Analysis* is a wide and very interesting field and, even though scientific researched has addressed this branch from some years, there is still a lot to do for the complexity of the subject. In next chapter I will discuss about one of the methods for the extraction of information about the behavior: the analysis of trajectory.

Chapter 2

Analysis of Spatio-Temporal Information

The characterization of the behavior of the objects moving in a scene starting from the analysis of the video sequences is a very difficult task, as a full comprehension need to account for various issues like the target's intention, the environmental context, the interactions with the other moving objects, the scene obstacles, and so on. We can divide all these features into low-level and high-level features: while low-level characteristics attain to the target's physical properties, like its size or its shape, high-level ones are related to information concerning the scene topology, the socio-psychological attitude, etc. The object's trajectory represents a medium level feature as it expresses from one side the spatial displacements of a moving object and, from the other side, it gives information about the object's dynamics on the scene.

The trajectories of the objects moving in the scene are obtained from the tracking phase, which is responsible for detecting moving objects, unequivocally identifying them and collecting the information about their motion characteristics in the scene. It must be observed, however, that the extraction of the trajectory is a very complex task due to many factors: occlusions and collision, scene

obstacles, low bitrate; in presence of such problems, the obtained trajectories are usually incomplete or noisy. An analysis of trajectory data affected by tracking errors would involve errors also in the recognition phase; in order to avoid this, some procedures are needed before the trajectories can be analyzed.

2.1 Trajectory Representation

One of the main problem when handling spatio-temporal data is their time-varying nature, which leads to trajectories with unequal length. In order to ensure effective comparison between trajectories with different lengths, different techniques have been proposed; alternatively, some other methods aim at modifying a set of trajectories so that to obtain trajectories with equal length.

Starting from an input set of raw trajectory data, a normalization procedure aims at processing each trajectory so that all trajectories will have the same size. The most simple approaches for trajectory normalization are:

1. zero padding [HXF+06] - given the input trajectory set S_T the prototypical size $\hat{T}$ is chosen to be equal to the size of the longest trajectory. All other trajectory, which are obviously shorter than $\hat{T}$ are added extra point values equal to zero so that to match the prototypical size.

2. track extension methods, similarly to zero padding, consist in adding some points at the end of a trajectory in order to obtain uniform lengths: in this case, however, rather than zeros, the points added are estimated by using the trajectory's dynamics at the last tracking time [HTWM04] .

3. Resampling and smoothing techniques - the resampling operation, aimed at obtaining same trajectories' lengths by interpolating the input raw trajectory data, can be performed in different ways: linear interpolation [ME02], [BCB03], [MT08a], subsampling [HXF+07], Douglas-Pecker algorithm [LCST06].

The smoothing of trajectories is mainly implied for the reduction of noise by applying simple filters [JJS04], [BQKS05] or through wavelets [LHH06].

Even being very simple and computationally efficient procedures, normalization methods need to operate on sets of complete trajectories, which is usually unfeasible in live tracking.

The dimensionality reduction of trajectory data is aimed at mapping the input data into a lower dimensional space for a greater manageability; the new space is obtained by defining a trajectory model and calculating the parameters that best describe this model.

Vector quantization reduction has been widely used to obtain a finite set of prototypical vectors which symbolize all the input data [SG99], [ZSV04].

2.1.1 Polygonal Approximation

Another widely investigated approach is polygonal approximation, which includes shape and contour representation [PCDN09]. These methods essentially consist in the interpolation of the input spatio-temporal data with a piecewise linear curve having a finite number of vertices; this will lead to a compact and still significant representation, as most of the informative content of trajectories lies in the points of maximum curvature. Some methods only rely on spatial coordinates and assume trajectories are simple bi-dimensional curves; the signal is approximated by a least-squares polynomial [MMZ05], [YF05]. The aim of any polygonal approximation method, indeed, is twofold: minimization of the error, that is finding the polygon that best fits the original curve, and minimization of the number of dominant points. These optimization problems can be solved, for instance, by using the compression ratio criteria $CR = N/M$ and the integral square error between the vertices of the trajectory and the linear segments of the polygonal curve [Ros97]. As the solution to the above two optimization problems implies a high computational cost (of the order of $O(N^2)$

or even $O(N^3)$), some authors have proposed to focus only on one of the two problems [CC92].

2.1.2 Spline Approximation

Another method for efficiently representing a trajectory is to find smooth curves, called splines [MY86], interpolating a set of points (called nodes) in a given interval, so that this curve is continuous in each point of the interval. Spline functions have been applied in many contexts; one of the simplest functions, the *Bezier functions*, have been successfully applied in mechanical design. In order to interpolate a set of points, a generalization of Bezier functions has been widely adopted, the *B-splines*:

$$A(u) = \sum_{i=0}^{n} N_{i,d}(u) p_i.$$

where p_i denotes the control points and $N_{i,d}(u)$ represent the spline basis functions of order d. The control points are nothing but the original sequence of points representing the spatial displacements of the moving object, that is the original trajectory. Without giving into mathematical details, the spline approximation entails the calculation of the optimal coefficient of the polynomial model fitting the control points.

As spline approximation can be seen as a subset of polynomial approximation, we still need to keep the degree of the polynomial low in order to avoid oscillations and to guarantee smoothness. One of the main advantages of using splines for the approximation of the trajectories obtained from the tracking phase is its invariance to affine transformations and its resilience to tracking errors.

2.1.3 PCA Coefficients

Principal Component Analysis (PCA) is a mathematical procedure converting a set of observations of (correlated) variables into another set of values of linearly uncorrelated variable, which are

called *principal components*, by means of orthogonal transformations. Being formulated in 1901 by Karl Pearson [Pea01], PCA has been successfully applied to an enormous variety of application contexts and numerous variations of it have been proposed. PCA-based methods belong to dimensionality reduction category as they aim to reduce the dimensionality of the input data by analyzing samples covariance.

Its application to spatio-temporal representation is quite immediate: starting from the input trajectory T, constituted by a set of random variables and a correlation matrix C, we can find the $k-th$ principal component through orthonormal linear transformation:

$$PC_k = a_k T.$$

where a_k is an eigenvector of the correlation matrix C which corresponds to its $k-th$ largest eigenvalue λ_k.

2.2 Trajectory Clustering

In order to identify the structure of the input spatio-temporal data, a well-known machine learning method is widely used in literature: Clustering. The main aim of this method is the grouping of the collected trajectories into categories which share a common property or which are "similar" according to a specific metric. Therefore it is needed, for each clustering procedure, to address the following three issues:

- Similarity metric
- Clustering procedures
- Cluster Validation

While a complete survey of clustering techniques for time-series data is presented in [WL05], we will here concisely discuss about the above three issues.

2.2.1 Similarity Metric

As the goal of a clustering techniques is the creation of categories sharing common properties, the definition of a proper and effective distance (or similarity) metric is essential. One of the issues which must be addressed when choosing a similarity measure is the length of trajectories: if, in fact, the possessed data has not undergone any pre-processing step aimed at normalizing trajectories' lengths, we need to choose a metric which is independent from the length parameter.

The basic distance measure for trajectories is, of course, the Euclidean distance; given the trajectories T_i and T_j of equal size, their distance is computed as

$$d_E(T_i, T_j) = \sqrt{(T_i - T_j)^T (T_i - T_j)}.$$

In case we have trajectories which do not share equal sizes, we can then use a size-invariant version of the Euclidean distance: given the two vectors T_i and T_j of sizes m and n respectively, the distance is given by [BI05b]

$$\overline{d}(T_i, T_j) = \frac{1}{m}\Big(\sum_{k=1}^{n} d_E(T_{i,k}, T_{j,k}) + \sum_{k=1}^{m-n} d_E(T_{i,n}, T_{j,n})\Big)$$

where $T_{j,n}$ is the last point used to accumulate distortion.

Even being very simple to be computed, the Euclidean distance is not so effective in most cases: in fact it is really efficient when the input trajectories are aligned. Therefore a pre-processing step of trajectory alignment would significantly improve the efficiency of the distance metric. Among the different distance metrics for unequal length signals, *Dynamic Time Warping (DTW)* has been widely adopted, especially for speech recognition issues; its main idea is to find an optimal alignment between trajectories by minimizing the distance between matched points [RJ93]. Another well-known alignment technique is *Longest Common SubSequence (LCSS)*, which aims at discard outliers and is resilient to noise[VKG02].

Another widely used distance metric is the Hausdorff distance, which measures the distance between two unordered sets [JJS04]; for its limitation in not considering the ordering, modified versions of it have been proposed [AMP06].

2.2.2 Clustering Procedures

Once defined the similarity measure we must establish a clustering procedure for grouping the trajectories into similar sets; these sets are also called clusters of routes. The route learning phase can be performed according to different approaches:

1. Iterative optimization, which is the most popular clustering techniques due to its simplicity and tractability. Its basic version makes use of Euclidean distance, while more efficient versions have also been presented, such as standard K-means [BKS07] or fuzzy C means [MT08a].

2. Online adaptive methods, which, differently from iterative methods, do not need huge amount of training data and the number of tracks must not be known a priori [PF06]. Online adaptive techniques are particularly effective for time-varying scenes as clusters are frequently updated.

3. Hierarchical approaches, which are divided between agglomerative [BAT05] and divisive [JJS04] according to the procedure of the tree-like structure defining the similarity relationships between trajectories (bottom-up for agglomerative, top-down for divisive methods).

4. Neural Networks, for instance the use of Kohonen's Self Organizing Maps (SOMs) [Koh90] for clustering the trajectories in a low-dimensional space but still preserving their topological properties. One of the possible disadvantages of this method is the long convergence time.

5. Co-Occurrence decomposition, successfully adopted for document retrieval and according to which trajectories are considered as bags of words with the following criteria: similar

bags contain similar words [XG05]. Therefore, starting from the training data, the resulting co-occurrence matrix is decomposed to obtain trajectories routes.

2.2.3 Cluster Validation

As the real number of clusters (or routes) is unknown, it is needed to perform procedures aimed at verifying the correctness of the learned routes, that is cluster validation. The majority of clustering algorithms requires an initial estimation of the expected number of clusters; as this estimation is very unlikely to be correct in the initial phase, most cluster validation techniques aim at updating this estimation as soon as more information are collected.

Other cluster validation techniques set the expected number of clusters by minimizing (or maximizing) some optimality criterion: this is the case of the Tightness and Separation Criterion (TSC) [HXF+06]; finally, it is possible to validate the clusters by means of information theory criteria, like in the case of Bayesian Information Criterion (BIC) [NK06].

2.3 Model-Based Behavior Analysis

Once we have built a scene model, by using one of the methods existing in literature, the main aim of our system still needs to be addressed: the automatic recognition of the events of interest in the scene. What is crucial, at this point, is what we intend with "interesting", in the sense that the importance of a given action performed in the scene is always dependent on the context in which it occurs. Just to give an example, a person running with something in his hands could be considered as normal in a video sequence depicting a relay race context; this same occurrence should, on the contrary, be considered as anomalous if related to a crowded open-air market scene. This does not mean that it is impossible to design a system capable of automatically recognizing the contextual significance of an event, but it is enough to assess the need for more complex reasoning in such systems.

As a matter of fact, the first proposals in this field were strictly related to a given context, so underlying the difficulty in gifting the system the concept of *abstraction*, that is the extraction of the low-level characteristics of a particular event and their projection into a generic application context. After these first attempts, it appeared clear that it would have been impossible to design a system for each of the required event typologies. Thanks to the evolution in the modeling and analysis techniques, it is today possible to design a system which, thanks to its learning capacities or through inferencing ability, can be profitably used in different contexts.

2.3.1 Access Control

One of the fundamental characteristics of a surveillance system, is, of course, the ability to detect intrusions (see figure 2.1), that is perimeter sentries.

Figure 2.1 Intrusion can pertain both private and commercial areas.

To this aim, it is usually useful to identify some areas in which

the intrusion is more likely to occur: these areas are called monitoring zones or virtual fences. Access control functions is very simply: when such events are detected, the system generates an alert which can be managed in different ways; for example, when the alert of a given area has been generated, the system can activate high resolution PTZ (Pan Tilt Zoom) cameras to obtain more detailed visual information for person recognition [TGH05] or vehicle classification [KGT07]. The monitoring of entry and exit zones has also been used for traffic flow analysis [LY00]. Finally, loitering people [BBS06] [BMPI05] can be recognized by analyzing the time spent by an object stationary within a given monitoring area.

2.3.2 Speed Profiling

As well as considering the spatial information, like in the case of access control events, we can also use the information about objects' dynamics: to this aim it is possible to use vehicles' speed measurements for speeding profiling [PJ07], [JH99] or for stationary vehicles recognition for traffic congestion detection [KKL+05]; other articles have addressed the speed state of every vehicle with respect to daily average measurements [MT08a].

Finally, the continuous measurement of speed values can help constructing a model aimed to the detection of anomalous events [JJS04].

2.3.3 Anomalous Behavior Detection

One of the most desirable characteristics of a surveillance/monitoring system is certainly the recognition of anomalous behaviors: again here comes back the notion of context, which affects the definition of normal/anomalous events. According to the POI/AP framework described in the previous paragraphs, one possibility is to consider as normal all those trajectories "similar" to APs, and label as "anomalous" all those trajectories which differ (according to a pre-defined threshold) from the known models (Activity Paths).

Figure 2.2 A pedestrian walking on the bike lane.

This same concept can be applied using methods differing from the POI/AP framework, like neural networks [OH00], [HTWM04], belief networks, HMMs (Hidden Markov Models), Gaussian velocity and curvature profile [JJS04]. As often occurs when dealing with threshold values, there will be a trade-off between efficiency and effectiveness: in some cases we could aim not to miss any real anomaly at the cost of having more false positives while, in other cases, we would like to avoid an unnecessary big amount of false positives.

Figure 2.3 A vehicle with anomalous trajectory.

Characterization of Interactions between Objects In any context the full comprehension of what really happens must account for the interactions occurring between the objects acting in the scene. This task results very difficult for many reasons, first of all the wide variety of interaction types, the different object shapes and properties, the area of interest, and so on.

One of the key concept in detecting and recognizing objects' interactions concerns the area within which each object feels secure: the personal space. According to this concept borrowed by psychology, it is possible to take into consideration all those events involving objects (people or vehicles) overstepping an object's personal area This measure is of course variable and depends not only on objects' typology but also on environmental and socio-cultural parameters; the personal area can vary, of course, on the basis of the object's shape [KRWS05], but also according to the object's motion [PT07].

The notion of personal area can be abstracted to consider vehicles' collisions at intersections by recognizing the overlap of their bounding boxes [HXX+04], [VMP03] or, more generically, for traffic intersection analysis [KMIS00], [SSL07], [Cha06]. A decisive feature of a system, more important than the detection of collision, would be the ability to prevent these events; this can be still accomplished by considering the vehicles' personal area and the APs or the known motion models.

2.4 Scene Modeling: the POI/AP Framework

As already explained in previous chapters, most visual surveillance still depends on a human operator [MT08b]; the sheer volume of video data coming from the numerous acquisition devices spread in both public and private spaces demands for proper techniques. Therefore computer vision techniques can be adopted to help automate the process and assist operators.

The automatic comprehension of the objects' behaviors within

a video sequence is a very challenging problem, as it involves the extraction of the visual information of interest, the choice of a suitable and convenient representation of these data and, finally, their comprehension. Being a difficult task by itself, the understanding problem is also complicated by the wide variability and unconstrained environments.

Most existing systems are designed only for given application contexts; this means they can work sufficiently well at the conditions under which they have been modeled, but their application in a different context is very unlikely.

When accounting for the high generality of the genre of objects of interest in a video scene, an example can contribute to clear the idea. Think, for instance, to homeland security or crime prevention, which are two of the hot topics of monitoring systems: in this contexts it is needed to trace the movements of any kind of object, such as individuals, children, pets, vehicles, moving in a wide variety of different environments. Such a rich activity requires techniques which are able to cope with a wide range of scenarios.

The analysis of trajectory dynamics permits to identify scene changes starting from low-level cues, so obtaining a significant independence from the application context. This said, for a full understanding of behaviors we still need some kind of knowledge about the context in which our objects of interest move. One of the techniques for modeling a complex scene is the POI/AP Framework.

When defining a scene of interest, at least two entities must be defined:

- The points of interest (*POIs*), that denote the image regions in which most of the actions are more likely to happen;
- The ACtivity Paths (*APs*), which characterize the motion of the objects from one POI to another.

Once defined POIs and APs, it is also needed a vocabulary for the unsupervised construction of the scene, in order to obtain the following achievements:

- Classification of activities, both regarding past and current events;
- Prediction of future activities;
- Detection of anomalies;
- Recognition of objects' interactions.

In addition, POI/AP models must be updated for the on-line introduction of new typologies of interactions as they occur.

Due to its capacity of being applied to different contexts, the POI/AP framework has been profitably implied in many articles in literature, among which:

- Monitoring of Parking Lots - aimed at ensuring the correct use of parking lots, events of interest in this category are the detection of abnormal driving behaviors [JWW+04], [RSJ04], the recognition of loitering of people [OH00], [MH00];
- Indoor Environments - analysis of the events occurring within indoor environments, like offices, laboratories, banks. In these cases the trajectories are constrained by the scene topology (doors, stationary objects). In [MT08a], once constructed the POI/AP framework, HMMs are used to represent APs, which are temporally adapted by means of online maximum likelihood regression. In [BK00] HMMs are adopted for modeling office activity and outdoor traffic. Naftel et al. [NK06], propose the use of Self Organizing Maps for the unsupervised learning of trajectory similarities. The authors in [XG05] propose the training of a behavior model using an unlabeled dataset. Finally, in [BBS06] is presented a system for the analysis of long-term pedestrian track data within a large hall of an Austrian railway station: the focus of the track analysis is the detection of places where people stop frequently or walk slowly, respectively.
- Outdoor Environments - outdoor environments are usually less constrained than indoor ones, as there is minor presence

of scene obstacles. In [RR05] human behavior is modeled as a stochastic sequence of actions; actions are described by a feature vector comprising both trajectory information (position and velocity), and a set of local motion descriptors. Some other works address the automatic extraction of frequently used pedestrian pathways from video sequences of natural outdoor scenes [ME02],[ME05]. As for indoor contexts, loitering has also been studied for outdoor scenes [BMPI05]. As spatial information are not always sufficient to detect potential anomalies, in [JJS04] the use of velocity and curvature information of a trajectory along with the spatial information is proposed. Some proposals are based on the use of tracking information for learning activity patterns [SG00b], [RRB06], [BAT05], [JH96], [YF05]. Other methods propose the use of Neural Networks for the construction of action models [SB00], [HXTM04].

