Evaluation of Probabilistic Queries in Moving Objects Databases

Talel Abdessalem
Ecole Nationale Supérieure des Télécommunications – UMR CNRS 5141
46, Rue Barrault
75634 Paris Cedex 13 – France
Talel.Abdessalem@enst.fr

Laurent Decreusefond
Ecole Nationale Supérieure des Télécommunications – UMR CNRS 5141
46, Rue Barrault
75634 Paris Cedex 13 – France
Laurent.Decreusefond@enst.fr

José Moreira
Departamento de Electrónica e Telecomunicações da Universidade de Aveiro / IEETA
Campo Universitário de Santiago
3810 - 193 Aveiro – Portugal
jmoreira@det.ua.pt

ABSTRACT

The representation of moving objects in spatial database systems has become an important research topic in recent years. As it is not realistic to track and store the location of objects at every time instant, one of the issues in this domain has to do with handling uncertainty in the location of moving objects. In this paper, we propose three statistical methods for computing probabilistic estimates about the location of a moving object at a certain time and show how to use them for evaluating probabilistic range queries. The focus is on applications dealing with the spatiotemporal behavior of non-network constrained moving objects, for monitoring or data-mining purposes, for instance.

General Terms

Measurement, Reliability

Categories and Subject Descriptors

H.2.8 [**Database management**]: Database Applications—*Spatial databases and GIS*

Keywords

Spatio-temporal databases, moving objects, spatio-temporal uncertainty

1. INTRODUCTION

The spatiotemporal databases research community is giving particular attention to moving objects applications. Real-time systems, using recent information about object's movement for anticipation of events in near future, or historical systems, recording information about object's movement during large periods of time for monitoring or data-mining purposes, are noteworthy examples. Both cases require functionality allowing answering questions of the kind where and when. However, as it is not possible to continuously monitor and record the location of moving objects, the knowledge that may be captured and stored by computer systems is only a partial representation of the spatiotemporal behavior of real-world objects. This gives rise to the problem of handling uncertainty in the location of moving objects.

There are several proposals to cope with this problem by extending spatiotemporal query languages with adequate semantics for dealing with uncertainty. The goal is to incorporate functionalities that allow answering questions such as "Which were the moving objects that have been within a certain area during some period of time, with a probability of at least 80%". Answering such kind of queries requires the use of stochastic processes to model the reality. It is then necessary to determine the aspects of the reality which one wishes to take into account. The recorded positions of the objects are certainly necessary but one can also use other information such as the maximum speed.

Building stochastic processes satisfying the constraints of position can be made using existing theory on conditioned processes [3]. Building processes satisfying the constraint of maximum speed can be made using the reflected processes theory [12, 7]. However, there are no methods to build models where the two types of constraints are satisfied simultaneously.

This paper presents two approaches for dealing with the computation of probabilistic estimates about the validity of answers to spatiotemporal queries. The focus is on the history of object's movement and it is assumed that the objects move freely in space. One approach grounds on the definition of density functions over an area bounding the set of all possible locations of a moving object at a certain time instant. The other uses stochastic processes and a limited number of constraints to generate trajectories between the positions recorded for an object.

The remaining part of this paper is organized as follows. Section 2 presents an overview of current proposals for dealing with uncertainty in the location of moving objects. Section 3 depicts the two main approaches that we have delineated to solve the proposed problem and presents three methods for computing probabilistic estimates about the location of moving objects. Section 4 compares the proposals in terms of expressivity and ability for dealing with the different semantics proposed in the spatiotemporal query languages literature. Finally, section 5 concludes the paper.

MobiDE'06, June 25, 2006, Chicago, Illinois, USA.

2. RELATED WORK

The location of moving objects is a continuous function of time. However, measurement instruments are inherently discrete and are not able to continuously capture the location of moving objects. Consequently, movement knowledge as it can be stored in a database system is just a partial representation of the actual spatiotemporal behavior of real-world objects. In such conditions, the answers to user queries about object's movement may not be consistent with real-world events [9, 14].

2.1 Uncertainty of past, present and future positions of moving objects

There are two main approaches for dealing with partial representations of movement information.

The first approach focuses on uncertainty about the future spatiotemporal behavior of moving objects [11, 18, 14]. This approach addresses the needs of real-time applications and location-based services: real-time traffic control, real-time mobile workforce management, digital battlefields, etc. These systems make use of speed patterns information in the construction of future movements and uncertainty is fixed in advance. This allows avoiding frequent updates of the database. In fact, the database is not updated as long as the actual object's movement deviation from its expected location is less than a previously fixed threshold.

The second approach [9, 8] focuses on past movements. It addresses the needs of mining applications of spatiotemporal data: traffic mining, environment monitoring, etc. In this case, the movement of an object is represented as a sequence of observations and uncertainty in the location of moving objects is closely connected with the size of the time intervals between consecutive observations.

In both cases, there are physical constraints on the movement of objects allowing limiting uncertainty in their position at a certain time. In the following, we will designate the set of all possible locations for a moving object at a certain time by *lens area*, as suggested by Pfoser and Jensen [9]. For simplicity of presentation, we will also assume that the objects move on a two-dimensional space.

When dealing with the anticipation of future events, the lens area is a circle centered on the expected location of the object O' (Figure 1) [16, 18]. The point O corresponds to the location where the object has been observed for the last time and the arrow is a velocity vector. The cross points to the actual location of the moving object. The circle bounds the maximum deviation allowed for an object at a given time instant. Objects are committed to send a location update when the deviation reaches that bound.

