IEICE TRANS. INF. & SYST., VOL.E94–D, NO.12 DECEMBER 2011

PAPER

Modeling Uncertainty in Moving Objects Databases

Shayma ALKOBAISI[†a)], **Wan D. BAE**[††b)], ***Members***, ***and*** **Sada NARAYANAPPA**[†††c)], ***Nonmember***

SUMMARY The increase in the advanced location based services such as traffic coordination and management necessitates the need for advanced models tracking the positions of *Moving Objects* (MOs) like vehicles. Due to computer processing limitations, it is impossible for MOs to continuously update their locations. This results in the uncertainty nature of a MO's location between any two reported positions. Efficiently managing and quantifying the uncertainty regions of MOs are needed in order to support different types of queries and to improve query response time. This challenging problem of modeling uncertainty regions associated with MO was recently addressed by researchers and resulted in models that ranged from linear which require few properties of MOs as input to the models, to non-linear that are able to more accurately represent uncertainty regions by considering higher degree input. This paper summarizes and discusses approaches in modeling uncertainty regions associated with MOs. It further illustrates the need for appropriate approximations especially in the case of non-linear models as the uncertainty regions become rather irregularly shaped and difficult to manage. Finally, we demonstrate through several experimental sets the advantage of non-linear models over linear models when the uncertainty regions of *MOs* are approximated by two different approximations; the *Minimum Bounding Box* (MBB) and the *Tilted Minimum Bounding Box* (TMBB).

key words: *moving object, spatiotemporal databases, uncertainty region, uncertainty modeling, uncertainty approximation, minimum bonding rectangle, range queries, false hits*

1. Introduction

In recent years, there has been an increasing number of location-aware spatiotemporal applications that manage continuously changing data, such as temperature, stock markets and moving objects (MOs). This paper focuses on MOs that are specific types of continuously changing data which are the most commonly used in spatiotemporal database applications. Tracking systems, mobile services and sensor-based systems now track millions of *Global Positioning Systems* (GPS) and *Radio-Frequency Identifications* (RFIDs) equipped location-aware mobile devices that can report the states of MOs. Medical applications such as the spatiotemporal distribution of disease incidences and epidemic propagation can be modeled using space-time dependent probability functions and hence cast as MO problems. Environmental applications are also concerned with MO data. For example, sensors attached to aircrafts are used to measure oil and chemical spill pollution in the seas. Homeland security and battlefield monitoring also deal with MO data. Consequently, these applications require efficient database management systems called *Moving object databases* (MODBs), that are able to store, update and query a large number of continuously changing MOs.

Spatiotemporal range queries [11] are the basis for most queries in MO databases. The following is an example of a spatiotemporal range query: "What are the license plate numbers of the taxis that were at Denver International Airport between 10 am and 10:20 am on January 15, 2008?". To accurately answer this type of location and time-based query, it is necessary to maintain the locations of a large number of MOs over time. Although MOs (taxis in the example) can continuously move or change, computer systems cannot deal with continuously occurring changes – this would effectively require infinite computational speed and sensor resolution. Thus, each object's continuously changing properties (e.g., location and velocity) can be only discretely updated (e.g., MODB collects MOs positions every N minutes). Hence, MOs are always associated with a degree of uncertainty, especially when there is a considerable time gap between two updated values. The discrete updating interval is moderately chosen such that it is practical to handle the amount of information. All the possible properties between two updated values form the *uncertainty region* of the MO.

To more efficiently answer spatiotemporal queries, the cost of the *filtering* and *refinement* steps needs to be reduced. The filtering step in the query processing identifies a set of candidates that are later checked for the final answer of the query in the refinement step. By further reducing the size of uncertainty regions associated with MOs, the approximations of these regions, which are indexed, are also reduced. This in turn reduces the the cost of the filtering and refinement is also reduced by narrowing down the number of candidates.

We summarize the main research problems considered in this paper as follows:

- Uncertainty Region Modeling: Since MO databases store locations at discrete times, the uncertainty region must include all possible object locations between two reported times. Uncertainty region modeling deals with minimizing the uncertainty regions as-

Manuscript received August 11, 2010.
Manuscript revised April 26, 2011.

[†]The author is with Faculty of Information Technology, United Arab Emirates University, UAE.
[††]The author is with the Department of Mathematics, Statistics and Computer Science, University of Wisconsin-Stout, USA.
[†††]The author is with Jeppesen, Inc., USA.
a) E-mail: shayma.alkobaisi@uaeu.ac.ae
b) E-mail: baew@uwstout.edu
c) E-mail: sada.narayanappa@jeppesen.com

DOI: 10.1587/transinf.E94.D.2440

sociated with moving objects while covering all possible locations. In this paper, we categorize the uncertainty models into two types: linear models which utilize first degree input to calculate the uncertainty regions of MOs and non-linear models which make use of second degree input in their calculations.
- Uncertainty Region Approximation: To enable fast filtering, uncertainty regions of MOs are usually approximated with simplified bounding regions. The accuracy of the filtering step can be improved by increasing the region's approximation accuracy with respect to the actual region. MO uncertainty regions can be complicated and irregularly shaped. Hence, approximations such as *Minimum Bounding Rectangles* (MBRs) in 2D, or *Minimum Bounding Boxes* (MBBs) in 3D, are needed to enable spatial indexes to be used for query processing.