- Objects' Interactions - the case of a scene with numerous objects interacting between each other is complex and particularly interesting; as in the case of the recognition of person interactions for video-surveillance, both for two-person [PA04a] and multiple persons interactions [PT07].

- Traffic Scenes - intelligent transportation systems is probably the most active research field within computer vision applied to trajectory analysis. Common desired task of these systems are the ability to detect and/or prevent accidents and collisions [AAM+05], [KMIS00], [SSL07], [HXX+04], traffic analysis [AMJP05], [KMIS00], [HXX+04], or to assist the drivers and increase the safety[Cha06], the estimation of motion parameters [SD03], [JH99], the use of hierarchical approaches for learning activities patterns [HXT04], [KKL+05], [WMG07], the detection and classification of traffic events [MNBSV06], [MMZ05], [HYCH06], the clustering of trajectories [LHH06], the real-time interpretations of interactions [KRWS05] and the monitoring of intersections [VMP03].

The POI/AP Framework, however, proves to be very useful

for any context related to behavior analysis; this is motivated by the need for more contextualized data, which means information about scene topology and its influence on moving objects' motion dynamics. For this reason, in the next chapter I will present a human behavior simulation model strongly based on topology information.

Chapter 3

A Social Force Model-based Stochastic Framework for Pedestrian Behavior Simulation

The simulation of pedestrian flows in indoor and outdoor environments has gained growing importance in the last years. Even if human behavior in complex real- life situations is far from being predictable, in certain situations it can be described using proper stochastic models obtained from social studies and analysis; in particular contexts, in fact, it has been demonstrated how the behavior of humans of the same ethnic group tend to follow common rules.

One of the researchers in the field of *pedestrian simulation*, Dirk Helbing, author of the widely used and known *Social Force Model*, through a qualitative and quantitative analysis of human behaviors in different contexts has synthesized the main human behavior features in the following issues:

- The main target of each person is the maximization of its comfort. This is achieved by a twofold behavior: from one side, it will search for the shortest path involving the smallest number of deviations, from the other side it will try to keep

a certain distance between itself and the other objects in the scene, whether they be individuals or obstacles. This distance is related to the previously mentioned personal area, which denotes the area within which each person feels itself comfortable and safe; its value is variable according to the environmental conditions, in fact it decreases in case of crowded scenes, meeting of known people and group formation.

- Each person possesses a preferential individual velocity, which average value has been measured to be $1.34ms^{-1}$ with standard deviation $0.26ms^{-1}$ [Wei92].

- When a group of people overtakes a stationary crowd, this group tends to form a row which will act like a river bed.

- When two pedestrian flows with opposite directions meet and cross all the involved individual tend to arrange themselves in lanes with uniform motion directions.

The above observations are also denoted as *self-organizing* phenomena and many attempts have been made to validate them into mathematical models.

A model able to realistically emulate human behaviors according to the context in which it is considered can be used for different purposes. Pedestrian simulation is widely adopted for urban planning and infrastructure design: simulated flows are used to detect remedial interventions aimed to maximize the usability of the scene elements or to predict the impact of planned structural modifications.

Another important application context of behavior simulation is evacuation and panic simulation: in this contexts it is needed to simulate growingly crowded evacuation events in order to assess the security infrastructure and policies. In addition, simulation is needed because we do not have enough real data and the number of events involving the participation of a high number of people is significantly increasing.

As an example of human behavior simulation for evacuation contexts, figure 3.1 depicts the bottleneck of a corridor: from simulation it is possible to observe that the configuration in the left image can lead to dangerous intersections of pedestrians' trajectories, so producing slow-downs or collisions of individuals, with possible falls and injuries. In the right image we can observe the corrective intervention needed to avoid the previous dangerous situations: here pedestrians adapt themselves to the minor space at their disposal. Such optimal configuration could also be obtained via a evolutionary algorithm with gradual improvements.

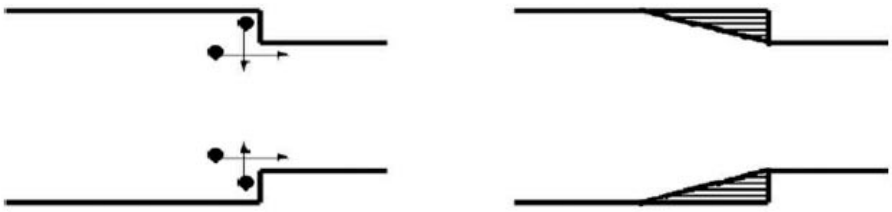


Figure 3.1 A bottleneck corridor

Behavior simulation is also important in at least another context: by using simulation in conjunction with video-surveillance systems, we can find heuristics for improving tracking capabilities and generate the reference trajectories used for the training of a *behavior recognition module*.

3.1 State of the Art

The various simulation strategies proposed in the years have adapted themselves to the progress and the computational devices available. Initial approaches were focused on the simulation of the evolution of a crowd according to the dynamics of a gas [MK07], using fluid dynamics equations to emulate the motion of a crowd inside a given environment. Such approaches are called *macroscopic* as they consider global parameters like flows, velocity and average density values. Their main disadvantages are the impossibility to model the behavior of single agents and the computational complexity of the equations.

Thanks to the availability of a greater computational power, recent approaches tend to be *microscopic* aiming at simulating the behavior of single pedestrians. Besides allowing to model individual pedestrians, they also permit to consider high-level features like sex, age, ethnic groups, and so on.

Among the microscopic simulation models, the most widely used ones are based on cellular automata and social forces.

3.1.1 Cellular Automata

Based on the observations of John Von Neumann and Stanislaw Ulam, cellular automata were initially assumed to be used in self-replicating systems. They can be considered as physical systems with discrete space and time [TPE07]; the space dimension is divided into cells with given size and shape: each cell can evolve in time according to fixed transition rules, which vary with respect to the type of automata. These simple rules, which control the transitions between the states of adjacent cells, can model complex systems.

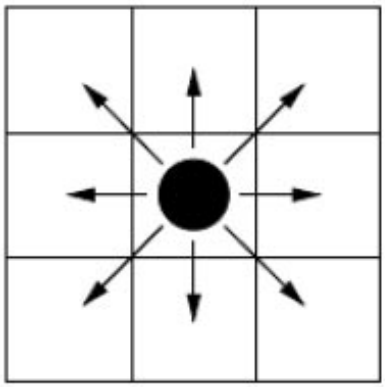
Cellular automata have been widely used for pedestrian simulation; for its simplicity, we will consider a very simple cellular automaton (CA) proposed in [Sch01], which is constituted by square cells (40 centimeters per side, this size being considered the approximate space occupied by a pedestrian in a crowd). Each cell possesses two different parameters used for the interactions of pedestrians and the scene topology:

- Static door field, which accounts for the scene topology and whose values are constant.

- Dynamic door field, which vary according to the interactions with pedestrians. For instance, if a pedestrian leaves a cell, this cell will increase its dynamic floor field value according to a diffusion model which will affect future interactions within this cell. This characteristic allows modeling the so-called self-organizing behaviors: pedestrians moving along

similar paths; a decay model will decrease this influence as the time passes.

In a CA motion also needs to be discrete, therefore it is needed to establish a simulation step, which will reproduce the time needed by a human to travel the side of a cell. More complex model allow the pedestrian move of more than one cell per iteration. Each pedestrian moves according to a probabilistic fashion: first a square 3x3 preference matrix is constructed, centered in the position of the pedestrian; each matrix element will contain the likelihood of the pedestrian to move towards that cell. The likelihood values can be obtained both through route selection algorithms or according to the pedestrians' purposes. A sample preference matrix is depicted in figure 3.2.



$M_{-1,-1}$	$M_{-1,0}$	$M_{-1,1}$
$M_{0,-1}$	$M_{0,0}$	$M_{0,1}$
$M_{1,-1}$	$M_{1,0}$	$M_{1,1}$

Figure 3.2 A sample preference matrix.

In each time instant, for every cell (i, j) belonging to the eight-neighborhood of the pedestrian we calculate the motion likelihood $p_{i,j}$ as follows:

$$p_{i,j} = NM_{i,j}D_{i,j}S_{i,j}(1 - n_{i,j}).$$

where $M_{i,j}$ is the matrix element of position (i, j), $D_{i,j}$ is the dynamic floor field value, $S_{i,j}$ is the static floor field value, $n_{i,j}$ is the occupation value of the cell (i, j) (1 if occupied, 0 if not) and N is a normalization factor used to ensure that the sum of the

single likelihoods is equal to 1. According to the cellular automata formulation, each cell can be occupied by only one pedestrian: when two ore more pedestrian want to move to the same cell, the one with greater relative likelihood will move.

3.2 The Social Force Model

Social Force Models (SFMs) are based from the following consideration: each person moving in a given scene undergoes various external impulse which affect its motion behavior; its main goal remains to reach a given destination, but during its path it will avoid collisions with other agents or obstacles, it will slow down in presence of elements of interest, and so on.

These impulses, theorized the German psychologist Kurt Lewin [LC51], can be seen as acting similarly to the concept of (social) forces. In their first formulation, the social forces were only related to the intrinsic properties of the target, but they could still be interpreted as its reaction to an external stimulus.

It was Dirk Helbing and PÃ¨ter Molnar who first applied this concept to model the self-organization phenomena of pedestrians [HM95]. Their mathematical formalization, in fact, based on concepts of Newtonian Physics, provided a mechanical interpretation of the external impulses soliciting pedestrian: these forces revealed themselves as accelerations modifying pedestrians' motion.

In each time instant every target undergoes an acceleration equal to:

$$\frac{d\vec{v}}{dt} = \vec{f}(t) + \xi(t).$$

where $\vec{f}(t)$ accounts for the overall effect of the forces acting on the pedestrian: the intentional force, aimed at maintaining the velocity and the direction needed tor each the destination, the repulsion force towards obstacles and the other pedestrians and the attraction force of the elements of interest. In addition to these forces, fluctuations are introduced by the term $\xi(t)$ to guarantee the diversity of the behaviors.

Each pedestrian tries to keep its preferential velocity and direction through an intentional force which applied a corrective acceleration; this force is not constant, as it can increase in case of hurry or decrease in case of panic situations. The defense of the personal area (or comfort area) is associated to a repulsive force which is inversely proportional to the distance between the pedestrian and the obstacle/other pedestrian [HMFB01].

Another interesting property is the asymmetry of the forces: an anisotropic coefficient, assuming values between 0 and 1, influences the intensity of the forces according to the pedestrian's field of view. This is motivated by social studies; humans, in fact, tend to perceive and react more intensely towards what happens in their field of view and, through the use of other senses (hearing) less reactively to what happens at their back.

The validity of this model has been shown in different context; in particular, in [HM95] is taken into consideration a corridor scene sized 10x50 meters with high density of pedestrians. Figure 3.3 shows a self-organizing phenomena: the formation of lanes when two opposite flows meet and cross.

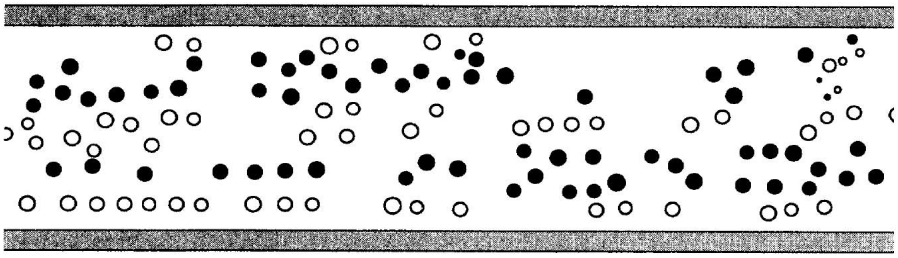


Figure 3.3 Self-organizing phenomena: lane formation [HM95].

3.2.1 Modifications of the HMFV social force model for pedestrian evolution

Social Force Model have been successfully used for evacuation scenarios [HFMV02]; the proposed HMFV model (from the authors' names, Helbing, Molnar, Farkas and Vicsek), however, still presented some limitations:

- Pedestrians overlapping is prevented through the introduction of an elastic constant; this constant, however, applies forces to the pedestrian which are even seven times superior to their weight, which is unrealistic.

- The minimum distance between two objects, beyond which the attraction/repulsion force tends to zero, makes the pedestrians exhibit acceleration values greater than the gravity acceleration.

These issues have been partially solved in [LKF05]; the authors, in fact, have introduced an algorithm for the removal of pedestrians' overlapping and an adaptive simulation time step; in addition, repulsion forces functions have been redefined in order to avoid unreal acceleration values.

The overall force acting on each pedestrian is defined as:

$$\vec{f}_{i,j} = \vec{f}_{socialrepulsion} + \vec{f}_{pushing} + \vec{f}_{friction}.$$

Each of the above forces contributes to the pedestrians' motion:

$$\vec{f}_{socialrepulsion} = Ae^{\frac{R_{ij}-d_{ij}}{B}}\vec{n}_{ij}.$$

$$\vec{f}_{pushing} = k\eta(R_{ij} - d_{ij})\vec{n}_{ij}.$$

$$\vec{f}_{friction} = k(R_{ij} - d_{ij})\vec{t}_{ij}.$$

where A is the module of the attraction/repulsion force, B is the fall-off value, k is a constant of the model, R_{ij} is the sum of the radii of the two pedestrians, d_{ij} is the distance of the pedestrians' centers, n_{ij} is the unitary vector from one pedestrian's center to the other's and t_{ij} is the unitary vector tangent to n_{ij} and directed opposite to the velocity of pedestrian i. The function η is introduced to nullify the pushing and friction effects when two pedestrian are not in contact and is defined as:

$$\eta(x) := \begin{cases} x & \text{if } x \geq 0, \\ 0 & \text{if } x < 0. \end{cases}$$

If the interaction occurs with an obstacle, the vectors $\vec{n}$ and $\vec{t}$ will be respectively perpendicular and tangent to the area of the obstacle.

The innovation of this model with respect to the HMFV model lies in the reformulation of the intentional acceleration:

$$\frac{dv}{dt} = -\frac{v - v_0(1+E)}{\tau} - \frac{f_{social}(r_{||})}{m} + b\frac{f_{social}(r_{||})}{m}. \tag{3.1}$$

where E is a new parameter, the excitement factor, whose value is calculated as:

$$\frac{dE}{dt} = -\frac{E}{T} + \frac{E_m}{T}\left(1 - \frac{v}{v_0}\right).$$

As it can be noticed from the above formulation, the temporal variation of E allows the pedestrian to accelerate when its velocity is lower than the preferential velocity. The first term of the (3.1) makes the pedestrian maintain its preferential velocity, the second term expresses the social force of repulsion towards the other pedestrians standing in front of us while the third term refers to the pedestrian at our back.

3.3 Stochastic Topological Social Force Model

As documented in the previous paragraphs, many models have been proposed and applied to different contexts; these models, however, even being optimal instruments for the modeling of given self-organizing phenomena in particular environments, are unable to reproduce more generic behaviors in generic contexts. In addition, as the goal of this PhD dissertation is the study of a system

for efficient indexing and retrieval of trajectories, the implementation of a human behavior simulation system has also fulfilled another requirement: the need for significant data for the experimental evaluation.

In fact, as it will be clearer in the next chapters, in order to perform a good experimental evaluation of the query processing strategies, it is needed a significant amount of data. As the available real-world trajectory datasets usually lack in both the number of trajectories and the average trajectory length, some synthetic generator have been proposed. These generators, however, all suffer from the same limitation: the generated data are not related to any context or topology and, for this reason, are not meaningful from a behavioral point of view, which is crucial for this study. This is the reason why we studied a social force model and a set of heuristics for the simulation of more complex human behaviors related to the contextual information (scene topology). In the following paragraphs I will discuss about its application to pedestrian behavior simulation, also describing some of the heuristics regulating the simulation model.

3.4 Implementation

The Human Behavior Simulation System (HBSS) has been implemented in C# with .NET Microsoft Visual Studio 2010 Professional; the main goal has been to obtain a suite for the simulation and the analysis of the generated data. In this chapter we will not discuss any detail, rather we will focus on the most significant aspects.

The HBSS is constituted by three modules: the editing module, the simulation module and the data display and analysis module.

3.4.1 Editing Module

A great attention has been devoted to the editing module as it allows the definition of the topology, which will strongly influence

pedestrians' behaviors. For this reason, the main aim of this module has been to give the user the possibility of a full customization of the scenario, with a library of default elements and the chance to create new elements and expand the default library.

The main entities in the definition of a scenario are the *Entry/Exit Point (EEP)*, the *Obstacle (Ob)* and the *Area of Interest (AoI)*. EEPs and Obs are described by their unique ID, position and size; AoIs, on the contrary, are much more complex. The editing module allows to define two AoI's typologies: plane AoIs represent outdoor freely accessible places like squares or parking areas, while solid AoIs can be used to represent structures having insurmountable limits like shops, banks, etc. The user can give thus define AoI's entry points for the target's access and interaction points, which are the points in which the interactions occur (for example the console for an ATM).

The interactions with the AoIs last in relation to a probabilistic framework and according to an average value and a standard deviation value defined in the system; of course, the user can change these values and define different values for each interaction point. In addition, it is possible to set a maximum AoI occupation value and a maximum AoI interactions value, which denote the spatial capacity and the maximum number of contemporary interactions allowed respectively. These values can be used for various aims: for disaster recovery, for instance, we define an AoI with null maximum AoI interaction value and we will test how the system would act in case of impossibility to use that resource; on the contrary, by defining a quite high value we will test the formation of queues and the management of the wait list for interactions.

Finally it is possible to discriminate the importance of each AoI by defining three different indexes (valued from 0 to 1):

- the *density index*, which is defined by the user and accounts for the importance of every AoI on the basis of its density.
- the *functional interest index*, which is defined by the user and represents an objective evaluation of the interest of use of an area.

- the *superficial index*, which is strictly related to the structural characteristics and is defined as:

$$I_{sup}(AoI_i) = \frac{Area(AoI_i)}{MaxArea}.$$

These three indexes are used to define the *attraction radius* of an AoI, which denotes the influence area within which pedestrians can undergo the attraction force of an AoI. The attraction radius is calculated in relations to a weighted sum of the indexes:

$$I_{Sum} = I_{density}(AoI_i) * 0.2 + I_{sup}(AoI_i) * 0.3 + I_{funct}(AoI_i) * 0.5.$$

therefore the radius is defined as:

$$R_{attr} = Max[Size(AoI_i)] + \{I_{Sum} * [Width(AoI_i) + Height(AoI)]\}.$$

3.4.2 Simulation Heuristics

The main class is the *Simulation* class, which coordinates the generation of pseudo-random targets' behaviors according to the configuration parameters and constraints. In order to guarantee good performance and data consistency, the best trade-off must be found in the choice of the timestep: a shorter timestep, in fact, would result in very precise data but very low performance due to the computation load. In this application we choose to define an adaptive timestep: in each time instant the timestep is chose within a given interval according to various information, like the state of the entire scenario, the positions of the target with respect to AoIs and Obs, all the external forces acting on the target, the acceleration to which each target is subject and the estimated time needed by the target to meet a topology element. This choice allows good simulation and visualization performance, while ensuring data integrity and consistency.