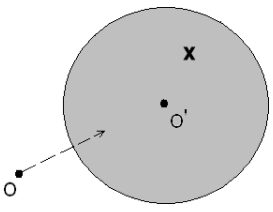
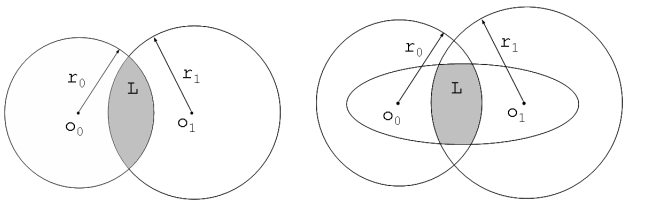


Figure 1: Lens area in a real-time system

For past movements, when the maximum velocity of the moving objects is known in advance, the lens area for a time instant t between two consecutive observations $o_0 = (t_0, p_0)$ and $o_1 = (t_1, p_1)$, is given by the intersection of two circles (Figure 2.1 (a)). The variables t_0 and t_1 stand for the time of observations, and p_0 and p_1 for the position of the object at those time instants. The radius of the circle in the left is $r_0 = v_{max} \times (t - t_0)$ and for the circle in the right is $r_1 = v_{max} \times (t_1 - t)$.

The lens area for a time interval $\langle t_a, t_b \rangle$, such that $t_0 < t_a < t_b < t_1$, is given by the intersection of an ellipsis, covering the set of all possible locations for the moving object during $\langle t_0, t_1 \rangle$, and two circles (figure 2.1 (b)). The radius of circle in the left is $r_a = V_{\max} \times (t_b - t_0)$ and for circle in the right is $r_b = V_{\max} \times (t_1 - t_a)$. For a detailed explanation of these formulas please refer to [8].



(a) Lens area at a time instant *(b) Lens area for a time interval*

Figure 2: Lens areas for monitoring and data mining systems

Considering that we have methods that allow estimating probabilities about the location of a moving object at any time instant, the evaluation of spatiotemporal query expressions could than be augmented with probabilistic estimates about the validity of answers to users queries. Thus, it would be possible to answer queries such as "Which are the planes for which the probability of being inside Area C within 5 minutes is at least 40%?", or "Which were the ships that were in a certain area during a given time interval, with a probably of at least 60%?".

The notion of probabilistic range queries for real-time applications has been introduced by [18] and developed in [13, 14]. Recently, this notion was extended to deal with nearest-neighbor queries about the movement of network and non-network constrained moving objects [5]. The communication cost for updating the database was also investigated [16].

On the other hand, [9, 8] have focused on dealing with uncertainty in the location of non-network constrained moving objects for monitoring and data mining systems. In addition, [6] discusses how the uncertainty of network constrained moving objects can be reduced by using reasonable modeling methods and location update policies. Finally, [10] added fuzziness in object location and considered the case of moving objects that may change their geometry in time.

For an overview of moving objects databases literature please refer to [2].

2.2 Probabilistic estimates about the location of moving objects

Current proposals for evaluation of estimations about the probability of presence of a moving object within a region at a certain time instant are based on the definition of density functions over lens areas. For real-time applications, the shape of lens areas is a circle [16, 18, 17]. The density function may be uniform, assuming that all locations within the circle have the same probability, or other user-defined function based on the distance towards the center of the cir-

cle, as proposed in [5]. It is also considered that the lens areas could have a complex geometry, but it is not shown how to evaluate the density functions over those geometries.

The geometry of lens areas in monitoring and data mining systems is more complex than a circle, as figure 2.1(b) shows. In [10], it is proposed to define an uniform distribution over the lens area. Then, the probability of presence of a moving object within a given region at a certain time instant is 30% if at least 30% of the lens area is within that region. However, there are certain locations (e.g., those near the center of the lens area) with higher probabilities then others (e.g.,those near the boundary of the lens area). Hence, using uniform distributions may not be adequate in such cases.

In addition, those methods based on density functions are not adequate to deal with queries where the temporal domain is a time interval. Notice that the probability of presence of a moving object within a given area at a certain time, should be less or equal than the probability of presence of the same moving object within the same area during any time that includes the previous one. However, this may not be the case when using this kind of methods.

For instance, consider figure 3 where A is an area of interest, L' is the lens area calculated for the movement object at a time instant t *(1st case)* and L'' is a lens area for the same object during time interval $\langle t_a, t_b\rangle \supset t$ *(2nd case).*

(1st case) Lens area for t — *(2nd case)* Lens area for $\langle t_a, t_b\rangle \supset t$

Figure 3: Intersection of a lens area (L) with an area of interest (A)

Considering that the distribution of probability in the lens area is uniform, then the probability of presence of the moving object within A at time t is $P(t) = \frac{area(A \cap L')}{area(L')}$ and the probability of being within A during time $\langle t_a, t_b\rangle$ is $P(\langle t_a, t_b\rangle) = \frac{area(A \cap L'')}{area(L'')}$. As $A \cap L'$ is equal to $A \cap L''$ and the area of L' is lesser than the area of L'' then, against evidence, $P(t)$ is greater than $P(\langle t_a, t_b\rangle)$.

In this paper we investigate solutions for these two issues and we propose a novel method for defining non-uniform distributions over lens areas with complex geometries. We also propose a new approach for estimating probabilities that is adequate to handle queries about the movement of objects where the temporal domain is a time interval.