This paper is an expansion of our earlier work in [2]. In that paper we proposed a model called the *Truncated Tornado* model that further identifies and removes uncertainty region areas that cannot be reached by MOs when modeled using recently proposed uncertainty models. This resulted in more accurate representation of uncertainty regions associated with MOs.

We expand upon that work in this paper for a comprehensive study of recently proposed models that are used to manage uncertainty associated with MOs, as well as provide a comprehensive study of our proposed model in [2]. We also study how different uncertainty approximations affect the performance of the uncertainty models. The key contributions of this paper can be summarized as follows:

- We provide an intensive summary of recently proposed related work in Sect. 3.
- Section 4 provides a comprehensive framework that unifies the representation used to quantify uncertainty regions generated by recently proposed models. This allows researchers to easily understand the main features of each model as well as compare the performance of each in a unified way.
- In the same section, we also propose an efficient uncertainty model, the *Truncated Tornado*, introduced in [2] that allows more accurate representation of uncertainty associated with MOs over all previously proposed models.
- Section 5 presents *Minimum Bounding Box* (MBB) representations of the studied models which are used to index the uncertainty regions of MOs.
- We provide a more accurate approximation of the uncertainty regions called the *Tilted Minimum Bounding Box* (TMBB) in Sect. 5. The *Truncated Tornado* model combined with the *Tilted Minimum Bounding Box* approximation provide a comprehensive framework that efficiently represents, quantifies and accurately approximates the uncertainty regions of MOs.
- In order to evaluate the applicability and efficiency of the studied models as well as the proposed model, we index the MBB and TMBB approximations of the models using R* trees [4]. We show through intensive experimental evaluations in Sect. 6 that the *Truncated Tornado* model combined with TMBB achieves the expected performance in all cases.

The rest of the paper is organized as follows: Section 2 provides applications where uncertainty management is needed. Related work is discussed in Sect. 3. Section 4 discusses linear and non-linear uncertainty models including our proposed *Truncated Tornado* model and Sects. 5 provides approximations for these uncertainty models. The discussed models and their approximations are analytically and experimentally evaluated in Sect. 6. Finally, Sect. 7 concludes the paper.

2. Applications

This section presents four examples of application scenarios in which uncertainty in locations of moving objects can be handled efficiently with the proposed framework.

2.1 Traffic Management

Many big cities have started traffic control and management projects in order to provide advanced transportation systems [19]. Traffic management is the process of controlling and optimizing the transportation on roads, especially in highly populated areas. Vehicles that are equipped with GPS devices or any location aware devices such as cell phones can report their positions every few seconds to a database. The data of moving objects' positions can be managed and processed to answer queries such as:

- How many cars were in a specific range between 10:00 am and 10:15 am on July 26th 2010?
- Which taxis were in a given area at a specific time?

Since exact positions of vehicles may not be known, it is only possible to find the vehicles which were possible to be in the result set. This can be achieved by finding the vehicles that have their uncertainty region overlapping the query range.

2.2 Animal Tracking

Animal tracking is another application of MO databases. In [18], environmental investigators attached satellite transmitters to two elephants released at the same location to track their movements as a part of research to follow movements of translocated elephants feeding on plantation and causing damage worth millions of dollars. The movement of one of the elephants was very random and covered very large area and did not follow a group, where the other covered much less area and was in a group many times. Such behavior of reported locations can help researchers identify some reasons and solutions to the problem investigated. The difference in the behavior of the two elephants in the experimental

result suggests for additional elephants to be sampled in the same way.

2.3 Health Monitoring

Characterizing tumor motion is one of the spatiotemporal database applications in health [27]. In radiotherapy, the objective is to control the tumor by injecting high lethal dose into the target volume, while trying not to affect the healthy organs by sparing the nearby tissues. Therefore, it is very important to quantify the target volume that needs to be affected by the dose, especially when tumors are subject to movements such as breathing in the case of lung tumor.

In most cases, to deal with the movement of the tumor, for example tumor expansion while breathing, a larger volume is considered in treatment. However, if a more accurate quantification method of the target volume exists by characterizing tumor motion, then large volumes of "good" tissues and organs can be removed from the target area while still covering any possible uncertainty.

2.4 Environmental Control

Another application where the proposed model in this paper is expected to achieve outstanding results is protecting the environment from pollution like oil spills [16], by different means such as tracking ships that cause sea pollution by spilling oil. It is important to minimize the uncertainty regions to be able to determine which ship in which region caused pollution.

3. Related Work

3.1 Uncertainty Modeling

Spatiotemporal data changes continuously over time, however the database can only deal with discrete changes. Therefore, the properties of MOs between two reported locations are subject to *uncertainty*. Uncertainty is also due to instrument and measurement error such as errors associated with the GPS device. Hence, uncertainty is an inherent aspect in spatiotemporal databases [24], and the locations of MOs stored in the database may not always represent the real positions of MOs. To be able to answer spatiotemporal queries such as "find all taxis that are within a specific area", uncertainty must be captured in the spatiotemporal database systems.