The *Social Force Manager* class is in charge of calculating, for each time instant, the acceleration to which the target is subject,

in relation to the forces acting on it and the target's intentions. The modulus and the direction of these forces is defined as:

$$\vec{f} = \vec{f}_{socialrepulsion} + \vec{f}_{pushing} + \vec{f}_{friction}$$
$$\vec{f} = Ae^{\frac{R_{ij}-d_{ij}}{B}}\vec{n}_{ij} + k\eta(R_{ij} - d_{ij})\vec{n}_{ij} + k(R_{ij} - d_{ij})\vec{t}_{ij}.$$

In addition, the overall acceleration of each target is also obtained through two other components which have been defined previously: the target's intention to maintain its preferential velocity and the target's intention to reach its destination.

Each object moving in the scene can be represented through the *Target* class, which encodes the information useful to describe the target and the parameters and methods for its motor control and its interactions with the scene.

Many other heuristics have been introduced to correctly and completely describe targets' motion in the scene, but an exhaustive discussion is beyond the scope of this dissertation. A better comprehension of target's motion characteristics, however, can be deduced by the possible states that each target can assume in the scene, which are describe through the Non-Deterministic Finite State Automaton (NDFSA) in figure 3.4.

Without going into detail of all the transition functions regulating this NDFSA, it can be said that a main stochastic framework supervise the generation of targets' motion behaviors giving them the maximum number of degrees of freedom, both for the motion characteristics and for their interactions with the scene. For example, a Dijkstra-based shortest path finding heuristics is used to allow each target to move around obstacles; targets' behavior are obtained through attraction and repulsion forces, as theorized in [LKF05], but with a great difference: the variables playing in these functions are also strictly related to the AoIs' characteristics and to the targets' intentions, so giving rise to what we have called *Stochastic Topological Social Force Model*. A further but not comprehensive example of the heuristics introduced to simulate at best human behavior concerns the target's choice to exit

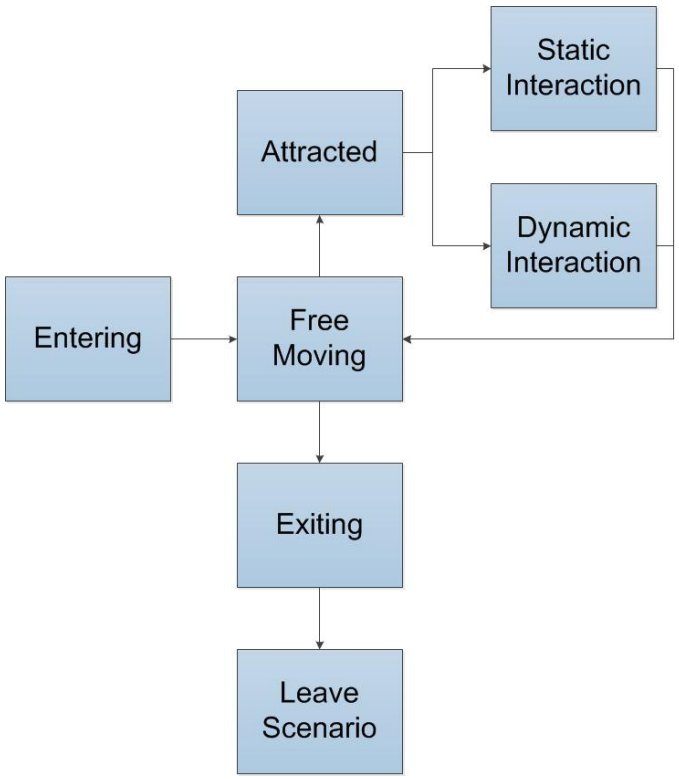


Figure 3.4 The NDFSA describing target's states transitions.

the scene: each target, in fact, is assigned a potential energy at its creation time, which it can use to perform tis actions in the scene; this value is chosen according to target's mass, the modulus of its preferential velocity and the average trajectory length defined in the system. Once this energy has been consumed, it will more likely decide to exit the scene.

Finally, in order to keep the simulation realistic and feasible, some constraints are adopted, both pre-defined in the system and introduced by the user in a preprocessing phase.

3.4.3 Data Display and Analysis

Once the simulation is completed, a Graphical User Interface has been designed for the display and the analysis of these data. The

storage and indexing of the data has been devoted to the Spatial Database Management System (SDBMS) PostGIS; for the complete analysis of the data, have also been made available a segmentation algorithm and query processing strategies. a detailed discussion about storage, indexing and query processing techniques will be the focus of next chapters.

The basic function of the data display and analysis module is the visualization of the simulation data; moreover, through a very easy to use query interface, the user can also submit queries to the system in order to extract information of interest about targets' behavior.

3.5 Conclusions

Pedestrian simulation has gained strong interest recently, in particular for the implementation of security measures for critical infrastructures. As the main goal of this dissertation is the presentation of an innovative approach for the efficient indexing and retrieval of spatio-temporal information and according to the limitation of the available real-world datasets in the number and in the average length of trajectories, we have proposed a human behavior simulation model whose synthetic data provide information that are not only sufficient to test the efficiency in terms of trajectory size and number of trajectory that can be handled, but also significant from a behavioral perspective, which is a crucial point when aiming at performing complex queries over moving objects.

In spite of their simplicity, Social Force Model proves to be effective in the simulation of human behaviors in many application contexts; in addition, we have presented a model which is also strongly influenced by the intentions of the targets acting in the scene; therefore, the obtained data is significantly correlated to the topology characteristics.

Although various heuristics have been introduced for a complete management of human behavior, the model can be further improved by adding additional features. Just to give an example,

the adding of information like targets' age, sex, ethnic group, will influence not only their motion but also their interactions with the topology's entities.

Chapter 4

An Efficient Bi-Dimensional Indexing Scheme for Three-Dimensional Trajectories

As discussed in the introduction of this dissertation, many application contexts require the efficient storage of the information collected by different sensors and related to moving objects. For an optimal management of this information, among the various data collection systems that have been proposed, it is worth mentioning Moving Object Databases (MODs), which aim at the efficient organization of data pertaining to the objects moving in a scene.

When referring to moving objects, we can consider many different information, but what we cannot do without is the information about moving objects' position over time, id est the spatio-temporal information.

Spatial Database Management Systems (SDBMS) are database systems designed and optimized for storing, indexing and operating on data related to objects in space, including points, lines and polygons. Through the concept of geometry, SDBMS allow the

management of data which can be not only numbers or characters, as for any other database system, but also spatial data types.

Classic DBMS use indexes for efficiently search data; these indexes, however, are not optimal for spatial data. SDBMS are designed to work better on spatial information thanks to the support for some additional functionalities like classic intersection, intersection of geometries, spatial measurements, and so on.

We choose to store spatio-temporal information in PostGIS, a well-known extension of PostgreSQL which adds support for geographic objects. PostGIS enhances PostGreSQL with R-Tree over GiST, the basic structure for the storage and indexing of spatio-temporal information, in compliance with the standards of Open Geospatial Consortium (OGC). The indexes available in PostGIS are, therefore, based on the well-known R-Trees, which constitute probably the most influential accessing methods in the area of spatial management and which will be discussed later on in this chapter.

4.1 Trajectory Indexing

In a pioneering paper [Gut84] Guttman proposed a structure, named R-Tree, able to achieve the efficient indexing of bi-dimensional rectangular objects in Very Large Scale Integration (VLSI) design applications. R-Trees hierarchically organize the geometric objects by means of Minimum Bounding Rectangles (MBR) [PT97], which are the smallest rectangles enclosing the related object. Each R-Tree's internal node corresponds to the MBR bounding its children and every R-Tree's leaf is a pointer to the objects (see figure 4.1). When it is required to insert a new object, first it must be found the node that, for each level, will involve the smallest expansion; for node splits, on the contrary, three possibilities held which differ in complexities but all of which are based on the minimization of the sum of the areas of resulting nodes.

Due to their simplicity and the ease to use them in SDBMS, R-Trees have been widely adopted and studied, and lots of mod-

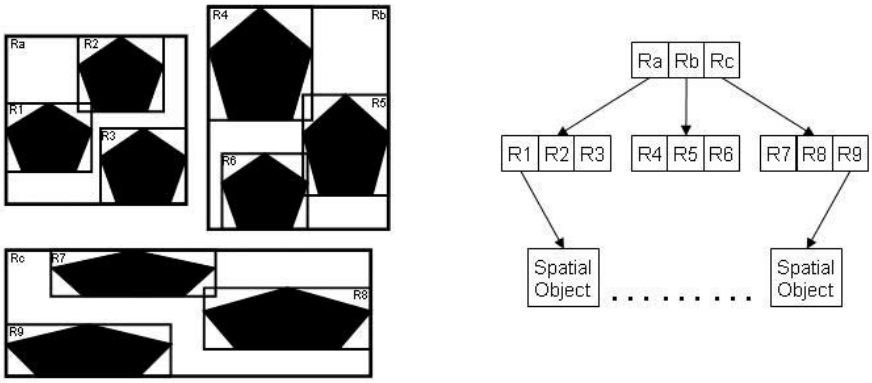


Figure 4.1 The structure of an R-Tree [MNPT05].

ifications of them have been proposed. For instance, using three-dimensional R-Trees [TVS96] the temporal coordinate is considered as an extra dimension; this approach results effective only when complete trajectories are available. Another extension of R-Trees are STR-Trees [PJT+00], which use a different insert/split algorithm and two different access methods: STR-Tree and TB-Tree.

In some cases object have limited motion possibilities: this is the case of constrained environments, like for example on a network of connected segments, a train moving on a rail circuit or vehicles in a urban context. In these cases a basic idea is to use bi-dimensional R-Trees to index the static segments of the network, like for the FNR-Trees [Fre03]: each leaf is associated to a segment and contains a pointer to a one-dimensional R-Tree indexing the time intervals of the objects' movements. An extension of these trees is MON-Tree, an indexing structure designed for constrained networks which uses a 2D R-Tree for indexing the polylines' bounding boxes and another 2D R-Tree for the temporal dimension; similarly to FNR-Tree, PARINET [SPZO+10] aims at modeling constrained networks as a graph by partitioning the trajectories; this indexing scheme has also been used for continuous indexing of moving objects [SPZO+11].

Other indexing schemes are aimed at solving the problem of

storing and indexing huge amount of trajectory data by reducing the redundancy in the representation of trajectories. In [RSEN05], for instance, a dynamic programming algorithm is used to minimize the number of I/O operations by first segmenting each trajectory and, subsequently, using an R-Tree for each of the trajectory segments. SETI [CEP03] consists in the segmentation of trajectories and the partitioning of their sub-trajectories into a collection of *spatial partitions*: according to the query typology, only the relevant sub-trajectories' partitions will be taken into account. Other approaches operate at a lower level, like TrajStore [CMWM10], which maintains an optimal index on the stored data and co-locates and compresses on a disk block the segments of a trajectory. The main idea of this system is to evolve as long as the user submits queries, so that the answer to most of the queries will only involve the reading of a small number of storage cells.

4.1.1 Common Limitations

All the above approaches have both advantages and disadvantages; some of them are particularly efficient for given domains, while the last ones aim to operate at a lower level (physical storage). All the above discussed three-dimensional indexing schemes, however, are commonly not available in both commercial and open source products. In fact these products, even making available very efficient indexes, are limited by the fact that the most important functionalities are typically restricted to work with bi-dimensional data. As in the case of the database system previously mentioned, PostGIS, there is support for three and even four-dimensional data but still lacks the support for the indexing and intersection operation in the three-dimensional case, as table in figure 4.2 clearly expresses.

The main difference in the supported and unsupported operations can be derived from an example related to the intersection operation. As a matter of fact, if we are only interested in knowing if the MBRs of two trajectories intersect, then a very efficient implementation of the simple intersection operation is available (de-

Function	geometry	geography	3D (2.5D)	Curves	SQL MM
=	✓	✓		✓	
«	✓				
|&>	✓				
&<|	✓			✓	
&&	✓	✓		✓	
&<	✓				
&>	✓				

Figure 4.2 PostGIS functions support table.

noted in PostGIS with '&&'); on the opposite, as usually happens in real applications, if we want to know if two trajectories intersect, meaning that there must be at least one intersection point which belongs to both trajectories, then we must find the intersection between the geometries, which is not efficiently supported for the three-dimensional case. In addition, the simple intersection is a necessary but not sufficient condition for the geometric intersection, as in the example in figure 4.3.

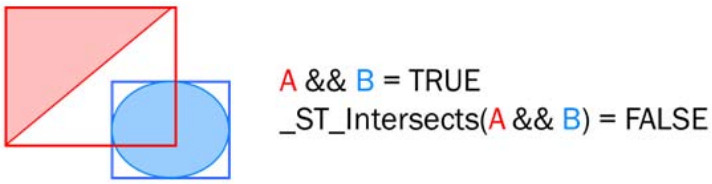


Figure 4.3 The two intersection operations in PostGIS.

The choice of the operations taken in exam is not casual: indexing and intersection operations, in fact, are the basis for most of the queries over spatio-temporal data. It should now be clear the reason why there is strong need for methods capable of supporting the three-dimensional operations of indexing and geometry intersection with efficient implementations.

4.2 The Proposed Approach

In order to overcome the limitation analyzed in the previous paragraphs, we propose a novel indexing scheme that, together with

a segmentation algorithm, will allow the use of very efficient bi-dimensional indexes for the three (and even N)-dimensional case. In the following,the various aspects of the proposal will be addressed: the representation of the trajectories, the indexing scheme compared to a another solution and the segmentations stage.

4.2.1 Trajectory Representation

According to the line segment model, a trajectory T^k can be represented as a sequence of spatio-temporal points:

$$T^k =< P_1^k, P_2^k, ..., P_n^k >$$

with:

$$P_i^k = (x_i^k, y_i^k, t_i^k) \quad \forall i \in [1, n].$$

Each pair (x_i^k, y_i^k) is referred to the spatial location of an object at the time instant t_i^k. The entire trajectory is approximated by a polyline, each segment being obtained by linear interpolation between two consecutive points (see figure 4.4).

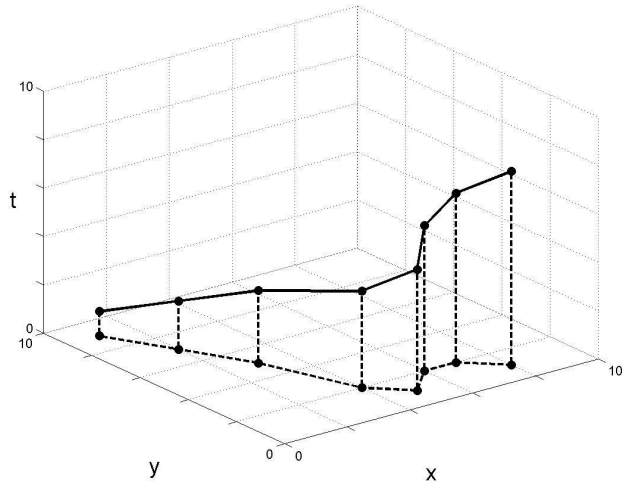


Figure 4.4 A three-dimensional trajectory, with x and y referring to the spatial dimension and y referring to time.

4.2.2 Indexing Scheme

A simple algorithm for retrieving the trajectories satisfying such a query is based on processing, for each trajectory, all its segments, starting from the first one: as soon as the intersection occurs, it can be concluded that the trajectory actually intersects the query box. One of the algorithms for determining whether a trajectory segment lies inside or outside a query box is the clipping algorithm.

We propose to use the 2D Cohen-Sutherland Line Clipping Algorithm [FVDFH12] (briefly summarized in figure 4.5). According to it, we subdivide the geometric plane into nine areas by extending the edges of the query rectangle and only the starting and ending point of each segment are considered (figure 4.5.a). If at least one of these endpoints lies inside the query box, the segment clearly intersects the query box, as happens for segment AB in figure 4.5.a.

When both the endpoints lie outside the query box, we can still use the endpoints position with respect to the query area to trivially verify whether the segment intersects the query box or not, as happens for segments CD and EF respectively in figure 4.5.b; otherwise, the segment needs to be split into two or more segments by considering the intersection points between the edges of the area to be clipped and their extensions. Two cases are possible: if at least one of the intersection points lies on the edge of the query box (as in the case of the segment GH in figure 4.5.c), then the considered segment intersects the rectangle, otherwise the intersection condition is not met (see segment IL in figure 4.5.c).

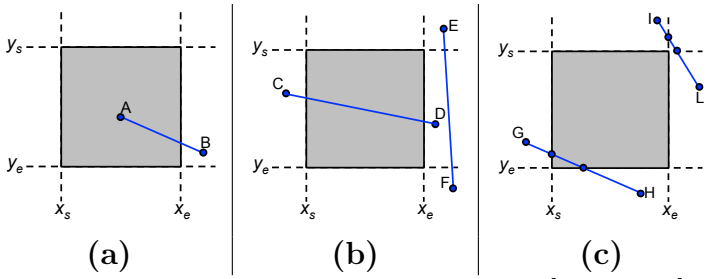


Figure 4.5 The Cohen-Sutherland Algorithm [FVDFH12].

This clipping algorithm can be easily extended for dealing with 3D trajectories by considering the three-dimensional plane (instead of the bi-dimensional one) and by subdividing the geometric plane into 27 spatial regions (rather than 9).

Unfortunately, despite its simplicity, the use of a clipping algorithm is not suited for handling with large datasets, so demanding for more efficient approaches. The main problem lies in the necessity for the clipping algorithm to process, for any trajectory, all its segments, starting from the first one, until the intersection occurs. The worst case arises when a trajectory does not intersect at all the query box; in this case, sure enough, all the trajectory's segments must be processed, making this approach unfeasible for large amount of data.

At this point it is evident that, in presence of a significant number of trajectories, more efficient approaches are mandatory; one first possibility consists in the use of suitable indexing strategies aimed to reduce the number of trajectories to be clipped.