3. PROBABILISTIC REASONING

This section presents the statistical tools that we propose for the evaluation of probabilistic estimates about the location of moving objects. We will denote the probability of presence of a moving object within a given region during a certain time tm, simply as $P(tm)$. We only consider the past objects' movement and we assume that it is represented as an ordered sequence of observations, denoted $\{(t, p)\}$, where p is a two-dimensional value denoting the location of the object at time instant t. We also consider that the objects move freely in space with no obstacles or networks constraining their movements.

We have investigated two main guidelines for the implementation of the proposed statistical tools: the *point-based* and *trajectory-based* approaches.

3.1 Point-based approach

As referred in section 2.2, there are several authors suggesting using a density function to estimate the probability of presence of a moving object at each point inside the lens area. Then, assuming that t denotes a time instant, the values of $P(t)$ may be calculated using the weight of the intersection of the lens area with the region considered, as shown in formula (1).

$$P(t) = \frac{W_{LensArea(t) \cap Region}}{W_{LensArea(t)}} \quad (1)$$

The main issue is the definition of a density function over a complex form, such as a lens area. Since we can easily define a density function over a circle, we propose to perform an anamorphosis of a lens area into a circle, as follows:

1. We define a local coordinates system for the lens area to simplify the formulation of the lens area equation.
2. We define an anamorphosis of the lens area into a circle of radius 1 and we associate a density of probability to that circle.
3. We define a transformation of the density of probability over the circle of radius 1 into a density of probability over the lens area.
4. We complete the process by the evaluation of $P(t)$.

3.1.1 Lens area equations

The lens area is given by the intersection of two circles. If one circle contains the other, the result of the intersection is the smaller circle. This situation may arise for time instants in the neighborhood of the instants of observations. Otherwise, the result of the intersection is an area similar to the one depicted in figure 4.

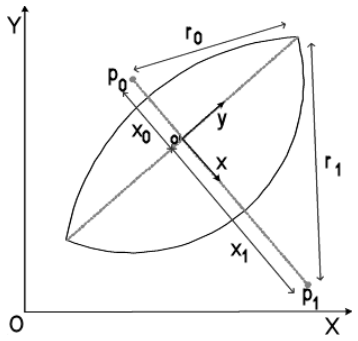


Figure 4: Lens area parameters

The lens area in figure 4 delimits the set of all possible locations for a moving object at time instant t, between two consecutive locations: p_0, observed at instant t_0, and p_1, observed at t_1. As presented in section 2.1, the left centered circle delimits the lens area of the object during the time interval $[t_0, t]$. The right centered circle delimits the lens area of this object during the time interval $[t, t_1]$.

The radiuses of the circles depend on the maximum velocity (V_{max}) defined previously:

$$r_0 = V_{max} \times (t - t_0) \qquad r_1 = V_{max} \times (t_1 - t) \tag{2}$$

To make the formulation of the lens area parameters easier, we use a local coordinates system based on the lens area axes (see figure 4). Let us consider d as the distance between p_0 and p_1. The origin o of the coordinates system corresponds to the projection of the lens area summits over the x-axis. Formally, o is considered as the center of mass, i.e. the barycenter[1], of points p_0 with mass $\frac{x_1}{d=(x_1-x_0)}$, and p_1 with mass $\frac{x_0}{d=(x_1-x_0)}$, and is defined as:

$$o\begin{pmatrix} 0 \\ 0 \end{pmatrix} = {}_{bar}\left\{\left(p_0\begin{pmatrix} x_0 \\ 0 \end{pmatrix}, \frac{x_1}{d}\right), \left(p_1\begin{pmatrix} x_1 \\ 0 \end{pmatrix}, \frac{x_0}{d}\right)\right\} \tag{3}$$

Applying the Al-Kashi's theorem (law of cosines) [1, 15], we obtain the following abscises for p_0 and p_1 in the local coordinates system:

$$x_0 = -\frac{d^2+{r_0}^2-{r_1}^2}{2d} \qquad x_1 = \frac{d^2-{r_0}^2+{r_1}^2}{2d} = d + x_0 \tag{4}$$

In the general case, the lens area is defined by two arcs located in the half-plans $x > 0$ and $x < 0$. Otherwise, it is a circle centered on p_0 or p_1. Hence, to represent the lens area, we use the following equations, where $f_R(y)$ denotes the right arc of the lens area and $f_L(y)$ denotes the left one:

$$f_L(y) = \begin{cases} x_1 - \sqrt{{r_1}^2 - y^2} & \text{, in the general case} \\ x_0 - \sqrt{{r_0}^2 - y^2} & \text{, if } x_1 - \sqrt{{r_1}^2 - y^2} > 0 \end{cases}$$
$$f_R(y) = \begin{cases} x_0 + \sqrt{{r_0}^2 - y^2} & \text{, in the general case} \\ x_1 + \sqrt{{r_1}^2 - y^2} & \text{, if } x_0 + \sqrt{{r_0}^2 - y^2} < 0 \end{cases} \tag{5}$$

Now, we can easily change from the global coordinates system to the local one, by a translation of vector $\begin{pmatrix} -o.X \\ -o.Y \end{pmatrix}$, that places the origin at o, followed by a rotation θ, such as:

$$\cos\theta = \frac{p_1.X - p_0.X}{d} \qquad \text{and} \qquad \sin\theta = \frac{p_1.Y - p_0.Y}{d} \tag{6}$$

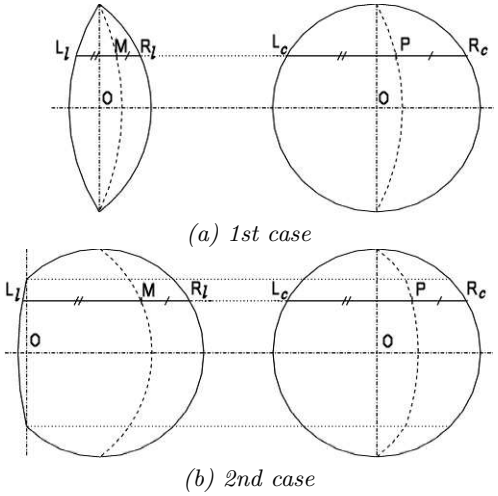
3.1.2 Lens area anamorphosis

As referred above, defining a density function over complex objects such as lens areas would not be a simple task. To cope with this problem, we use an anamorphosis, to transform the lens area into a circle of radius 1 (figure 5). The density function will be then defined over the circle.

To achieve such transformation, we define an affine bijection [15] between the lens area and the circle. This one-to-one transformation preserves collinearity (i.e., all points lying on a line initially still lie on a line after transformation), as well as the ratios of masses and distances (i.e., the barycenter of a line segment stills the barycenter of the corresponding line segment after transformation). Hence, determining the point in the circle that corresponds to a point in the lens area, is formally defined as follows:

- Consider a point ${}_M\begin{pmatrix} x \\ y \end{pmatrix}$ in the lens area (figure 5).

[1]Consider two points A_1 and A_2 defined by their cartesian coordinates (x_1, y_1) and (x_2, y_2). The mass, also referred to as the weighting coefficients, for each point is m_1 and m_2, respectively. The *barycenter* of $((A_1, m_1), (A_2, m_2))$ is a point d with cartesian coordinates (x_g, y_g) such as: $x_g = \frac{m_1 x_1 + m_2 x_2}{m_1 + m2}$ and $y_g = \frac{m_1 y_1 + m_2 y_2}{m_1 + m2}$



(a) 1st case

(b) 2nd case

Figure 5: Lens area anamorphosis

- Using the explicit equations (5), we can define the endpoints of the line segment containing M, as ${}_{L_l}\begin{pmatrix} x_L = f_L(y) \\ y \end{pmatrix}$ and ${}_{R_l}\begin{pmatrix} x_R = f_R(y) \\ y \end{pmatrix}$.
- Supposing that M is the barycenter of the line segment $\overline{L_l R_l}$, then M must verify equation (7). This means that mass coefficients of L_l and R_l are proportional to their distance from M.

$$M = bar\left\{\left(L_l, \frac{x_R - x}{x_R - x_L}\right), \left(R_l, \frac{x - x_L}{x_R - x_L}\right)\right\} \tag{7}$$

- Let us denote the maximum height of a lens area by r. Depending on the shape of the lens area, r may be equal to the length of the line segment between the summits of the lens area (figure 5(a)), or it may be equal to the radius of the smaller of the two circles that define the lens area. The latter occurs for time instants near to the instants of observations, when the lens area is a circle or when it looks like the one presented in figure 5(b). Equation 8 shows how to calculate r.

$$r = \begin{cases} \sqrt{{r_0}^2 - {x_0}^2} & \text{, in the general case} \\ r_0 & \text{, if } x_0 > 0 \\ r_1 & \text{, if } x_1 < 0 \end{cases} \tag{8}$$

- The points L_c and R_c in the boundary of the unity disk (figure 5), that correspond to the points L_l and R_l in the boundary of the lens area, are defined as follows:

$$L_c\begin{pmatrix} -\sqrt{1 - \frac{y^2}{r}} \\ \frac{y}{r} \end{pmatrix} \quad \text{and} \quad R_c\begin{pmatrix} \sqrt{1 - \frac{y^2}{r}} \\ \frac{y}{r} \end{pmatrix} \tag{9}$$

- Point P in the unity disk is the image of M, obtained by an affine one-to-one transformation of line segment $\overline{L_l R_l}$ in the lens area into $\overline{L_c R_c}$ in the unity disk:

$$P = {}_{bar}\left\{\left(L_c, \frac{x_R - x}{x_R - x_L}\right), \left(R_l, \frac{x - x_L}{x_R - x_L}\right)\right\} \tag{10}$$

- Hence, P is the image of M obtained by an anamorphosis α such as:

$$M\begin{pmatrix} x \\ y \end{pmatrix} \xrightarrow{\alpha} P\begin{pmatrix} \frac{2x - x_L - x_R}{x_R - x_L}\sqrt{1 - \left(\frac{y}{r}\right)^2} \\ \frac{y}{r} \end{pmatrix} \tag{11}$$

3.1.3 Defining the density function

The density of probability over the lens area is then defined through the density of probability over the unity disk. However, we cannot simply apply the transformation and keep the probability corresponding to each point, since the differential surface element has been changed by the anamorphosis:

$$\int_L f(\alpha(x,y))\, dx\, dy \neq \int_D f(x,y)\, dx\, dy = 1 \quad (12)$$

where, L denotes the lens area and D the unity disk.