Although many spatial databases today available, both open source and commercial ones, provide very efficient spatial indexing techniques, these indexes schemes are unfortunately typically restricted to deal with bi-dimensional data; for this reason, it is worth trying to represent the actual 3D problem in terms of (one or more) 2D sub-problems, for a complete fruition of the efficient available bi-dimensional indexes.

4.2.3 Comparison with previous method

In [dSV11] was proposed a method that, given a trajectory T^k and a query box B, both T^k and B were first projected on the three coordinate planes. Let T^k_{kz} and B_{kz} be the projections of T^k and of B on the kz plane: if the trajectory intersects the 3D query box, then each trajectory projection must also intersect the correspondent query box projection:

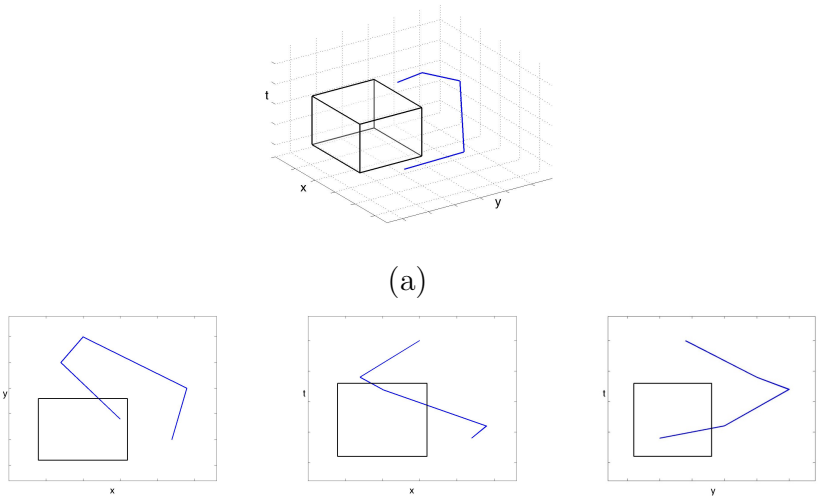


Figure 4.6 An example of 3D trajectory (a) and its projections on the different coordinate planes xy (b), xt (c) and yt (d). Although the trajectory does not intersect the query box, its projections do it.

$$T^k \cap B \neq \emptyset \Rightarrow \left\{ \begin{array}{c} T^k_{xy} \cap B_{xy} \neq \emptyset \\ T^k_{xt} \cap B_{xt} \neq \emptyset \\ T^k_{yt} \cap B_{yt} \neq \emptyset \end{array} \right\} \tag{4.1}$$

Equation 4.1 represents a necessary but not sufficient condition, as the opposite is clearly not true. In fact, if all trajectory's projections intersect the correspondent box projection on the considered spaces, they do not necessarily intersect the 3D query box too. To better explain this concept, figure 4.6.a shows, in the 3D space, a trajectory that does not intersect a given query box: it can be noticed that all the trajectory projections intersect the correspondent query box projections (Figure 4.6.b-d).

Thus, as a matter of fact, if all projections of T^k intersect the correspondent box projections, we consider T^k as a candidate to be clipped in the three-dimensional space; the guess is that there will be not too many *false positives*.

According to the above considerations, in [dSV11], for each

three-dimensional trajectory T^k, it was needed to store the three bi-dimensional trajectories obtained by projecting T^k on the xy plane (T^k_{xy}), on the xt plane (T^k_{xt}) and on the yt plane (T^k_{yt}).

Given a box B representing the query to be solved, we similarly considered B_{xy}, B_{xt} and B_{yt}.

With this strategy, by using one of the available bi-dimensional indexes, it is possible to find on each plane the following three trajectory sets (Θ_1, Θ_2, Θ_3) in a very simple and efficient manner:

$$\Theta_{xy} = \{T_{xy} : MBR(T_{xy}) \cap B_{xy} \neq \emptyset\} \tag{4.2}$$

$$\Theta_{xt} = \{T_{xt} : MBR(T_{xt}) \cap B_{xt} \neq \emptyset\} \tag{4.3}$$

$$\Theta_{yt} = \{T_{yt} : MBR(T_{yt}) \cap B_{yt} \neq \emptyset\} \tag{4.4}$$

The set T of the trajectories candidate to be clipped in the 3D space is thus trivially defined as:

$$\Theta = \{T : T_{xy} \in \Theta_{xy} \wedge T_{xt} \in \Theta_{xt} \wedge \Theta_{yt} \in T_{yt}\} \tag{4.5}$$

The main advantage of this proposal was, of course, the possibility to use the widely available and efficient bi-dimensional indexes; despite this, this proposal still presented two weak points: first, for a n points trajectory, it was needed to redundantly store $6 \cdot n$ values ($2 \cdot n$ for each of the three coordinate planes); second, the use of the bi-dimensional indexes was not optimized: in fact, the MBR of each projected trajectory can easily span a great percentage of the whole area.

In the following two subsections, the above problems will be separately handled and solutions for them will be presented [dLSV12a].

4.2.4 Spatial Complexity Reduction

It is possible to observe that, for a given trajectory T^k, rather than storing the three different trajectory projections in each coordinate plane, we can store T^k as the original sequence of points in the 3D space, and separately maintain three different bidimensional MBRs: $MBR_{xy}(T^k)$, $MBR_{xt}(T^k)$ and $MBR_{yt}(T^k)$.

$MBR_{xy}(T^k)$ (respectively $MBR_{xt}(T^k)$ and $MBR_{yt}(T^k)$) is obtained by projecting on the xy (respectively xt and yt) plane the three-dimensional MBR of T^k.

It is worth noting that the redundancy introduced by the three MBR projections is not dependent on the number of points in the trajectory and, therefore, has only a marginal impact on the spatial complexity, since it only requires the storage of six pairs of points.

Assuming such a scheme, on each 2D plane we find the trajectories intersecting the corresponding 2D query box in a very efficient manner by using one of the available 2D indexes. Let Γ_{xy}, Γ_{xt} and Γ_{yt} be the resulting sets of trajectories defined as:

$$\Gamma_{xy} = \{T : MBR_{xy}(T) \cap B_{xy} \neq \emptyset\} \tag{4.6}$$

$$\Gamma_{xt} = \{T : MBR_{xt}(T) \cap B_{xt} \neq \emptyset\} \tag{4.7}$$

$$\Gamma_{yt} = \{T : MBR_{yt}(T) \cap B_{yt} \neq \emptyset\}, \tag{4.8}$$

where, as usual, B_{xy}, B_{xt} and B_{yt} are the projections of the 3D query box B. The set Θ of the trajectories candidate to be clipped in the 3D space is therefore now defined as:

$$\Theta = \Gamma_{xy} \cap \Gamma_{xt} \cap \Gamma_{yt} \tag{4.9}$$

Figure 4.7 resumes the method: given a set of trajectories and a query box (Figure 4.7.a), we discard the green trajectory since its $MBR_{xy}(T)$ and $MBR_{yt}(T)$ do not intersect the corresponding projection of the black query box (Figure 4.7.b and Figure 4.7.c). The candidate set Θ is thus composed by the other two trajectories, the red and the blue one, which are finally clipped, obtaining the desired output represented by the blue trajectory (Figure 4.7.e), which is the only trajectory satisfying the parameters of the query submitted by the user.

4.2.5 Segmentation Algorithm

It should be clear at this point that the entire system performance will strongly depend on the indexing phase and, as a consequence,

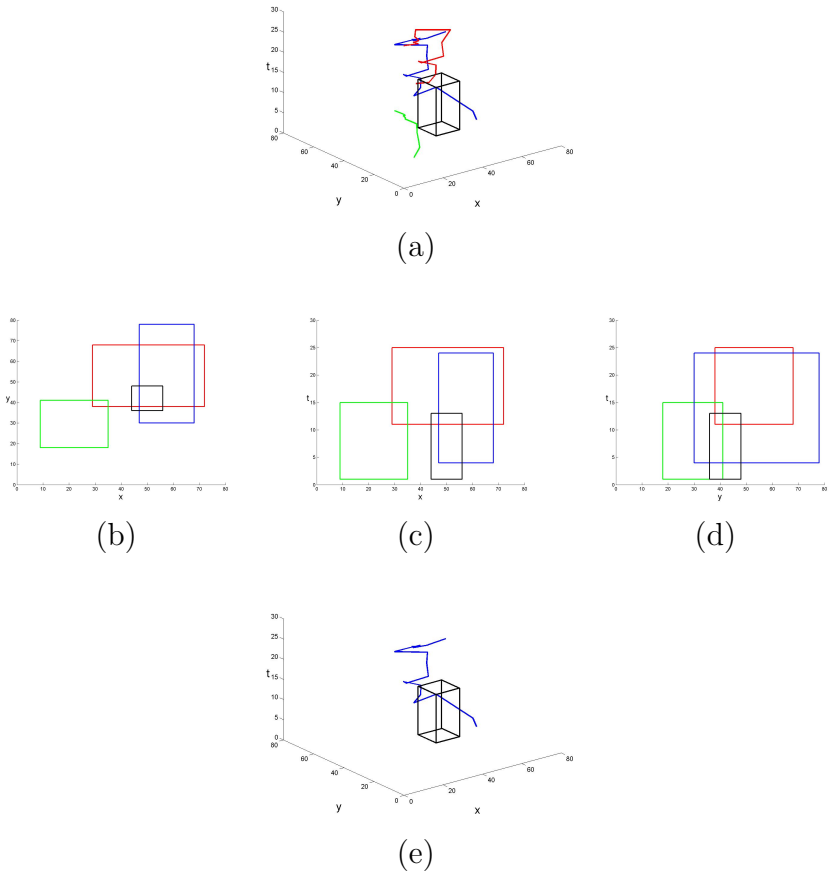


Figure 4.7 An overview of the method. (a) a query box and three trajectories; (b), (c), (d): the projections of trajectories' MBRs on the coordinate planes; (e) the final result of our method, after the application of the clipping algorithm on the blu and red trajectories.

on the capability to reduce the number of trajectories to be clipped in the three-dimensional space. At a more detailed analysis, the *selectivity* of the indexes in each plane is related to the area of the corresponding MBR which, in turn, only depends on the trajectory geometry, so being (apparently) fixed. For this reason we perform a segmentation stage in order to increase the selectivity of the indexes.

Segmentation algorithms aim at subdividing each trajectory into consecutive smaller units, which we will refer to as trajectory units. We are interested in a segmentation algorithm able to exploit the characteristics of the available bi-dimensional indexes; this can be accomplished by decreasing the area of the projected MBR of each trajectory unit, which is the basis of the representation of trajectories.

The proposed algorithm works recursively: initially (that is at iteration 0) it assumes that the trajectory T^k is composed by a single unit ${}^0U_1^k$, that is split into a set of m consecutive smaller units $\{{}^1U_1^k, \ldots, {}^1U_m^k\}$; each of the ${}^1U_i^k$ is in turn inspected and, if the stop criteria are not satisfied, it is further split.

Let us analyze how a generic unit ${}^{(i-1)}U = \{P_1, \ldots, P_m\}$ is split into $\{{}^iU_1, \ldots, {}^iU_n\}$; we first choose a *split-dimension* and a *split-value*. Assume, as an example and without loss of generality, that x has been chosen as the *split-dimension* and let x^* be the *split-value*. In addition, assume that $x_1 < x^*$. According to these hypotheses, ${}^iU_1$ is the set of the consecutive points lying on the left of the *split-value*:

$$ {}^iU_1 = \{P_1, \ldots, P_k\} \tag{4.10} $$

where P_k is the first point such that $x_k \geq x^*$. Then, the second unit will be formed by the sequence of consecutive points lying on the right of the *split-value*:

$$ {}^iU_2 = \{P_{k+1}, \ldots, P_l\} \tag{4.11} $$

where P_l is the first point such that $x_k \leq x^*$. The inspection of ${}^{(i-1)}U$ ends when the last point P_m is reached.

According to the above considerations, the criteria for the choice of the two parameters, *split-dimension* and *split-value*, play a crucial role. Since we aim at optimizing the indexing strategy, the proposed segmentation algorithm is based on the occupancy percentage on each 2D coordinate plane.

First we calculate the coordinate plane corresponding to the maximum among the three occupancy percentage values O_{xy}, O_{xt}

and O_{yt} of the trajectory unit MBRs, with respect to the correspondent global volume of interest V:

$$O_{xy} = \frac{MBR^{xy}(U)}{V^{xy}} \tag{4.12}$$

$$O_{xt} = \frac{MBR^{xt}(U)}{V^{xt}} \tag{4.13}$$

$$O_{yt} = \frac{MBR^{yt}(U)}{V^{yt}} \tag{4.14}$$

Without loss of generality, suppose that the maximum occupancy percentage value is O_{xy} and, consequently, the corresponding plane is xy; let *width* and *height* be the two dimensions of $MBR_{xy}(U)$, respectively along the coordinates x and y; the *split-dimension sd* is defined as:

$$sd = \begin{cases} x & \text{if width} > \text{height} \\ \\ y & \text{otherwise} \end{cases}$$

Given the *split-dimension sd* we choose, as the *split-value* sd^*, the MBR average point on the coordinate *sd*.

Figure 4.8 sketches the execution of the first iteration of our algorithm on the trajectory T (assumed to be composed at this iteration by a single unit U).

In Figure 4.8.a we are assuming that our volume of interest is:

$$0 \leq x \leq 150; 0 \leq y \leq 50; 0 \leq t \leq 30 \tag{4.15}$$

We first consider $MBR_{xy}(U)$, $MBR_{xt}(U)$ and $MBR_{yt}(U)$ (Figure 4.8.b) obtaining that:

$$O_{xy} > O_{xt} > O_{yt}. \tag{4.16}$$

According to the above inequalities, we choose to operate on the xy plane. As the values of the dimensions are:

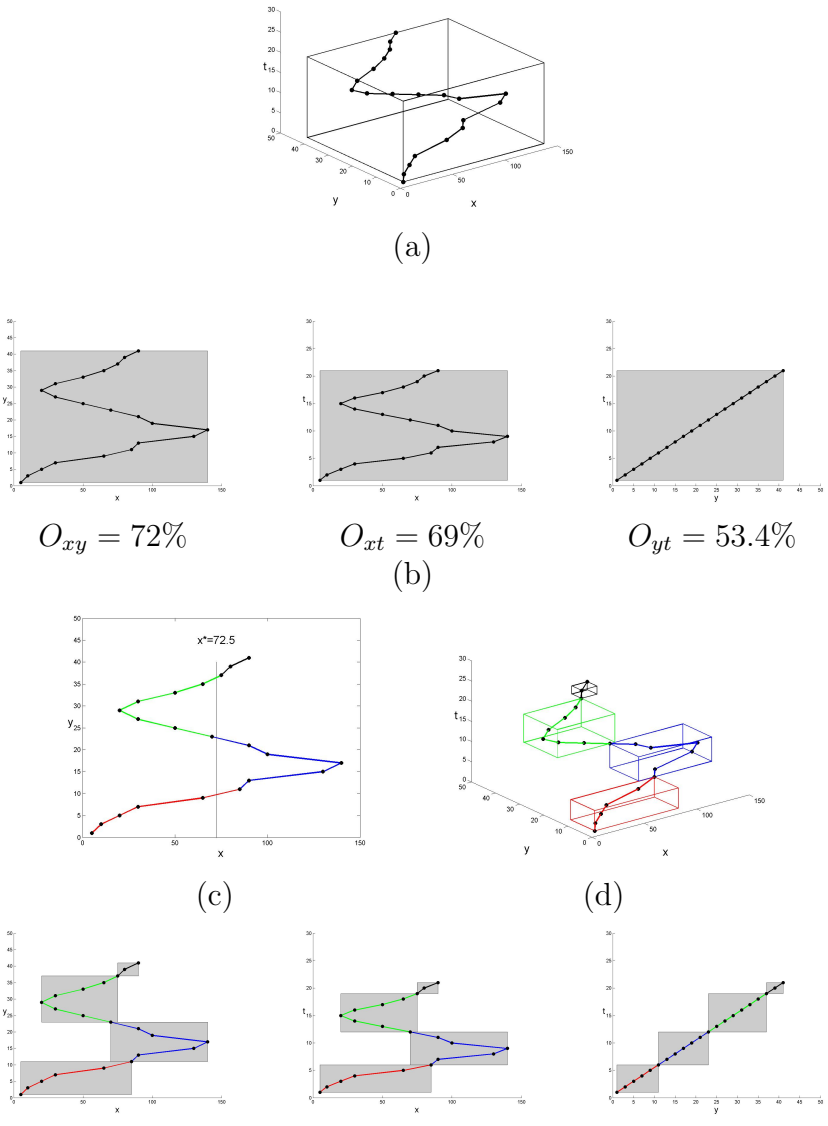


Figure 4.8 An overview of the segmentation algorithm.

$$\text{width} = x_{max} - x_{min} = 135 \tag{4.17}$$

$$\text{height} = y_{max} - y_{min} = 40, \tag{4.18}$$

assuming that x has been chosen as the split-dimension, the split-value will be easily obtained as follows:

$$x^* = \frac{x_{max} + x_{min}}{2} = 72.5 \tag{4.19}$$

$$\tag{4.20}$$

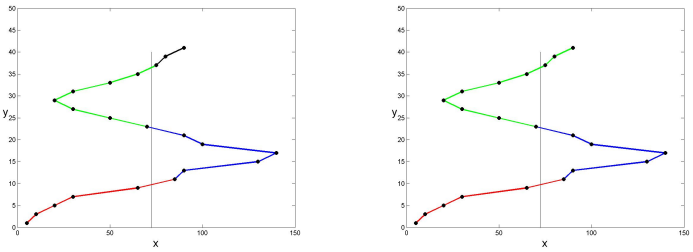
After the above described operations, we can now segment the trajectory unit; Figure 4.8.c shows the current iteration, that is the segmentation of the unit while, in Figure 4.8.d are shown, with different colors, the obtained 4 units with the corresponding MBRs. Last, Figure 4.8.e shows the projections of the obtained MBRs on each coordinate plane.

The algorithm ends when all the trajectory units cannot be further subdivided, since at least one of the stop conditions has been reached for each unit; in particular, we employ two stop criteria. First, we choose not to segment trajectory units whose MBR areas are smaller than a fixed percentage of the entire scenario (PA^{min}); furthermore, we do not segment a unit with less than PS^{min} points.

Finally, a further refinement is needed in our algorithm for handling with the formation of the last unit; in fact, when the generic unit U^i is splitted, it can happen that the last unit is only composed by a few points. In order to avoid this, an ad-hoc strategy is introduced and graphically explained in Figure 4.9.a: the last unit (the black one) is composed by three points, but LU^{min} is set to four. For this reason, this unit will be merged with the previous one (the green one, see Figure 4.9.b).

4.2.6 Experimental Evaluation

In order to characterize the efficiency of the proposed method, several queries were performed. We conducted our experiments on



(a) (b)

Figure 4.9 The effect of LU_{min} on a segmented unit.

a PC equipped with an Intel quad core CPU running at 2.66 GHz, using the 32 bit version of PostgreSQL 9.1 server and the 1.5.3 version of PostGIS. Data have been indexed using the standard bi-dimensional R-tree over GiST (Generalized Search Trees) indexes; as the specialized literature confirms, this choice guarantees higher performance in case of spatial queries with respect to the PostGIS implementation of R-trees.

We represent each trajectory unit as a tuple:

$$(\underline{ID}, \underline{UID}, U, MBR_{xy}, MBR_{xt}, MBR_{yt})$$

where ID is the moving objects identifier, UID identifies the trajectory unit, and U is the 3D trajectory unit, represented as a sequence of segments (a PostGIS 3D multi-line). Finally MBR_{xy}, MBR_{xt} and MBR_{yt} are the three unit's MBRs in each coordinate plane, xy, xt and yt respectively, represented as PostGis BOX geometries. Once data have been indexed, PostGIS provides a very efficient function to perform intersections between boxes and MBRs in a bi-dimensional space.

The experimental results have been obtained by testing the system performance on synthetic data, which have been generated as follows. Let W and H be the width and the height of our scene and S the temporal interval. Each trajectory T^i starting point is randomly chosen in our scene at a random time instant t_1^i; the trajectory length L^i is assumed to follow a Gaussian distribution, while the initial directions along the x axis and the y axis, respectively d_x^i and d_y^i, are randomly chosen. At each time step t,

we first generate the new direction, assuming that both d_x^i and d_y^i can vary with probability PI_x and PI_y respectively; subsequently, we choose the velocity along x and y at random. The velocity is expressed in pixels/seconds and is assumed to be greater than 0 and less than two fixed maximum, V_x^{max} and V_y^{max}. Therefore, the new position of the object can be easily derived; if it does not belong to our scene, new values for d_x and/or d_y are generated. We refer to the scene populated with trajectories as the *Scenario*. Table 4.10 reports the free parameters and the values for the creation of the 30 different scenarios used in our experiments as well as the parameters used by the segmentation algorithm. Note that the worst case, corresponding to the maximum values of T and L, results in 10^4 trajectories with 10^4 points, for a total of 10^8 points to store and process; these values are over and above if compared with many real world datasets.

Figure 4.10 The parameters used in our experiments.

Scene width (pixels)	10^4
Scene height (pixels)	10^4
Time interval length (secs)	10^5
Number of trajectories (T)	$\{1, 2, 3, 5, 10\} * 10^3$
Points in each trajectory (L)	$\{1, 2, 3, 5, 10\} * 10^3$
PI_x	5%
PI_y	5%
V_x^{max}	10 pixels/secs
V_y^{max}	10 pixels/secs
PA^{min}	1%
PS^{min}	100
LU^{min}	10

For the evaluation of the efficiency of the proposed segmentation algorithm, we generated and segmented 6000 trajectories with $L \in \{1000, 2000, 3000, 4000, 5000, 10000\}$; for each trajectory T^i we measured the number of obtained segments (N_{seg}^i) and the time needed to segment the trajectory (T_{seg}^i). Last, the obtained N_{seg}^i and T_{seg}^i are averaged over L, so obtaining $\overline{N_{seg}}(L)$ and $\overline{T_{seg}}(L)$. Figure 4.11 shows on the left the averaged number of segments ($\overline{N_{seg}}$) as L changes; not surprisingly we have, with very good ap-

proximation, that the number of segments linearly increases with L. On the right of Figure 4.11 is shown, in milliseconds, the averaged time $\overline{T_{seg}}$ needed to segment a trajectory with L points; with good approximation, $\overline{T_{seg}}$ quadratically increases with L.

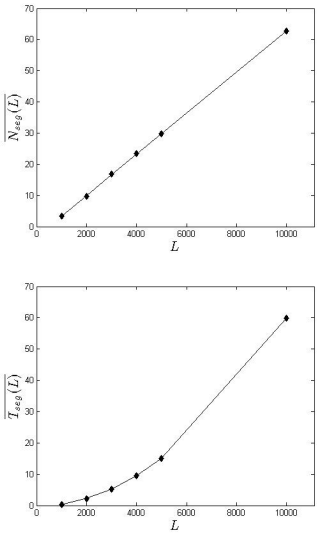


Figure 4.11 The performance of the segmentation algorithm.

The time needed to process a generic query (QT) is a function of at least 4 parameters, namely the number of trajectories T, the average trajectories' length L, the query cube dimension D_c, expressed as percentage of the entire scenario, and the position of the query box P_c:

$$QT = f(T, L, D_c, P_c). \tag{4.21}$$

Among the above parameters, P_c strongly influences the time needed to extract the trajectories as these are not uniformly distributed, especially in real world scenarios. In order to avoid the dependency on the query cube position, we decided to repeat the query a number of times inversely proportional to the query cube dimension, positioning the query cube in different positions, as shown in row N of Table 4.12; finally, results are averaged to obtain:

$$\overline{QT} = f(T, L, D_c). \tag{4.22}$$

D_c	1%	5%	10%	20%	30%	50%
N	200	40	20	10	7	4

Figure 4.12 Number N of times each query is repeated as D_c varies.

For the description of the experimental results we define three different set of experiments, obtained by fixing two of the three parameters T, L and D_c and showing the variation of $\overline{QT}$ with respect to the third (free) parameter; an example is shown in Figure 4.13, which expresses the values of $\overline{QT}$ with D_c and L fixed and T variable. Each diamond refers to the real value in seconds of $\overline{QT}$ for a given value of T; Figure 4.13 shows the curve which best interpolates the diamonds: in this case the approximation is linear, so obtaining a line. It is worth pointing out that, for the sake of readability, the figures for the experimental results will be expressed in a semi-log scale as, even not permitting to display part of the curves interpolating small values, it provides a greater comprehension of the system behavior for large values of the parameters.

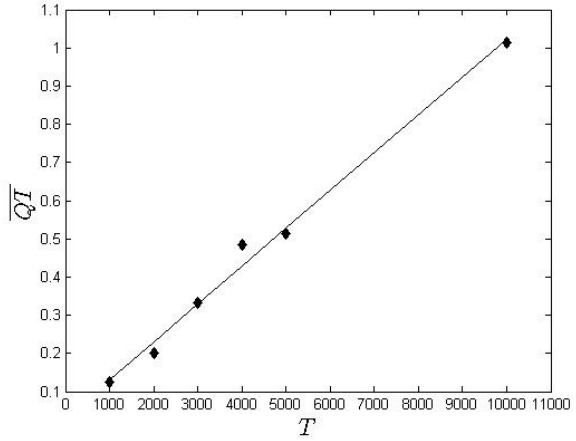
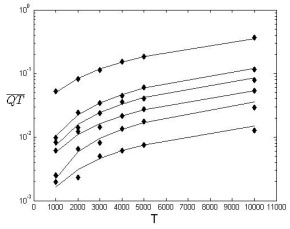


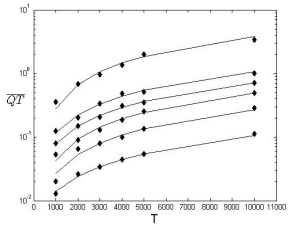
Figure 4.13 $\overline{QT}$ (in seconds) as the number of trajectories increases with $D_c = 5\%$ and $L = 5000$.

In Figure 4.14, $\overline{QT}$ relates to the variable number of trajectories T, for various values of D_c; the number of curves corresponds

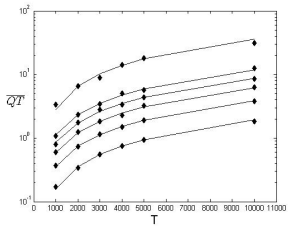
to the different fixed values of $L = \{1, 2, 3, 5, 10\} * 10^3$. The relationship between $\overline{QT}$ and T has been analyzed by polynomially approximating $QT(T)$: note that $\overline{QT}$ linearly increases with T, with a very small factor of approximation.



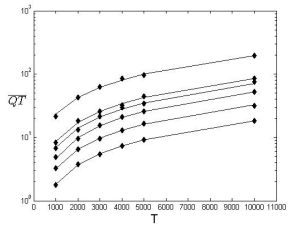
$D_c = 1\%$



$D_c = 5\%$



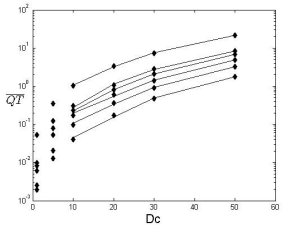
$D_c = 20\%$



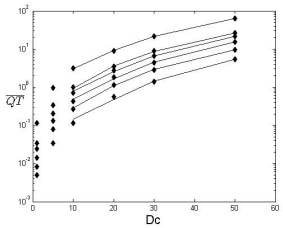
$D_c = 50\%$

Figure 4.14 $\overline{QT}$ (in seconds) as the number of trajectories increases having the number of points in each trajectory as parameter.

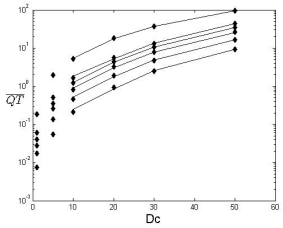
Diamond points in Figure 4.15 express $\overline{QT}$ in relation to the query box dimensions D_c and for $T = 3.000$ and $T = 10.000$; the number of curves corresponds to the different fixed values of $L = \{1, 2, 3, 5, 10\} * 10^3$. In this case we obtain that $\overline{QT}$ quadratically depends on D_c.



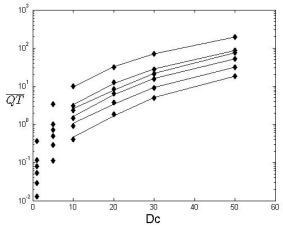
$T = 1000$



$T = 3000$

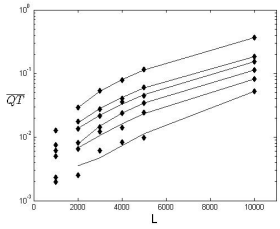


$T = 5000$

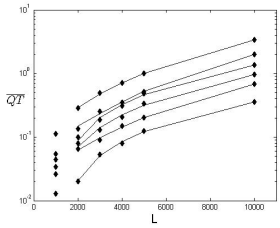


$T = 10000$

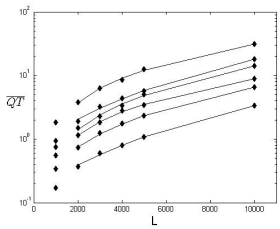
Figure 4.15 $\overline{QT}$ (in seconds) as the dimension of the querying cube (in percentage of the whole volume) increases and having L as parameter.



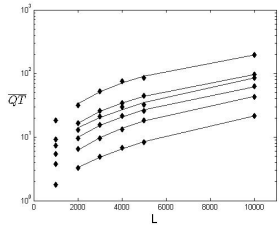
$D_c = 1\%$



$D_c = 5\%$



$D_c = 20\%$

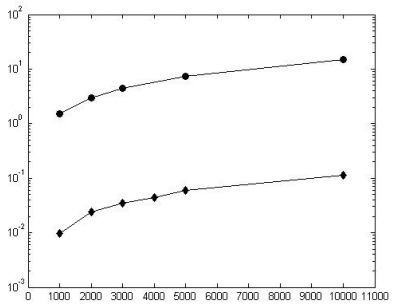


$D_c = 50\%$

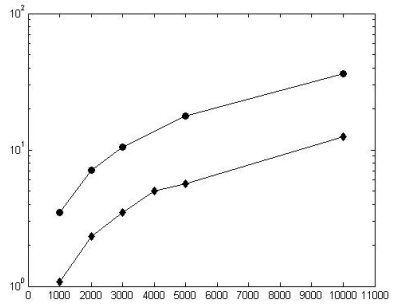
Figure 4.16 $\overline{QT}$ (in seconds) as the number of points in each trajectory increases and having the number of trajectories as parameter.

In Figure 4.16, the diamonds express $\overline{QT}$ as a function of L, for $D_c \in \{1\%, 5\%, 20\%, 30\%\}$, while the number of curves corresponds to the different fixed values of $T = \{1, 2, 3, 5, 10\} * 10^3$. Again we obtain that $\overline{QT}$ quadratically depends on L.

Finally, Figure 4.17 highlights the enhancement of the proposed indexing scheme as compared with the one presented in [dSV11], for $D_c \in \{1\%, 20\%\}$. In particular, the diamonds refer to the new method, while the circles refers to the previous one. Note that, thanks to a novel indexing strategy and segmentation algorithm, the system performance significantly improved while the redundancy in stored data has been significantly removed.



$D_c = 1\%$



$D_c = 20\%$

Figure 4.17 The results obtained with the solution proposed in [dSV11] (circles) compared with the results obtained with the solution here as T varies (L=5000).

Chapter 5

A System for Storing and Retrieving Huge Amount of Trajectory Data, Allowing Spatio-Temporal Dynamic Queries

One of the more interesting application contexts, in which there is growing need for techniques for the automatic analysis of spatio-temporal information, is traffic flow analysis. The security in our streets and highways has gained a crucial importance and many efforts are being made to increase the degree of confidence of the users. The need for security in traffic scenarios has many reasons, first of all the fact that these scenarios are daily interested by accidents, traffic queues, highway code violations, driving in the wrong lane or on the wrong side, and so on. In addition, traffic data analysis can also be employed to measure the system quality of service and improve the efficiency of our street networks.

In order to collect traffic data, cameras represent a suitable solution for their relative low cost of maintenance, the possibility of installing them virtually everywhere and, finally, the capability of analyzing more complex events.

In the context of camera-based traffic surveillance tools, in this chapter I will present an application of the approach analyzed in the previous chapter, contextualized to Traffic Flow Analysis and the formalization of innovative query typologies [dLSV12c]. The system is composed by three main components, sketched in the conceptual architecture of Figure 5.1:

1. The *Video Analytics Engine* (*VAE*) which, starting from the acquired video stream, detects the objects in the scene and then extracts the objects' trajectories together with their appearance information; as the tracking algorithm is outside of the scope of this thesis, the reader can refer to [ILGB12] for a detailed description of the tracking approach and the methodology for ensuring the consistent labeling of moving objects.

2. The *Storage Engine* (*SE*) is in charge of storing the extracted data by means of suited indexing strategies.

3. The *Retrieval Engine* (*RE*) allows to retrieve salient data for visualization and statistical purposes on the basis of the specific queries submitted by the user through a *Graphical User Interface* (GUI).

The scientific literature is recently enriching with papers dealing with the above mentioned phases or, in several cases, proposing whole architectures implementing a video-based analytics system. For example, in order to extract the motion trajectories of vehicles and pedestrians, different tracking algorithms have been proposed, as in [DMM+11] and [CWHF11]; many of these approaches focus on vehicles tracking and allow the traffic flow analysis in relation to the type of vehicle, properly using an object classification stage [GMMP02] [MT08a].

Despite the relatively high number of papers on trajectory extraction and vehicle and pedestrian classification, it is worth pointing out that only a modest attention has been devoted to storing and retrieving systems able to cope with very large amount of

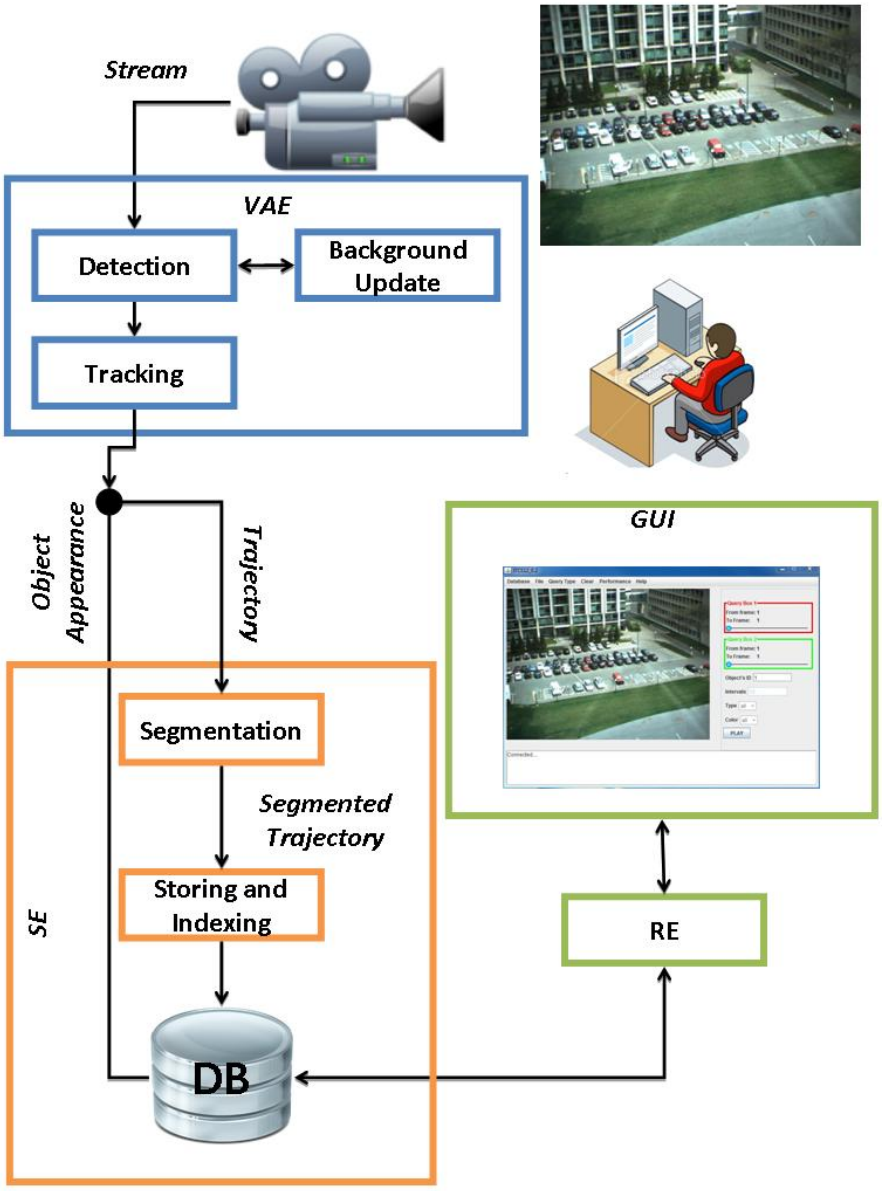


Figure 5.1 An overview of the architecture of the proposed system.

trajectory data and sufficiently general to deal with the needs of different application domains. This is an important and not negligible feature, especially when considering crowded real world scenarios (like highway intersections, city crossroads and important junctions). In these cases it is required that billions of trajectories must be stored and that, on this wide database, the user must be able to submit complex queries involving geometric and temporal data.