Hence, the density of probability must be normalized to guarantee that the whole probability is equal to 1. As in computation it is not possible to deal with infinite sets, we have introduced the notion of granularity g in this model. We define g as the distance between two consecutive points (granules). Hence, the coordinates of each point become multiples of g. We define the weight of a point as its corresponding density of probability over the unity disk. Considering all the points in a lens area, the sum of their weights gives the total weight of the lens area W_{tot}.

$$W_{tot} = \sum_{\substack{m>0 | m\,g - x_0 < r_0 \\ m<0 | x_1 - m\,g < r_1}} \; \sum_{n | (m\,g - x_0)^2 + (n\,g)^2 < {r_0}^2} f(\alpha(m\,g, n\,g)) \quad (13)$$

The probability associated to each point is defined as the ratio of its weight over the total weight of the lens area.

$$P\left(M\begin{pmatrix} {}^{m_M\,g} \\ {}_{n_M\,g} \end{pmatrix}\right) = \frac{f(\alpha(m_M\,g, n_M\,g))}{W_{tot}} \quad (14)$$

3.1.4 Probability of presence of an object within a region

Considering a region Z and a lens area L, then $P(t)$ is proportional to the weight of the intersection zone $L \cap Z$. As the space has been decomposed into granules, the value of $P(t)$ may be easily calculated by summing the weight of the points within L that intersect Z:

$$P(t) = \frac{1}{W_{tot}} \sum_{(m\,g, n\,g) \in L \cap Z} f(\alpha(m\,g, n\,g)) \quad (15)$$

The computation of this probability may be performed simultaneously with the computation of the total weight of the lens area. The weight of each point in the intersection zone is added simultaneously to the weight of the lens area W_{tot} and to the weight of the intersection zone $W_{L \cap Z}$. Then, $P(t)$ is obtained as follows:

$$P(t) = \frac{W_{L \cap Z}}{W_{tot}} \quad (16)$$

3.2 Trajectory-based approach

The *Trajectory-based* approach consists in the generation of a large number of trajectories between each pair of consecutive observations recorded for the object. The probability of presence of a moving object within a given region, during a time interval Δt, is then estimated by the number of trajectories intersecting that region, over the total number of trajectories generated (17).

$$P(\Delta t) = \frac{\#_{trajectories(\Delta t) \cap region}}{\#_{trajectories\ generated}} \quad (17)$$

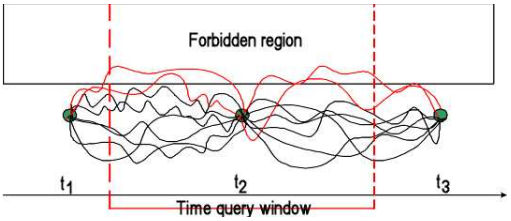


Figure 6: Trajectory-based approach

Let us now consider a motion section defined by three consecutive observations, as shown in figure 6. Let $\mathcal{T}_{0,1}$ (respectively $\mathcal{T}_{1,2}$) be the set of the $N_{0,1}$ (respectively $N_{1,2}$) trajectories generated for the first (respectively second) step of the motion section. Let $\mathcal{Z}_{0,1}$ (respectively $\mathcal{Z}_{1,2}$) be the subset of the $K_{0,1}$ (respectively $K_{1,2}$) trajectories that do not intersect the forbidden region. The obtained set of trajectories over the two steps of the motion section is then $\mathcal{T}_{0,2} = \mathcal{T}_{0,1} \times \mathcal{T}_{1,2}$, and the subset of the trajectories that do not intersect the forbidden region is $\mathcal{Z}_{0,2} = \mathcal{Z}_{0,1} \times \mathcal{Z}_{1,2}$. Thus, the probability of presence of the moving object in the given region during the query window Δt may be calculated as follows:

$$P_{0,2}(\Delta t) = \frac{\#(\mathcal{T}_{0,2} - \mathcal{Z}_{0,2}(\Delta t))}{\#\mathcal{T}_{0,2}} = \frac{\#\mathcal{T}_{0,2} - \#\mathcal{Z}_{0,2}(\Delta t)}{\#\mathcal{T}_{0,2}}$$

$$P_{0,2}(\Delta t) = 1 - \frac{\#(\mathcal{Z}_{0,1}(\Delta t) \times \mathcal{Z}_{1,2}(\Delta t))}{\#(\mathcal{T}_{0,1} \times \mathcal{T}_{1,2})}$$

$$P_{0,2}(\Delta t) = 1 - \frac{\#\mathcal{Z}_{0,1}(\Delta t) \#\mathcal{Z}_{1,2}(\Delta t)}{\#\mathcal{T}_{0,1} \#\mathcal{T}_{1,2}} = 1 - \frac{K_{0,1}}{N_{0,1}} \frac{K_{1,2}}{N_{1,2}}$$

Hence:

$$P_{0,2}(\Delta t) = 1 - \overline{P}_{0,1}(\Delta t) \overline{P}_{1,2}(\Delta t)$$

For the general case, where $0 < i < j$, we obtain:

$$P_{i,j}(\Delta t) = 1 - \prod_{k \in [i, j-1]} \overline{P}_{k,k+1}(\Delta t) \quad (18)$$

The key issue for this approach is the development of a generator for movement data. The generated movements should be random, but they must comply with some physical constraints on the movement of real world objects.

We have implemented two kinds of generators that we designate by the Brownian motion generator and the vector-oriented motion generator. Only the latter requires knowing the maximum velocity of the object.