An example of these systems is [CSPZ03], which proposes a learning-based framework for capturing, modeling and indexing the spatio-temporal relations of the moving objects. Optimal route planning is dealt in [EJW05], presenting a web interface which supports shortest path or shortest time queries by use of the underlying geometrical structure. Kumar et al. [KRWS05] present a rule-based framework for activity detection in traffic videos and propose a novel Bayesian network approach for robust classification; they also handle with behavior recognition, analyzing the interactions between the moving objects and the scenario. Other approaches aim at building activity models by applying clustering algorithms [AMP10][GSF11].

The systems above cited are good examples of the potentiality of spatio-temporal queries in a given domain, but a step towards their geometrical generalization could significantly improve their usability; their main limitation, in fact, lies in the fact that the parameters characterizing the queries are mostly pre-determined and cannot be chosen by the user at query time. Their rationale is to have a system architecture devised and optimized for supporting a bunch of queries, each one referring to a given spatial area; this is sufficient to solve the corresponding retrieving problem, but the capability of choosing at query time (i.e. exactly when the query is thought) the area in which we are interested in is neglected. This is the case of similarity-based systems, which can be used for complex tasks like vehicle behavior analysis, but are limited by the impossibility to change the similarity metric. Other systems are devoted to vehicles trajectory clustering, which can be useful in particular contexts but, once performed the clustering, there is

not any possibility of modifying the classes and, as a consequence, the available functionalities.

An example of query supported by the above mentioned systems can be given by asking the system to retrieve all those vehicles flowing within the different lanes: this is achieved by segmenting the scene in different areas of interest (the lanes), so allowing the engine to store and retrieve the objects interesting those areas. The limitation becomes evident when we would determine, for example, the vehicles running exactly over the center line (so occupying two half lanes): in this case it is necessary to define at query time the area in which to find vehicles (a rectangle centered on the line dividing the two lanes).

While it is simple to imagine how much the flexibility degree of such a type of system can increase, it is not likewise to design a system architecture having these potentiality. The main contributions to this aim can be obtained by browsing the literature coming from the database field; for example, a widely adopted solution for bi-dimensional spatial indexing is based on R-trees [Gut84], which hierarchically organize geometric bi-dimensional data representing each object using its *Minimum Bounding Rectangle (MBR)*. Starting from Guttman's pioneering paper, many other indexing schemes able to handle with spatio-temporal data have been proposed for many application contexts, most of which are optimizations of R-trees [PJT+00], [SR03], [PAA+11], [CEP03].

However the cited approaches, even presenting efficient solutions from different perspectives, are typically not widely supported, both in commercial and freely available products. In this chapter will be presented an efficient application of the *SE*, which is based on a novel indexing scheme and a segmentation stage for the optimization of the indexes' selectivity [DLSV12b]; both these last contributions have been the concern of the previous chapter. Once data have been collected and properly stored, our system allows to efficiently solve, by means of the *RE*, *Dynamic Spatio-Temporal* (*DST*) queries, e.g. queries finding all the trajectories passing through an area defined directly within the query (i.e. at query time). As it will be shown in the next section, other inter-

esting kind of queries can be formulated in terms of specializations of *DST* queries.

5.1 Query Processing

Different techniques exist for processing queries over spatio-temporal information but, as already observed, a fundamental operation, which is common to almost any kind of trajectory query, consists in the verification of the intersection between a given set of trajectories and one or more spatio-temporal volumes. The scientific literature is rich of definitions for queries, often providing different names for the same query type. However, even though a commonly accepted classification is not given, most authors refer to the classification proposed by Pfoser [Pfo02], which distinguishes two fundamental query types:

- coordinate-based queries,
- trajectory-based queries.

Coordinate-based queries can be seen as the classic spatio-temporal queries as they focus on the position of the moving objects in time; on the contrary, trajectory-based queries are based on the use of other information, such as the proximity to the areas of interest, the presence of obstacles, or the use of derived information like speed, acceleration, direction: the former are called topological queries, while the latter are called navigational queries. Coordinate-based queries can be used to retrieve information about precise time instants (point queries), time intervals (range queries, also called time interval queries) or to retrieve the k closest objects' trajectories to a given position in a given time instant (k-nn queries).

Another significant differentiation among queries can be made by subdividing them on the basis of the temporal information:

* past queries,

* current queries,

* future queries.

Past queries are referred to events that have already occurred and use past data to retrieve information; current queries aim to retrieve information about the current time instant on the basis of past information; finally, future queries are used to predict future events on the basis of the information known at the moment.

Given the above criteria for distinguishing the various query types, lots of queries have been proposed, which can be considered as belonging to one typology or another or, more often, to both typologies. For instance, in [PAA+11] three different queries have been proposed:

- full trajectory pattern matching queries, which compare each location between query pattern and moving object trajectories and return only the trajectories that match fully;

- Partly trajectory pattern matching queries, whose aim is not the exact matching of the trajectory and the query pattern, but the matching of sub-trajectories;

- Recurrent trajectory pattern matching queries, which aim to solve the inverse problem of finding the trajectories which are common to a given set of mobile objects.

Similarly in [RH11] whole pattern, subpattern and reverse subpattern matching queries are presented.

Other queries have been formalized to address continuous moving objects, which essentially means that the objects is moving and the queries are performed in real-time. Continuous Within Queries [HL11] can be used to find the moving objects whose distances from the moving query object are less than or equal to a user-given distance d_ϵ at each time instant; analogously, Continuous Obstructed Range queries [LGL11] consider the presence of obstacles when calculating the distance between objects: at each

time instant they provide the set of object points whose distance from the query point is less than a threshold (the radius).

The above cited query typologies are only some examples of the numerous ones proposed in literature, nevertheless they are useful to understand the versatility of a query type able to modify its parameters; moreover, if the variation of these parameters can occur at query time, then we will be able to use the same query for any of the above objectives.

5.2 Dynamic Spatio-Temporal Queries: Formulation

After the application of a tracking algorithm [ILGB12], we have obtained the set of objects moving in the scene, each one represented as the triple:

$$O^k =< I^k, T^k, A^k >$$

where I^k is the object's identifier, T^k is its 3D trajectory and A^k is the vector of characteristics representing additional information about object's appearance (as color, type, size, etc). T^k is in turn defined as a sequence of spatio-temporal points:

$$T^k =< P_1^k, P_2^k, ..., P_N^k >$$

where the generic point $P_i^k = (x_i^k, y_i^k, t_i^k)$ represents the spatial location (x_i^k, y_i^k) of an object at the time instant t_i^k. From now on, we will use the line segments model [PJT+00], each segment being the line connecting two consecutive points.

A *Dynamic Spatio-Temporal (DST)* query allows to detect all those objects passing through a given spatial area in a given time interval. In the contest of a traffic management system, *DST* queries can be used for different retrieval purposes. Typical uses are: *"verify whether a given vehicle was crossing a given intersection between 4.30 pm and 5.30 pm yesterday"*, or *"find all the vehicles passing by an area within San Peter's Square on 14th March*

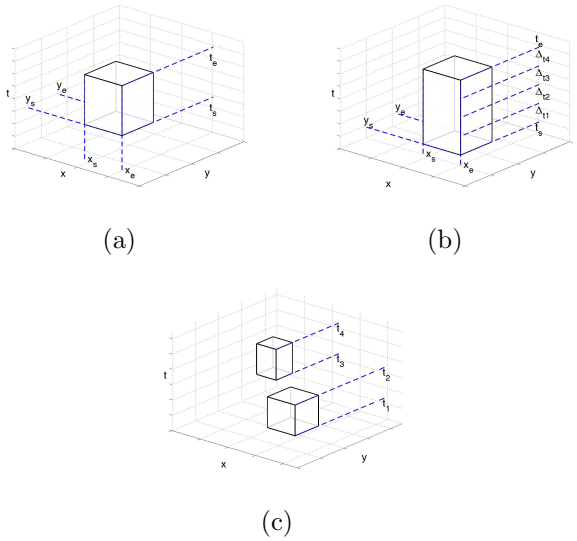


Figure 5.2 Geometric interpretation of different types of query: *DST* (a), *F-DST* (b) and *M-DST* (c).

2011 in the morning". More specific information can be obtained by augmenting *DST* queries with additional information about objects' appearance: *"find the number of red trucks leaving highway I59 from exit 14 running across the middle lane between 9-12 am yesterday"*, or *"verify whether a given motorcycle was walking through a level crossing between 6-8 pm yesterday"*.

A key concept to be stressed is that the area and the time interval of interest are assumed to be defined at query time. Here, we think to the area as a rectangle with coordinates (x_s, y_s) and (x_e, y_e) while $[t_s, t_e]$ are the starting and final time instants. According to this assumption, each *DST* query can be associated to a query box B (Figure 5.2(a)).

Another query type, hereinafter named as Flow-DST *(F-DST)*, is well suited for analyzing the traffic flow in the observed scene; it allows to retrieve information of the type *"find the number of vehicles passing by a given area (dynamically defined at query time) on highway S14 each hour from 8 am to 6 pm yesterday"*, or *"find the number of vehicles passing by a given toll-house each six hours*

during the last two days". From a geometric point of view, a *F-DST* can be seen as the application of various *DST* queries so as to obtain results at fixed time intervals, as shown in Figure 5.2(b).

A further type of query, called Multi-DST *(M-DST)*, has been defined to retrieve more complex and structured information; examples of this query type are *"find the number of vehicles that have passed by a given intersection and then have exited by a given off-ramp two days ago"* and *"find all the yellow vehicles passing by highway I-55 and then by Interstate 60 from 2 to 4 pm today"*. As Figure 5.2(c) shows, this query typology can be seen as the application of two (or more) *DST* queries, typically having temporally successive query boxes.

The solution to all the above queries, from a geometric perspective, involves the application of the intersection operations: verify if at least one of each trajectory's segments intersects the query box. This means that this formulation benefits from the indexing scheme and the segmentation presented in this dissertation and introduced and discussed in the previous chapter.

For a better comprehension of the use of these queries within a video analysis system, we can think to the video sequences extracted from the monitoring cameras as sequences of images depicting the scene. When submitting a DST query to the system we are asking it to consider a given temporal interval, a given spatial sequence and verify some properties; once this properties have been verified, we are interested in obtaining the complete list of objects possessing that property. From an image analysis perspective, this means the extraction of the frames of interest and the verification of the intersection operations within them, like shown in figure 5.3.

5.3 Experimental results

The database has been implemented by storing the trajectories' data in Postgres using the extension PostGIS; data are indexed using the standard bi-dimensional R-tree over *Generalized Search*

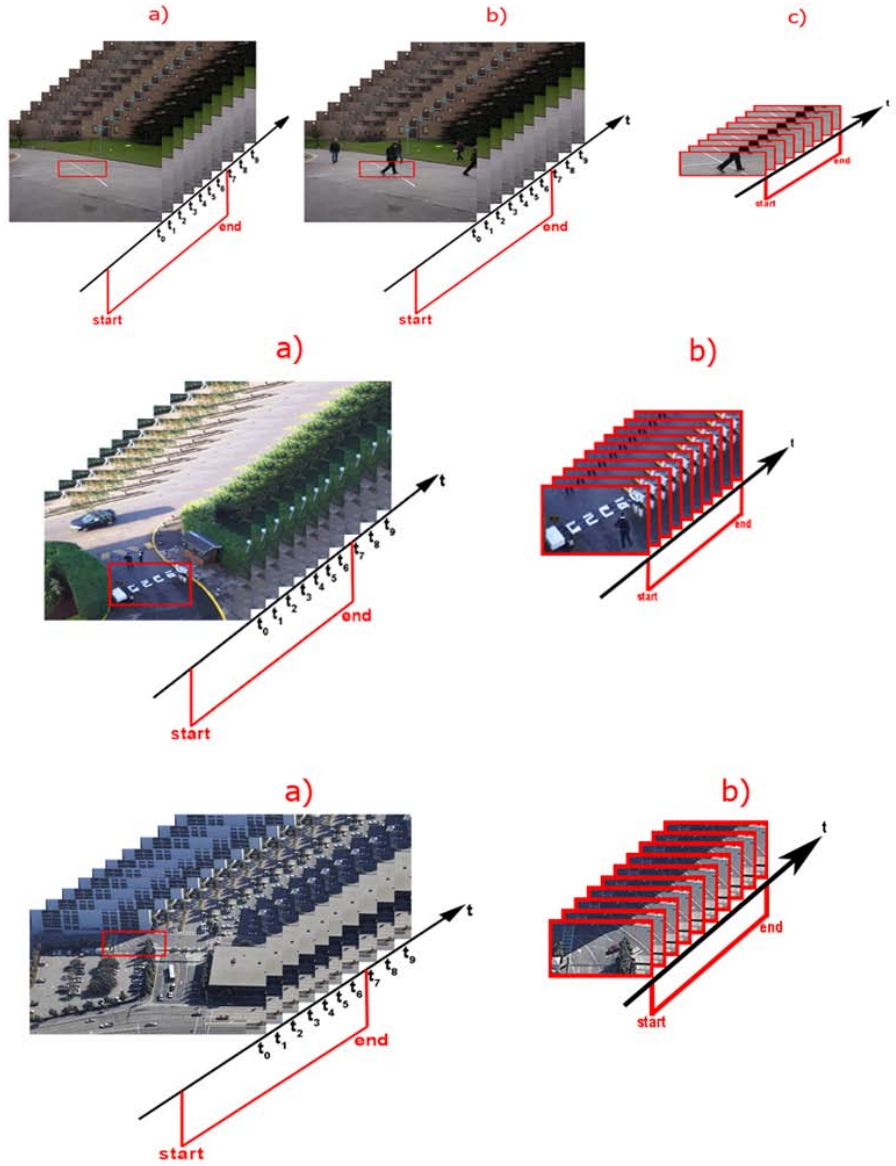


Figure 5.3 The application of a three DST queries from an image analysis perspective.

Trees (GiST) indexes since, how the specialized literature highlights, this choice guarantees higher performance in case of spatial queries, if compared with the PostGIS implementation of R-trees.

The experiments were conducted on a PC equipped with an Intel quad core CPU running at 2.66 GHz, using the 32 bit version of the PostrgreSQL 9.1 server and the 1.5 version of PostGIS.

For an intuitive construction of the queries, it has been designed and implemented a GUI (Figure 5.4), thanks to which the user can define all the geometric and temporal constraints.

In particular, for a *DST* query, the user specifies (using a sliding bar) the temporal interval and the spatial region of interest by drawing a rectangle on the 2D plane. As for *F-DST* queries, the user is also required to insert the number of intervals required for the analysis. Finally, the definition of *M-DST* queries (with $M = 2$) require the definition of the different query boxes: the user has to draw two different rectangles and specifies two different time intervals for each query box.

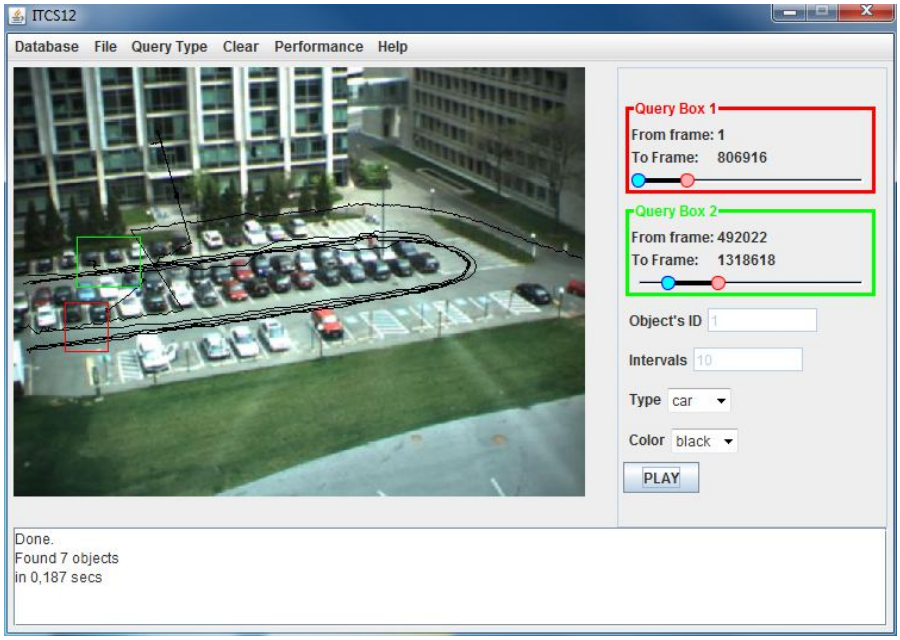


Figure 5.4 The Graphical user Interface designed for the Traffic Analysis Application.

As stated previously in this dissertation, each query can be represented as a 3D cube and it is straightforward to observe

that the time needed to process a generic DTS query (QT) is a function of many parameters, since it is clearly dependent on the number of trajectories T, on the trajectories' length L, on the query cube dimension D_c (expressed as percentage of the volume[1] $V = W_S * H_S * I$), and on the position of the query box P_c. In particular, P_c strongly influences the time needed to extract the trajectories as, in real world scenarios, the trajectories are not uniformly distributed. To avoid the dependence on the query cube position, we decided to repeat the query a number of times inversely proportional to the query cube dimension, as shown in columns D_c and N of Table 5.5; finally, results have been averaged to obtain:

$$\overline{QT}_{DST} = f(T; L; D_c) \tag{5.1}$$

D_c	N	T^1	T^2	T^3	$\overline{QT}_{DST}$
1%	200	0.003	0.010	0.009	0.022
5%	40	0.007	0.064	0.115	0.186
10%	20	0.013	0.154	0.320	0.487
20%	10	0.038	0.533	1.383	1.954
30%	7	0.097	1.566	4.014	5.673
50%	4	0.173	5.878	14.924	20.975

Figure 5.5 Averaged time (in seconds) to solve a DST query.

5.3.1 Real-World Dataset

To test the performance of our indexing scheme, we first used the freely available MIT trajectory dataset [GWNM08], obtained from a parking lot scene within five days[2]; the dataset is composed of approximately $4 * 10^4$ trajectories with 108.81 points in each trajectory (on average). At loading time, each trajectory has been

[1] W_S and H_S are the width and the height of our scene while I be the whole time interval we are interested in.

[2] Vehicles' color and type have been assigned at random.

segmented using $PA^{min} = 1$ and $PS^{min} = 100$, so obtaining approximately $1.92 * 10^6$ segments with 23.71 points in each segment (on average). Table 5.5 shows $\overline{QT}_{DST}$ (in seconds) as D_c varies.

$\overline{QT}_{DST}$ results from the sum of three terms: T^1 is the time needed to select the segments whose bounding box intersect the query box on each bi-dimensional plane, T^2 is the time to clip the segments while T^3 is the time needed to extract the whole trajectory, so obtaining:

$$T^Q_{DST} = T_1 + T_2 + T_3, \tag{5.2}$$

Starting from this consideration it is possible, through simple considerations, to obtain the expected performance of both *F-DST* and *M-DST* queries.

F-DST is the application of many *DST* queries in sequence. Suppose, for example, to ask our system to retrieve the number of vehicles passing through Interstate 55 from 5 pm and 6 pm each ten minutes; what our system would do is to perform six *DST* queries, one for each 10 minutes interval between 5 pm and 6 pm, only counting the number of instances satisfying the query in each interval and giving, as the final result, the total sum of the count. This means that, for each *DST* query, the system performs only the intersection and the clipping stages, giving the total count as the final result of the *F-DST* query.

According to the above considerations, the expected $\overline{QT}_{F-DST}$ of a *F-DST* query asking for the number of objects intersecting query box B in the time interval (t_1, t_n) each of the N time intervals is defined as:

$$\overline{QT}_{F-DST} = N * (T_1 + T_2), \tag{5.3}$$

in which T_1 and T_2 are the intersection and clipping times respectively for each of the N *DST* queries; finally, the sum of the N count values is given as a result.

*M-DST*s are slightly more complex as the multiple query boxes are virtually independent. In this case, the system processes each query box as a single *DST* query, applying the intersection and

clipping operations for each query box; finally, the extraction phase provides the trajectories satisfying the *M-DST* query.

The expected $\overline{QT}_{M-DST}$ of a *M-DST* query with M bounding boxes $(B_1, B_2, ..., B_M)$, each having its time interval, is:

$$\overline{QT}_{M-DST} = M * (T_1 + T_2) + T_3, \tag{5.4}$$

in which T_1 and T_2 are the intersection and clipping times respectively for each of the M *DST* queries and T_3 is the time needed to extract the trajectories' result set.

5.3.2 Synthetic Data

Last, we tested our information retrieval system with synthetic data, which have been generated as follows. Let W_S and H_S be the width and the height of our scene and I be the time interval we are interested in. Each trajectory starting point is randomly chosen in our scene at a random time instant t_1; the trajectory length L is assumed to be fixed while the initial directions along the x axis and the y axis, respectively d_x and d_y, are randomly chosen. At each time step t, we first generate the new direction, assuming that d_x and d_y can vary with probability PI_x and PI_y respectively; subsequently, we randomly chose the velocity along x and y. The velocity is expressed in pixels/seconds and is assumed to be greater than 0 and less than two fixed maxima, V_x^{max} and V_y^{max}. Therefore the new position of the object can be easily derived; if it does not belong to our scene, new values for d_x and/or d_y are generated.

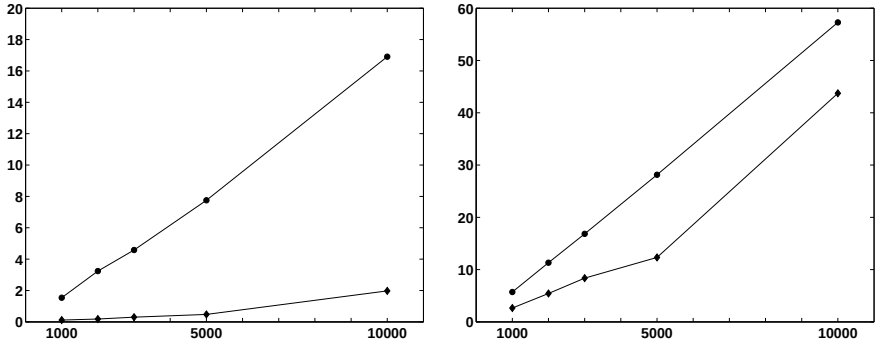
We have used the parameters reported in Table 5.6 to compare the current system with the one proposed in [dSV11] that, while still using off-the-shelf bi-dimensional indexes, does not segment trajectories and introduces some redundancy in the data to be stored; $PA^{min} = 1\%$ and $PS^{min} = 300$ have been used in the segmentation stage.

Figure 5.7 shows the $\overline{QT}_{DST}$ (in seconds) obtained for small sized query cubes ($DC = 5\%$, (a)) as well as for big sized ones ($DC = 30\%$, (b)). There is a significant improvement for small

W_S (pixels)	10^4
H_S (pixels)	10^4
I (seconds)	10^5
L (points)	5000
T	$\in \{1, 2, 3, 5, 10\} * 10^3$
PI_x	5%
PI_y	5%
V_x^{max} (pixels/secs)	10
V_y^{max} (pixels/secs)	10

Figure 5.6 The parameters used to generate synthetic data.

query cubes and an interesting improvement for large cubes. Intuitively, this is due to the fact that $\overline{QT}_{DTS}$ is lower bounded by the time needed to extract trajectories.



(a) (b)

Figure 5.7 $\overline{QT}_{DST}$ (in seconds) as T varies having $L = 5000$. Circles represent results obtained with the system described in [dSV11] while diamonds are the ones obtained with the current system [dLSV12c] for $D_C = 5\%$ (a) and for $D_C = 30\%$ (b).

Conclusions and Future Directions

In this thesis I have described the work carried out to address a very up-to-date problem: the need to efficiently collect and organize the spatio-temporal data coming from different sensors and use these data to provide location-based services for smart communities. One of the key observations is the constant growth of the number of smart sensors, which are modern devices capable of collecting our position in real-time. Given the proliferation of such devices, together with the acquisition peripherals already existing, the amount of collected spatio-temporal data demands for ad-hoc strategies for its storage and indexing.

Once extracted the trajectories of the object moving in the scene, we can use Moving Object Databases for handling these data. One of the first choices is related to their representation, given the need to obtain a compact representation while still preserving the contextual information about objects' dynamics. The central focus, indeed, has been the proposal of a novel indexing scheme able to exploit the characteristics of Spatial Database Management Systems.

As existing systems provide very efficient 2D indexes, but do not support efficient indexing and retrieval of three-dimensional data, it has been proposed a method for the reformulation of any N-dimensional problem in terms of bi-dimensional problems, so allowing the use of the available efficient indexes.

Moreover, in order to increase the selectivity of these indexes, which is strongly connected to the Minimum Bounding Rectangle

representation, we introduced a segmentation stage for the reduction of the redundancy produced by this representation.

An application of the proposed approach has been given in chapter 5 by presenting a traffic flow analysis system; in the context of intelligent transportation systems, we have formalized three specialized queries based on an innovative query type, the Dynamic Spatio-Temporal query, whose most important feature is the possibility to choose its parameters at query time.

The efficiency of the proposed approach has been proven by experimental evaluation both on synthetic and real-world data. In particular, in order to generate synthetic data which were meaningful also from a topological perspective, we presented a simulation model based on Social Force Models for the simulation of pedestrian behavior.

The approach presented in this dissertation covers some important aspects for the design of automatic activity analysis systems. The main contribution of this work lies in the proposal of a universal methodology for the reformulation of a high dimensional problem in terms of bi-dimensional problems, as this means that existing efficient solutions can be adopted; starting from this achievement, the approach has been used as the basis for the design of a system for the storage, indexing and efficient retrieval of three-dimensional trajectories using spatio-temporal queries, whose parameters can be actualized at query time.

We think that the approach proposed in this dissertation still possesses some potentialities; in the near future, in fact, we hope to apply it to a very interesting application context, the analysis of customers' trajectories to retrieve information about their habits, their preferences and potential discomfort areas related to the displacement of the products for sale, the obstacles and the points of interest (like placards, promotional advertisements, etc.).

Open issues about the proposed approach are mainly related to the complexity of the extraction of trajectories from the database, as it is the most onerous operation at the moment. To this aim, further improvements in the performance will be hopefully achieved by applying the clipping algorithm in parallel to each candidate

trajectory; this step can be easily implemented using multi-threading, in order to take advantage of multi-core and multi-processors systems.

Finally, a further reduction of the extraction time can be obtained by introducing strategies aiming to compress the stored spatio-temporal data.

Bibliography

[AAM+05] S. Atev, H. Arumugam, O. Masoud, R. Janardan, and N.P. Papanikolopoulos, *A vision-based approach to collision prediction at traffic intersections*, Intelligent Transportation Systems, IEEE Transactions on **6** (2005), no. 4, 416–423.

[AMJP05] S. Atev, O. Masoud, R. Janardan, and N. Papanikolopoulos, *A collision prediction system for traffic intersections*, Intelligent Robots and Systems, 2005.(IROS 2005). 2005 IEEE/RSJ International Conference on, IEEE, 2005, pp. 169–174.

[AMP06] Stefan Atev, Osama Masoud, and Nikos Papanikolopoulos, *Learning traffic patterns at intersections by spectral clustering of motion trajectories*, Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on, IEEE, 2006, pp. 4851–4856.

[AMP10] Stefan Atev, Grant Miller, and Nikolaos P Papanikolopoulos, *Clustering of vehicle trajectories*, Intelligent Transportation Systems, IEEE Transactions on **11** (2010), no. 3, 647–657.

[BAT05] D. Biliotti, G. Antonini, and J.P. Thiran, *Multilayer hierarchical clustering of pedestrian trajectories for automatic counting of people in video sequences*, Application of Computer Vision, 2005.

WACV/MOTIONS'05 Volume 1. Seventh IEEE Workshops on, vol. 2, IEEE, 2005, pp. 50–57.

[BBS06] N. Brandle, D. Bauer, and S. Seer, *Track-based finding of stopping pedestrians-a practical approach for analyzing a public infrastructure*, Intelligent Transportation Systems Conference, 2006. ITSC'06. IEEE, IEEE, 2006, pp. 115–120.

[BCB03] Maren Bennewitz, G Cielniak, and W Burgard, *Utilizing learned motion patterns to robustly track persons*, Proc. of Joint IEEE Int. Workshop on VS-PETS, Citeseer, 2003, pp. 102–109.

[BD02] Gary R. Bradski and James W. Davis, *Motion segmentation and pose recognition with motion history gradients*, Machine Vision and Applications **13** (2002), 174–184 (English).

[BI05a] O. Boiman and M. Irani, *Detecting irregularities in images and in video*, Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on, vol. 1, oct. 2005, pp. 462 – 469.

[BI05b] Oren Boiman and Michal Irani, *Detecting irregularities in images and in video*, Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on, vol. 1, IEEE, 2005, pp. 462–469.

[BK00] M. Brand and V. Kettnaker, *Discovery and segmentation of activities in video*, Pattern Analysis and Machine Intelligence, IEEE Transactions on **22** (2000), no. 8, 844–851.

[BKS07] Faisal I Bashir, Ashfaq A Khokhar, and Dan Schonfeld, *Object trajectory-based activity classification and recognition using hidden markov models*, Image Processing, IEEE Transactions on **16** (2007), no. 7, 1912–1919.

[BMPI05] N.D. Bird, O. Masoud, N.P. Papanikolopoulos, and A. Isaacs, *Detection of loitering individuals in public transportation areas*, Intelligent Transportation Systems, IEEE Transactions on **6** (2005), no. 2, 167–177.

[BQKS05] Faisal Bashir, Wei Qu, Ashfaq Khokhar, and Dan Schonfeld, *Hmm-based motion recognition system using segmented pca*, Image Processing, 2005. ICIP 2005. IEEE International Conference on, vol. 3, IEEE, 2005, pp. III–1288.

[CC92] W Chan and F Chin, *Approximation of polygonal curves with minimum number of line segments*, Algorithms and Computation (1992), 378–387.

[CC03] Amit K. Roy Chowdhury and Rama Chellappa, *A factorization approach for activity recognition*, Computer Vision and Pattern Recognition Workshop, 2003. CVPRW '03. Conference on, vol. 4, june 2003, p. 41.

[CCK02] Fangxiang Cheng, W.J. Christmas, and J. Kittler, *Recognising human running behaviour in sports video sequences*, Pattern Recognition, 2002. Proceedings. 16th International Conference on, vol. 2, 2002, pp. 1017 – 1020 vol.2.

[CEP03] V Prasad Chakka, Adam C Everspaugh, and Jignesh M Patel, *Indexing large trajectory data sets with seti*, Ann Arbor **1001** (2003), 48109–2122.

[Cha06] C.Y. Chan, *Defining safety performance measures of driver-assistance systems for intersection left-turn conflicts*, Intelligent Vehicles Symposium, 2006 IEEE, IEEE, 2006, pp. 25–30.

[CMWM10] Philippe Cudre-Mauroux, Eugene Wu, and Samuel Madden, *Trajstore: An adaptive storage system for*

very large trajectory data sets, Data Engineering (ICDE), 2010 IEEE 26th International Conference on, IEEE, 2010, pp. 109–120.

[CSPZ03] Shu-Ching Chen, Mei-Ling Shyu, Srinivas Peeta, and Chengcui Zhang, *Learning-based spatio-temporal vehicle tracking and indexing for transportation multimedia database systems*, Intelligent Transportation Systems, IEEE Transactions on **4** (2003), no. 3, 154–167.

[CWHF11] Yen-Lin Chen, Bing-Fei Wu, Hao-Yu Huang, and Chung-Jui Fan, *A real-time vision system for nighttime vehicle detection and traffic surveillance*, Industrial Electronics, IEEE Transactions on **58** (2011), no. 5, 2030–2044.

[dLSV12a] A. d'Acierno, M. Leone, A. Saggese, and M. Vento, *An efficient bi-dimensional indexing scheme for three-dimensional trajectories*, International Journal On Advances in Intelligent Systems **5** (2012), no. 3 and 4, 220–233.

[DLSV12b] A. D'Acierno, M. Leone, A. Saggese, and M. Vento, *Efficient extraction of motion flow data from a repository of three-dimensional trajectories using bi-dimensional indexes*, IMMM 2012, The Second International Conference on Advances in Information Mining and Management, 2012, pp. 79–84.

[dLSV12c] Antonio d'Acierno, Marco Leone, Alessia Saggese, and Mario Vento, *A system for storing and retrieving huge amount of trajectory data, allowing spatio-temporal dynamic queries*, Intelligent Transportation Systems (ITSC), 2012 15th International IEEE Conference on, IEEE, 2012, pp. 989–994.

[DMM+11] Helgo Dyckmanns, Richard Matthaei, Markus Maurer, Bernd Lichte, Jan Effertz, and D Stuker, *Object tracking in urban intersections based on active use of a priori knowledge: Active interacting multi model filter*, Intelligent Vehicles Symposium (IV), 2011 IEEE, IEEE, 2011, pp. 625–630.

[dSV11] Antonio d'Acierno, Alessia Saggese, and Mario Vento, *A redundant bi-dimensional indexing scheme for three-dimensional trajectories*, IMMM 2011, The First International Conference on Advances in Information Mining and Management, 2011, pp. 73–78.

[EJW05] Stefan Edelkamp, Shahid Jabbar, and Thomas Willhalm, *Geometric travel planning*, Intelligent Transportation Systems, IEEE Transactions on **6** (2005), no. 1, 5–16.

[ETK+03] H.-L. Eng, K.-A. Toh, A.H. Kam, J. Wang, and W.-Y. Yau, *An automatic drowning detection surveillance system for challenging outdoor pool environments*, Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on, vol. 1, oct. 2003, pp. 532 –539.

[FNPB03] Jeff P. Foster, Mark S. Nixon, and Adam Prugel-Bennett, *Automatic gait recognition using area-based metrics*, Pattern Recognition Letters **24** (2003), no. 14, 2489 – 2497.

[Fre03] Elias Frentzos, *Indexing objects moving on fixed networks*, Advances in Spatial and Temporal Databases (2003), 289–305.

[FVDFH12] James D Foley, Andries Van Dam, Steven K Feiner, and John F Hughes, *Computer graphics: principles and practice*, Addison-Wesley, 2012.

[GF09] Valérie Gouaillier and A Fleurant, *Intelligent video surveillance: Promises and challenges*, Technological and commercial intelligence report, CRIM and Technôpole Defence and Security **456** (2009), 468.

[GMMP02] Surendra Gupte, Osama Masoud, Robert FK Martin, and Nikolaos P Papanikolopoulos, *Detection and classification of vehicles*, Intelligent Transportation Systems, IEEE Transactions on **3** (2002), no. 1, 37–47.

[GSF11] Maxime Gariel, Ashok N Srivastava, and Eric Feron, *Trajectory clustering and an application to airspace monitoring*, Intelligent Transportation Systems, IEEE Transactions on **12** (2011), no. 4, 1511–1524.

[Gut84] Antonin Guttman, *R-trees: a dynamic index structure for spatial searching*, vol. 14, ACM, 1984.

[GWNM08] Eric Grimson, Xiaogang Wang, Gee-Wah Ng, and Keng Teck Ma, *Trajectory analysis and semantic region modeling using a nonparametric bayesian model*.

[HFMV02] Dirk Helbing, Illes J Farkas, Peter Molnar, and Tamás Vicsek, *Simulation of pedestrian crowds in normal and evacuation situations*, Pedestrian and evacuation dynamics **21** (2002).

[HL11] Yuan-Ko Huang and Lien-Fa Lin, *Continuous within query in road networks*, Wireless Communications and Mobile Computing Conference (IWCMC), 2011 7th International, IEEE, 2011, pp. 1176–1181.

[HM95] Dirk Helbing and Peter Molnar, *Social force model for pedestrian dynamics*, Physical review E **51** (1995), no. 5, 4282.

[HMFB01] Dirk Helbing, Peter Molnar, Illes J Farkas, and Kai Bolay, *Self-organizing pedestrian movement*, Environment and planning B **28** (2001), no. 3, 361–384.

[HTWM04] Weiming Hu, Tieniu Tan, Liang Wang, and Steve Maybank, *A survey on visual surveillance of object motion and behaviors*, Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on **34** (2004), no. 3, 334–352.

[HXF+06] Weiming Hu, Xuejuan Xiao, Zhouyu Fu, Dan Xie, Tieniu Tan, and Steve Maybank, *A system for learning statistical motion patterns*, IEEE Transactions on Pattern Analysis and Machine Intelligence **28** (2006), no. 9, 1450–1464.