3.2.1 Brownian motion generator

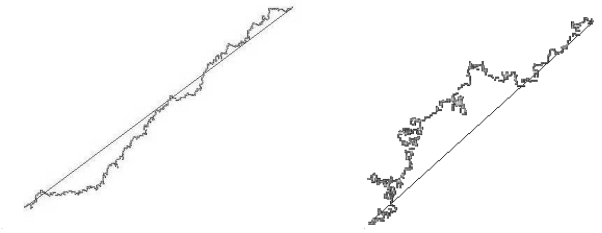
The Brownian motion is originally used to describe the movement of particles that receive a random number of impacts of random strength, from random directions, during a certain period of time. The movement of the particles between two impacts is linear and uniform. No other interaction with the particles exists. This theory has been firstly set by Robert Brown in 1827, when observing pollen particles floating on water. It is applied today to many different domains, like financial assets modeling or signal processing. In the later the Brownian motion theory is used to simulate noise associated to processed signals. A brown noise

(or "Brownian"), is a noise in which each successive sample is a small random increment or decrement above the previous sample. We follow a similar process to generate our Brownian movements.

Brownian movements

In our case, Brownian movements are generated between consecutive observations. Hence, the origin and the destination of each generated movement must coincide with the given observations. The Brownian movement generation consists in a suite of constant steps between the origin and the destination. The number of steps is fixed in advance.

We propose two generators of Brownian movements. The first one generates *one-dimensional Brownian movements*. This kind of movement does not engender backward steps. The moving object executes a suite of constant steps following the same direction: from the origin to the destination. Time is also decomposed on a regular (uniform) basis. Within each step the projection of the movement over the line defined by the origin and destination is uniform. However, the movement over the perpendicular directions is directed by a Brownian law. Figure 7.a shows a representation of the so-called one-dimensional Brownian motion.



a) One-dimensional b) Two-dimensional

Figure 7: Brownian motion

The second generator combines two one-dimensional Brownian motions to obtain a random movement, called *two-dimensional Brownian movement*. There is no longer regularity between the generated movement steps. In particular, we can observe backward movement steps. Thus, the movement is more realistic and our experiments revealed that the set of trajectories generated covers an area that is closer to the shape of a lens area, than that one covered by the one-dimensional generator. This kind of movement is illustrated in figure 7.b.

Implementation

To implement the generators described above, we define the Brownian as follows:

$$\begin{cases} B_0 = 0 \\ B_k = B_{k-1} + \sqrt{p} \,.\, N(0,1) & \forall k \in [1, n-1] \end{cases} \quad (19)$$

where n is the number of points of the Brownian, p is the fixed step and $N(0,1)$ is a generator of random values accordingly to a Gaussian law, with a zero average value and a variance equal to 1.

As the origin and destination of the movement are preset, the Brownian generator must fulfill the following criteria: $\tilde{B_0} = \tilde{B}_{N-1} = 0$. This is obtained in the following equation:

$$\tilde{B_k} = B_k - k\frac{B_{n-1}}{n-1} \qquad \forall k \in [0, n-1] \quad (20)$$

The advantage of the one-dimensional Brownian motion is that it is simple to implement. This movement is generated easily using a Brownian value to which we apply a rotation and a translation. The obtained vector is then added to an initial movement vector obtained on the basis of uniform movement following the axis origin-destination.

Let τ be the path generated between two points $A(x_a, y_a)$ and $B(x_b, y_b)$ and τ_d the path from A to B following a straight line:

$$\begin{cases} \tau_d(0) = A \\ \tau_d(n-1) = B \\ \tau_d(k) = A + k\frac{\vec{AB}}{n-1} & \forall k \in [0, n-1] \end{cases} \quad (21)$$

The one-dimensional movement is then calculated for each step using the following formula, where σ is a user defined parameter that enables amplifying the Brownian values:

$$\tau(k) = \tau_d(k) + trans(rot(\sigma \,.\, \tilde{B_k})) \qquad \forall k \in [0, n-1] \quad (22)$$

The generation of two-dimensional Brownian movement requires using four parameters $\sigma_{[2,2]}$ instead of a single σ, and two independent Brownians.

$$\tau(k) = \tau_d(k) + \begin{pmatrix} \sigma_{11} & \sigma_{12} \\ \sigma_{21} & \sigma_{22} \end{pmatrix} . \begin{pmatrix} \tilde{B_k^1} \\ \tilde{B_k^2} \end{pmatrix} \quad \forall k \in [0, n-1] \quad (23)$$

The shape of the movements generated is influenced by the four parameters $\sigma_{[2,2]}$. The parameters in the first line have an influence on the shifting of positions in the direction of the line connecting the origin and destination, and the other parameters have an influence on the shifting of positions in the perpendicular direction.

3.2.2 *Vector-oriented motion generator*

This movement generator may be suitable for applications where it is possible to estimate in advance the maximum velocity of the moving objects. It may also be applicable for situations for which the Brownian movements doesn't represent a realistic solution.

Vector-oriented movements

A vector-oriented movement is a movement composed of a suite of steps, each one is calculated using a random speed and orientation values. The main issue for this approach is the convergence of the generated movements towards the chosen destination, without interfering with the desirable random features of the generated values. To deal with this problem, we have defined a circumference with a center at the origin and radius equal to the distance between the origin and the destination. Then the movement generated must converge towards the circumference. For each step, the cape angle is generated accordingly to a distribution centered towards the closest point of the circumference. Once a point of the circumference is attained, it is enough to perform a rotation to make the last point generated coincide with the destination intended (figure 8).