[HXF+07] Weiming Hu, Dan Xie, Zhouyu Fu, Wenrong Zeng, and Steve Maybank, *Semantic-based surveillance video retrieval*, Image Processing, IEEE Transactions on **16** (2007), no. 4, 1168–1181.

[HXT04] W. Hu, D. Xie, and T. Tan, *A hierarchical self-organizing approach for learning the patterns of motion trajectories*, Neural Networks, IEEE Transactions on **15** (2004), no. 1, 135–144.

[HXTM04] W. Hu, D. Xie, T. Tan, and S. Maybank, *Learning activity patterns using fuzzy self-organizing neural network*, Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on **34** (2004), no. 3, 1618–1626.

[HXX+04] W. Hu, X. Xiao, D. Xie, T. Tan, and S. Maybank, *Traffic accident prediction using 3-d model-based vehicle tracking*, Vehicular Technology, IEEE Transactions on **53** (2004), no. 3, 677–694.

[HYCH06] J.W. Hsieh, S.H. Yu, Y.S. Chen, and W.F. Hu, *Automatic traffic surveillance system for vehicle tracking*

and classification, Intelligent Transportation Systems, IEEE Transactions on **7** (2006), no. 2, 175–187.

[IB00] Y.A. Ivanov and A.F. Bobick, *Recognition of visual activities and interactions by stochastic parsing*, Pattern Analysis and Machine Intelligence, IEEE Transactions on **22** (2000), no. 8, 852 –872.

[ILGB12] G. Iovane, M. Leone, P. Giordano, and E. Benedetto, *Information fusion based on fuzzy fourier transform of fingerprints for image watermarking*, IEEE Workshop on Biometric Measurements and Systems for Security and Medical Applications, IEEE, 2012, pp. pp–46.

[JH96] N. Johnson and D. Hogg, *Learning the distribution of object trajectories for event recognition*, Image and Vision Computing **14** (1996), no. 8, 609–615.

[JH99] Y.K. Jung and Y.S. Ho, *Traffic parameter extraction using video-based vehicle tracking*, Intelligent Transportation Systems, 1999. Proceedings. 1999 IEEE/IEEJ/JSAI International Conference on, IEEE, 1999, pp. 764–769.

[JJS04] I.N. Junejo, O. Javed, and M. Shah, *Multi feature path modeling for video surveillance*, Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on, vol. 2, IEEE, 2004, pp. 716–719.

[JWW+04] L. Jiao, Y. Wu, G. Wu, E.Y. Chang, and Y.F. Wang, *Anatomy of a multicamera video surveillance system*, Multimedia systems **10** (2004), no. 2, 144–163.

[KGT07] Ramsin Khoshabeh, Tarak Gandhi, and Mohan M Trivedi, *Multi-camera based traffic flow characterization & classification*, Intelligent Transportation

Systems Conference, 2007. ITSC 2007. IEEE, IEEE, 2007, pp. 259–264.

[KKL+05] S. Kamijo, H. Koo, X. Liu, K. Fujihira, and M. Sakauchi, *Development and evaluation of real-time video surveillance system on highway based on semantic hierarchy and decision surface*, Systems, Man and Cybernetics, 2005 IEEE International Conference on, vol. 1, IEEE, 2005, pp. 840–846.

[KMIS00] S. Kamijo, Y. Matsushita, K. Ikeuchi, and M. Sakauchi, *Traffic monitoring and accident detection at intersections*, Intelligent Transportation Systems, IEEE Transactions on **1** (2000), no. 2, 108–118.

[Koh90] Teuvo Kohonen, *The self-organizing map*, Proceedings of the IEEE **78** (1990), no. 9, 1464–1480.

[KRWS05] P. Kumar, S. Ranganath, H. Weimin, and K. Sengupta, *Framework for real-time behavior interpretation from traffic video*, Intelligent Transportation Systems, IEEE Transactions on **6** (2005), no. 1, 43–53.

[KSR+04] A. Kale, A. Sundaresan, A.N. Rajagopalan, N.P. Cuntoor, A.K. Roy-Chowdhury, V. Kruger, and R. Chellappa, *Identification of humans using gait*, Image Processing, IEEE Transactions on **13** (2004), no. 9, 1163 –1173.

[LC51] Kurt Lewin and Dorwin Cartwright, *Field theory in social science*.

[LCST06] MH-Y Liao, Duan-Yu Chen, Chih-Wen Sua, and Hsiao-Rang Tyan, *Real-time event detection and its application to surveillance systems*, Circuits and Systems, 2006. ISCAS 2006. Proceedings. 2006 IEEE International Symposium on, IEEE, 2006, pp. 4–pp.

[LGL11] Zhicheng Li, Yunjun Gao, and Yansheng Lu, *Continuous obstructed range queries in spatio-temporal databases*, System Science, Engineering Design and Manufacturing Informatization (ICSEM), 2011 International Conference on, vol. 2, IEEE, 2011, pp. 267–270.

[LHH06] X. Li, W. Hu, and W. Hu, *A coarse-to-fine strategy for vehicle motion trajectory clustering*, Pattern Recognition, 2006. ICPR 2006. 18th International Conference on, vol. 1, IEEE, 2006, pp. 591–594.

[Lim09] RNCOS E-Services Private Limited, *World gps market forecast to 2013*, Research and Markets, 2009.

[LKF05] Taras I Lakoba, David J Kaup, and Neal M Finkelstein, *Modifications of the helbing-molnar-farkas-vicsek social force model for pedestrian evolution*, Simulation **81** (2005), no. 5, 339–352.

[LN06] Fengjun Lv and Ramakant Nevatia, *Recognition and segmentation of 3-d human action using hmm and multi-class adaboost*, Computer Vision - ECCV 2006 (AleÅ¡ Leonardis, Horst Bischof, and Axel Pinz, eds.), Lecture Notes in Computer Science, vol. 3954, Springer Berlin Heidelberg, 2006, pp. 359–372.

[LY00] Andrew HS Lai and Nelson HC Yung, *Vehicle-type identification through automated virtual loop assignment and block-based direction-biased motion estimation*, Intelligent Transportation Systems, IEEE Transactions on **1** (2000), no. 2, 86–97.

[ME02] D. Makris and T. Ellis, *Path detection in video surveillance*, Image and Vision Computing **20** (2002), no. 12, 895–903.

[ME05] Dimitios Makris and Tim Ellis, *Learning semantic scene models from observing activity in visual*

surveillance, IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics **35** (2005), no. 3, 397–408.

[MH00] R.J. Morris and D.C. Hogg, *Statistical models of object interaction*, International Journal of Computer Vision **37** (2000), no. 2, 209–215.

[MHK06] T.B. Moeslund, A. Hilton, and V. Krüger, *A survey of advances in vision-based human motion capture and analysis*, Computer vision and image understanding **104** (2006), no. 2, 90–126.

[MK07] A Malevanets and R Kapral, *Continuous-velocity lattice-gas model for fluid flow*, EPL (Europhysics Letters) **44** (2007), no. 5, 552.

[MMZ05] S. Messelodi, C.M. Modena, and M. Zanin, *A computer vision system for the detection and classification of vehicles at urban road intersections*, Pattern Analysis & Applications **8** (2005), no. 1, 17–31.

[MNBSV06] J. Melo, A. Naftel, A. Bernardino, and J. Santos-Victor, *Detection and classification of highway lanes using vehicle motion trajectories*, Intelligent Transportation Systems, IEEE Transactions on **7** (2006), no. 2, 188–200.

[MNPT05] Yannis Manolopoulos, Alexandros Nanopoulos, Apostolos N Papadopoulos, and Yannis Theodoridis, *R-trees: Theory and applications*, Springer, 2005.

[MT08a] Brendan Tran Morris and Mohan Manubhai Trivedi, *Learning, modeling, and classification of vehicle track patterns from live video*, Intelligent Transportation Systems, IEEE Transactions on **9** (2008), no. 3, 425–437.

[MT08b] B.T. Morris and M.M. Trivedi, *A survey of vision-based trajectory learning and analysis for surveillance*, Circuits and Systems for Video Technology, IEEE Transactions on **18** (2008), no. 8, 1114–1127.

[MY86] Gerard Medioni and Yoshio Yasumoto, *Corner detection and curve representation using cubic b-splines*, Robotics and Automation. Proceedings. 1986 IEEE International Conference on, vol. 3, IEEE, 1986, pp. 764–769.

[NK06] A. Naftel and S. Khalid, *Classifying spatiotemporal object trajectories using unsupervised learning in the coefficient feature space*, Multimedia Systems **12** (2006), no. 3, 227–238.

[OH00] J. Owens and A. Hunter, *Application of the self-organising map to trajectory classification*, Visual Surveillance, 2000. Proceedings. Third IEEE International Workshop on, IEEE, 2000, pp. 77–83.

[PA04a] S. Park and J.K. Aggarwal, *A hierarchical bayesian network for event recognition of human actions and interactions*, Multimedia Systems **10** (2004), no. 2, 164–179.

[PA04b] Sangho Park and J.K. Aggarwal, *Semantic-level understanding of human actions and interactions using event hierarchy*, Computer Vision and Pattern Recognition Workshop, 2004. CVPRW '04. Conference on, june 2004, p. 12.

[PAA+11] J Priyadarshini, P AnandhaKumar, M Aparna, JK Geetha, and N Shobana, *Indexing and querying technique for dynamic location updates using r kd trajectory trie tree*, Recent Trends in Information Technology (ICRTIT), 2011 International Conference on, IEEE, 2011, pp. 1143–1148.

[PCDN09] Nicola Piotto, Nicola Conci, and Francesco GB De Natale, *Syntactic matching of trajectories for ambient intelligence applications*, Multimedia, IEEE Transactions on **11** (2009), no. 7, 1266–1275.

[Pea01] Karl Pearson, *Liii. on lines and planes of closest fit to systems of points in space*, The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science **2** (1901), no. 11, 559–572.

[PF06] C Piciarelli and GL Foresti, *On-line trajectory clustering for anomalous events detection*, Pattern Recognition Letters **27** (2006), no. 15, 1835–1842.

[Pfo02] Dieter Pfoser, *Indexing the trajectories of moving objects*, IEEE Data Engineering Bulletin **25** (2002), no. 2, 3–9.

[PJ07] Patrick De Pelsmacker and Wim Janssens, *The effect of norms, attitudes and habits on speeding behavior: Scale development and model building and estimation*, Accident Analysis & Prevention **39** (2007), no. 1, 6 – 15.

[PJT+00] Dieter Pfoser, Christian S Jensen, Yannis Theodoridis, et al., *Novel approaches to the indexing of moving object trajectories*, Proceedings of VLDB, Citeseer, 2000, pp. 395–406.

[PT97] Dimitris Papadias and Yannis Theodoridis, *Spatial relations, minimum bounding rectangles, and spatial data structures*, International Journal of Geographical Information Science **11** (1997), no. 2, 111–138.

[PT07] S. Park and M.M. Trivedi, *Multi-person interaction and activity analysis: a synergistic track-and body-level analysis framework*, Machine Vision and Applications **18** (2007), no. 3, 151–166.

[RH11] Gook-Pil Roh and Seung-won Hwang, *Tpm: Supporting pattern matching queries for road-network trajectory data*, Proceedings of the 14th International Conference on Extending Database Technology, ACM, 2011, pp. 554–557.

[RJ93] Lawrence Rabiner and Biing-Hwang Juang, *Fundamentals of speech recognition*.

[Ros97] Paul L. Rosin, *Techniques for assessing polygonal approximations of curves*, Pattern Analysis and Machine Intelligence, IEEE Transactions on **19** (1997), no. 6, 659–666.

[RR05] N. Robertson and I. Reid, *Behaviour understanding in video: a combined method*, Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on, vol. 1, oct. 2005, pp. 808 – 815 Vol. 1.

[RRB06] N. Robertson, I. Reid, and M. Brady, *Behaviour recognition and explanation for video surveillance*, Crime and Security, 2006. The Institution of Engineering and Technology Conference on, IET, 2006, pp. 458–463.

[RSEN05] Slobodan Rasetic, Jörg Sander, James Elding, and Mario A Nascimento, *A trajectory splitting model for efficient spatio-temporal indexing*, Proceedings of the 31st international conference on Very large data bases, VLDB Endowment, 2005, pp. 934–945.

[RSJ04] P. Remagnino, AI Shihab, and GA Jones, *Distributed intelligence for multi-camera visual surveillance*, Pattern recognition **37** (2004), no. 4, 675–689.

[SB00] N. Sumpter and A. Bulpitt, *Learning spatio-temporal patterns for predicting object behaviour*, Image and Vision Computing **18** (2000), no. 9, 697–704.

[Sch01] Andreas Schadschneider, *Cellular automaton approach to pedestrian dynamics-theory*, arXiv preprint cond-mat/0112117 (2001).

[SD03] T.N. Schoepflin and D.J. Dailey, *Dynamic camera calibration of roadside traffic management cameras for vehicle speed estimation*, Intelligent Transportation Systems, IEEE Transactions on **4** (2003), no. 2, 90–98.

[SG99] Chris Stauffer and W Eric L Grimson, *Adaptive background mixture models for real-time tracking*, Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on., vol. 2, IEEE, 1999.

[SG00a] C. Stauffer and W. Grimson, *Learning patterns of activity using real-time tracking*, IEEE Transactions on Pattern Analysis and Maching Intelligence **22(8)** (2000), 747–757.

[SG00b] C. Stauffer and W.E.L. Grimson, *Learning patterns of activity using real-time tracking*, Pattern Analysis and Machine Intelligence, IEEE Transactions on **22** (2000), no. 8, 747–757.

[SPZO+10] I Sandu Popa, Karine Zeitouni, Vincent Oria, Dominique Barth, and Sandrine Vial, *Parinet: A tunable access method for in-network trajectories*, Data Engineering (ICDE), 2010 IEEE 26th International Conference on, IEEE, 2010, pp. 177–188.

[SPZO+11] Iulian Sandu Popa, Karine Zeitouni, Vincent Oria, Dominique Barth, and Sandrine Vial, *Indexing in-network trajectory flows*, The VLDB Journal - The International Journal on Very Large Data Bases **20** (2011), no. 5, 643–669.

[SR03] Zhexuan Song and Nick Roussopoulos, *Seb-tree: An approach to index continuously moving objects*, Mobile Data Management, Springer, 2003, pp. 340–344.

[SSL07] N. Saunier, T. Sayed, and C. Lim, *Probabilistic collision prediction for vision-based automated road safety analysis*, Intelligent Transportation Systems Conference, 2007. ITSC 2007. IEEE, IEEE, 2007, pp. 872–878.

[SSS05] Y. Sheikh, M. Sheikh, and M. Shah, *Exploring the space of a human action*, Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on, vol. 1, oct. 2005, pp. 144 – 149 Vol. 1.

[TGH05] Mohan M Trivedi, Tarak L Gandhi, and Kohsia S Huang, *Distributed interactive video arrays for event capture and enhanced situational awareness*, Intelligent Systems, IEEE **20** (2005), no. 5, 58–66.

[TPE07] Pablo Cristian Tissera, Marcela Printista, and Marcelo Luis Errecalde, *Evacuation simulations using cellular automata*, Journal of Computer Science and Technology **7** (2007), no. 1, 14–20.

[TVS96] Y Theoderidis, Michael Vazirgiannis, and Timos Sellis, *Spatio-temporal indexing for large multimedia applications*, Multimedia Computing and Systems, 1996., Proceedings of the Third IEEE International Conference on, IEEE, 1996, pp. 441–448.

[VKG02] Michail Vlachos, George Kollios, and Dimitrios Gunopulos, *Discovering similar multidimensional trajectories*, Data Engineering, 2002. Proceedings. 18th International Conference on, IEEE, 2002, pp. 673–684.

[Vla12] J. Vlahos, *Come anticipare il crimine*, Le Scienze **n. 522** (2012), 82–87.

[VMP03] H. Veeraraghavan, O. Masoud, and N.P. Papanikolopoulos, *Computer vision algorithms for intersection monitoring*, Intelligent Transportation Systems, IEEE Transactions on **4** (2003), no. 2, 78–89.

[VRCC03] N. Vaswani, A. Roy Chowdhury, and R. Chellappa, *Activity recognition using the dynamics of the configuration of interacting objects*, Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on, vol. 2, june 2003, pp. II – 633–40 vol.2.

[Wei92] Ulrich Weidmann, *Transporttechnik der fußgänger*.

[WL05] T Warren Liao, *Clustering of time series data - a survey*, Pattern Recognition **38** (2005), no. 11, 1857–1874.

[WMG07] X. Wang, X. Ma, and E. Grimson, *Unsupervised activity perception by hierarchical bayesian models*, Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on, IEEE, 2007, pp. 1–8.

[WNTH04] Liang Wang, Huazhong Ning, Tieniu Tan, and Weiming Hu, *Fusion of static and dynamic body biometrics for gait recognition*, Circuits and Systems for Video Technology, IEEE Transactions on **14** (2004), no. 2, 149 – 158.

[WSH04] Wen Wang, Andreas Stolcke, and Mary P Harper, *The use of a linguistically motivated language model in conversational speech recognition*, Acoustics, Speech, and Signal Processing, 2004. Proceedings.(ICASSP'04). IEEE International Conference on, vol. 1, IEEE, 2004, pp. I–261.

[WTNH03] Liang Wang, Tieniu Tan, Huazhong Ning, and Weiming Hu, *Silhouette analysis-based gait recognition for human identification*, Pattern Analysis and Machine Intelligence, IEEE Transactions on **25** (2003), no. 12, 1505 – 1518.

[XG05] T. Xiang and S. Gong, *Video behaviour profiling and abnormality detection without manual labelling*, Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on, vol. 2, IEEE, 2005, pp. 1238–1245.

[YF05] W. Yan and D.A. Forsyth, *Learning the behavior of users in a public space through video tracking*, Application of Computer Vision, 2005. WACV/MOTIONS'05 Volume 1. Seventh IEEE Workshops on, vol. 1, IEEE, 2005, pp. 370–377.

[YmSxSL05] Hui Yu, Guang min Sun, Wen xing Song, and Xiao Li, *Human motion recognition based on neural network*, Communications, Circuits and Systems, 2005. Proceedings. 2005 International Conference on, vol. 2, may 2005, pp. 2 vol. (xviii+1411).

[YY05] Xiang Yu and Simon X Yang, *A study of motion recognition from video sequences*, Computing and Visualization in Science **8** (2005), no. 1, 19–25.

[ZSV04] Hua Zhong, Jianbo Shi, and Mirkó Visontai, *Detecting unusual activity in video*, Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on, vol. 2, IEEE, 2004, pp. II–819.