Implementation

The time interval between two observations is decomposed into steps representing the smallest unit for the simulation. All steps have the same duration. New orientation and speed values are generated at each step accordingly to the following criteria:

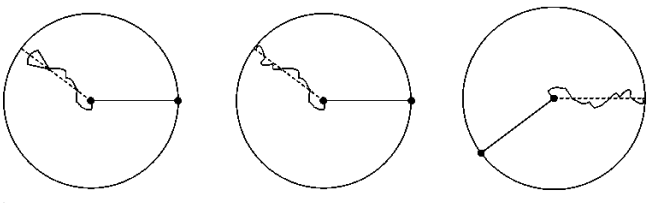


Figure 8: Generation of vector-oriented movements

- For the cape angle, the distribution is centered towards the closest point of the circumference. Considering that $\alpha_{avg}(n)$ is the mean value of the angles generated for steps 1 through $(n-1)$, the new orientation $\alpha(n)$ generated at step n is bounded by Δ_{max}, as follows:

$$\alpha(n) \in [\alpha_{avg}(n) - \Delta_{max}, \alpha_{avg}(n) + \Delta_{max}] \qquad (24)$$

- For the speed value at step n, the distribution is centered on the average speed $v_{avg}(n)$, which should be equal to the average speed required to reach the closest point of the circumference.

Our experiments revealed that the trajectories generated converge quickly towards the neighborhood of the circumference, but after that, they turned around during a certain time before reaching it.

To cope with this situation, we tried to decrease the variance of the random variables over time accordingly to a chosen law (linear, quadratic or exponential). In this way, convergence is imposed artificially but the results obtained were satisfactory for our model.

Finally, it may arise that, during the generation of a movement, we obtain a average speed value greater than the maximum speed allowed. In this case, that generated part of the movement is simply discarded and a new one is generated.

4. COMPARISON OF THE METHODS

The point-based and the vector-oriented methods are applicable to systems where it is possible to estimate the maximum velocity of the objects in advance. Otherwise, the Brownian motion method should be used.

The *point-based* method allows estimating values for $P(t)$, only when t is a time instant. Extending this method to cope with time intervals is not trivial and we were not able to find a solution allowing maintaining the desirable properties from the statistical point of view.

The *vector-oriented motion* method allows estimating values for $P(t)$, where t is a time instant or a time interval.

We have developed a tool that implements the two methods just referred. Figure 9 is a screenshot for a simulation using the vector-oriented method. It shows a static spatial region (a triangle) and a sequence of three observations (the dots in the origin and destination of the trajectories). The ellipses are the lens areas for the time intervals defined for each pair of consecutive observations. The shaded area, corresponding to the intersection of two circles with the ellipsis in the left, is the lens area for a time interval previously defined. The lines represent the trajectories generated according to the specifications presented in this paper. The window in the bottom shows that the value estimated for the probability of presence of the moving object within the triangle during the time interval considered was 16.667%.

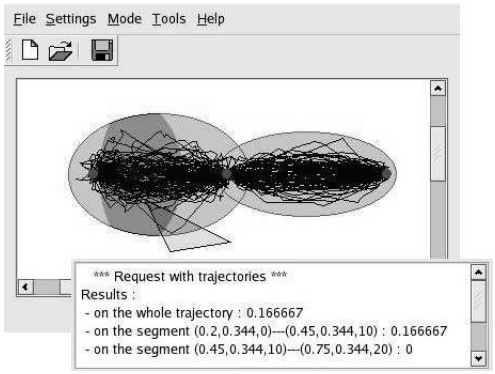


Figure 9: Vector-oriented motion tool

The *Brownian motion* method should be used for systems where the maximum velocity of the moving objects cannot be estimated accurately in advance. Under these circumstances, it is not possible to define lens areas.

The simulation depicted in figure 10 is similar to the previous one. Notice that this tool displays static polygons in a triangulated form and trajectories are not actually displayed. Instead, it is used a color scale, where the color of the pixels in the lens area are darker for smaller probabilities and lighter for higher probabilities.

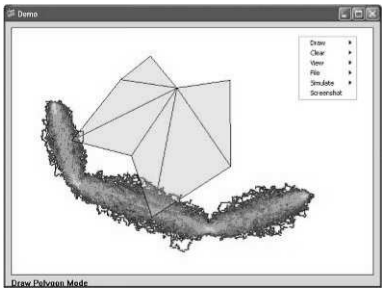


Figure 10: Brownian motion tool

These two figures also show that the density of trajectories within lens areas is not uniform. In both cases, the density of trajectories around the shortest path between two consecutive observations is higher than for locations faraway from this path.

For methods grounding on a maximum velocity value, it is also possible to guaranty that when $P(t) = 0$, the moving object has not been within the specified region during that time. Consequently, these methods may also be used to implement semantics adequate to answer questions such as "Which were the moving objects that surely have been within a specified region during a certain time" or "The list of all moving objects that could have been within a specified region", as proposed in [8].

The same does not hold for Brownian motion, as the parameters used for carrying out this method do not allow establishing an accurate bounding of lens areas.

It is also possible to implement other semantics. For instance, one may be interested in defining a probable operator and consider that probable means a probability of at least 80%, unlikely means at most 20%, or that a query yields true if the probability is at least 60%, among many others reasoning possibilities.

5. CONCLUSION

This paper deals with the computation of estimates about spatiotemporal events for moving objects systems. The focus is on the history of object's movement and it is assumed that the objects move freely in space.

We have followed two main approaches, a point-based approach and a trajectory-based approach, and proposed three methods, with different features, that may be applied to a wide range of moving objects systems. We succeeded on two main points: (1) We have developed a method based on non-uniform statistical distributions for estimating the location of moving objects at a certain time. This task was especially hard, due to the complex geometry of the lens area that bounds the possible locations of a moving object in the past. (2) We have explored an approach based on stochastic processes theory, to model the movement of an object between two locations. These locations correspond to observations stored in a database system. This approach was used to implement two methods for handling uncertainty in the location of moving objects.

At a first glance, it seemed that the point-based approach was the most obvious solution. This also was the solution that was envisaged in the literature. However, when using non-uniform distributions, the method based on this approach is considerably more difficult to implement than the others. Besides, this method is only applicable to answer probabilistic queries about the location of a moving object at a certain time instant, i.e., time intervals are not allowed. On the other hand, trajectory-based methods are able to cope with both cases and they also are simpler to implement.

The emphasis of this paper was on feasibility – identification of possible approaches and development of methods to put them into practice – and expressivity – which are the domains of application and the limitations, when applying these methods to the implementation of the different semantics proposed in spatiotemporal literature –.

We are currently working on the evaluation of the performance and the reliability of the methods proposed in this paper. The former, benchmarking for performance evaluation, would indicate which method is the most efficient from response times point of view. The latter concerns the quality of the estimations. This is an interesting issue, also raised in [4], as we are interested in comparing the reliability of the results of probabilistic queries, but in fact we do not know which are the true results.

6. ACKNOWLEDGMENTS

We would like to thank Cyril Branciard, Aurélien Coquard, Yvain Thonnart, Nghoc-Minh Vo, Cédric Lacage, Haykel Tagourti and Thomas Sirvent for their collaboration. They have worked on the development and implementation of the statistical tools presented in this paper, and their contribution was fundamental for the achievement of this work.

7. REFERENCES

[1] Theoreme d'Al-Kashi. *Wikipedia, l'encyclopedie libre, http://fr.wikipedia.org.*

[2] T. Abdessalem, C. du Mouza, J. Moreira, and P. Rigaux. *Management of Large Moving Objects Databases: Indexing, Benchmarking and Uncertainty in Movement Representation*, chapter 10, pages 225–249. In Spatial Databases: Technologies, Techniques and Trends. IDEA Group Publishing, 2005.

[3] F. Baudoin and M. Thieullen. Pinning class of the Wiener measure by a functional: related martingales and invariance properties. *Probab. Theory Related Fields*, 127(1):1–36, 2003.

[4] R. Cheng, D. V. Kalashnikov, and S. Prabhakar. Evaluating probabilistic queries over imprecise data. In *Proc. of SIGMOD*, San Diego, CA, USA, 2003.

[5] R.Cheng , D. V. Kalashnikov, and S. Prabhakar. Querying Imprecise Data in Moving Object Environments. In *IEEE Trans. Knowl. Data Eng.*, 16(9):1112–1127, 2004.

[6] Z. Ding and R. H. Güting. Uncertainty management for network constrained moving objects. In *Proc. of DEXA*, pages 411–421, Zaragoza, Spain, 2004.

[7] P.-L. Lions and A.-S. Sznitman. Stochastic differential equations with reflecting boundary conditions. *Comm. Pure Appl. Math.*, 37(4):511–537, 1984.

[8] J. Moreira, C. Ribeiro, and T. Abdessalem. Query Operations for Moving Objects Database Systems. In *Proc. of ACMGIS*, pages 108–114. ACM Press, 2000.

[9] D. Pfoser and C. Jensen. Capturing the uncertainty of moving-object representations. *Lecture Notes in Computer Science*, 1651:111–132, 1999.

[10] D. Pfoser and N. Tryfona. Capturing fuzziness and uncertainty of spatiotemporal objects. *Fifth East-European Conference on Advances in Databases and Information Systems*, pages 112–126, 2001.

[11] A. P. Sistla, O. Wolfson, S. Chamberlain, and S. Dao. Querying the Uncertain Position of Moving Objects. In *Temporal Databases: Research and Practice*, volume 1399, pages 310–337. Springer-Verlag, 1998.

[12] H. Tanaka. Stochastic differential equations with reflecting boundary condition in convex region. *Hiroshima Math. J.*, 9:163–177, 1979.

[13] G. Trajcevski. Probabilistic range queries in moving objects databases with uncertainty. In *Proc. of the 3rd ACM intern. workshop on Data engineering for wireless and mobile access*, pages 39–45. ACM Press, 2003.

[14] G. Trajcevski, O. Wolfson, F. Zhang, and S. Chamberlain. The geometry of uncertainty in moving objects databases. In *Proc. of EDBT*, volume 2287 of *Lecture Notes in Computer Science*, pages 233–250. Springer, 2002.

[15] E. W. Weisstein. *MathWorld–A Wolfram Web Resource, http://mathworld.wolfram.com.*

[16] O. Wolfson, A. P. Sistla, S. Chamberlain, and Y. Yesha. Updating and querying databases that track mobile units. *Distributed and Parallel Databases*, 7(3):257–387, 1999.

[17] Yufei Tao, Reynold Cheng, Xiaokui Xiao, Wang Kay Ngai, Ben Kao, Sunil Prabhakar. Indexing Multi-Dimensional Uncertain Data with Arbitrary Probability Density Functions. In *Proceedings of the 31st VLDB Conference*, pages 922–933, 2005.

[18] O. Wolfson, A. P. Sistla, B. Xu, J. Zhou, S. Chamberlain, Y. Yesha, and N. Rishe. Tracking moving objects using database technology in DOMINO. In *Proc. of NGITS*, volume 1649 of *Lecture Notes in Computer Science*, pages 112–119. Springer, 1999.