The uncertainty of the dynamic properties of a MO can be represented as an *uncertainty region* between two known locations (reported points). The uncertainty regions includes all the possible positions an object could have been in between the reported points. Thus, an uncertainty region can be parameterized by two reported locations at an interval t_1, t_2, respectively. An *uncertainty model* is a computational approach to represent the uncertainty regions associated with MOs. The path that a MO follows between two reported locations is called a *trajectory*. Two main types of spatiotemporal queries exist based on the authors in [14]. The first is coordinate-based queries that return only properties (e.g., ID, name, etc.) of MOs, or the number of objects that satisfy the query. The second type is trajectory-based queries that require the exact information about objects' trajectories. A query example of the second type can be "what are the objects that did not leave a given area for the last ten minutes". To answer such a query, the trajectories (location segments connecting reported positions) of the MOs need to be retrieved. In this paper, we target in our proposed uncertainty models queries of the first type. An example is "what MOs intersected a given area at a specific time". More about trajectory-based query examples and indexing can be found in [15].

Many uncertainty models for moving objects have been proposed based on the underlying applications. The uncertainty model in [26] assumes that the location of a moving object at any point in time is within a certain distance d from its last reported location. Based on that assumption, the uncertainty region of the MO at any location-time instance is represented by a circle of radius d centered at the reported location. Another model proposed in [26] assumes only linear movements of objects. The object's location at any time instance is bounded by a certain interval along the movement trajectory. In [17], [24], the moving objects are assumed to move in straight movements with known velocities. However, objects can deviate from these straight paths by certain distances. Uncertainty regions of moving objects in [23], [24] are presented in 3D as cylindrical bodies, which represent all the possible positions between two reported past locations. In this paper, we refer to that model as the *Cylinder* model. The authors in [10] proposed parametric space indexing for historical trajectory data. They approximate known sequences of MO locations of a trajectory with a single continuous polynomial. Their experimental results showed that their approximation method was more accurate than traditional MBRs, and that their proposed PA-tree [10] resulted in a 20%–60% reduction in the number of I/Os compared with MBR-based indexes when performing range queries.

The authors in [13] proposed one of the most common uncertainty models, showing that when the maximum velocity of an object is known, the uncertainty region between any two reported locations can be represented as an error ellipse. Two consecutive positions are linearly interpolated to produce a complete trajectory of a moving object. The authors demonstrated how to process uncertainty range queries for trajectories using the error ellipse. This error ellipse uncertainty region model is the projection of a three-dimensional spatiotemporal uncertainty region onto the two-dimensional data space. Another popular model is found in [8]. It represents the uncertainty region as an intersection of two half cones. Each cone constrains the maximum deviation from two known locations in one movement direction. It also introduces multiple granularities to provide multiple views of a moving object. We call this model the *Cone* model.

Similar to the idea proposed in [8], the uncertainty re-

gion of continuously changing data objects between two reported positions P_1 and P_2 in [28] is defined to be the overlap between two funnels; one defined from P_1 to P_2 and the other from P_2 to P_1. This uncertainty model defines both past and future spatiotemporal uncertainties of any dimensionality.

Recently in [29], a non-linear extension of the funnel model named *Tornado* was presented in [28]. The main idea of the *Tornado* model is that many moving objects move with momentum. This higher degree model reduces the size of the uncertainty region by taking into account the temporally varying higher order derivatives, such as velocity and acceleration. It also defines the uncertainty region to be the overlap between two funnels defined by 2nd degree functions, each has a shape of a "tornado" –hence the name *Tornado* model.

3.2 Uncertainty Approximation

Since uncertainty regions can be rather irregular which makes them computationally expensive and unsuitable for indexing, approximations of these regions are needed. For example, MBRs or parametric boundary polynomials [10] that cover more detailed uncertainty regions of the underlying uncertainty model can be indexed for efficient filtering.

There is a lack of research in investigating the effect of different object approximations on the false-hit rate. In [5], the authors investigated six different types of static spatial objects approximations. Their results indicated that depending on the complexity of the objects and the type of queries, the approximations ellipse and rotated bounding box outperform the axis-parallel bounding box. It is the reduced number of false hits that yields a considerable improvement in total query time when using the proposed approximations. However, there is a need for similar research in spatiotemporal databases.

Research in the field of computational geometry has resulted in several object approximation solutions. In [6], it was proven that a minimal area rectangle circumscribing a convex polygon has at least one side flush with an edge of the polygon. This fact was utilized in [22], allowing the use of the "*rotating calipers*" algorithm to find all minimal rectangles in linear time. A solution for approximating objects in 3D using minimum volume boxes was provided in [12]. The author presented the algorithm for computing the exact arbitrarily-oriented minimum volume bounding box of a set of points in R^3. His proposed algorithm runs in $O(n^3)$. The authors in [3], proposed an efficient solution of calculating a $(1+\epsilon)$-approximation of the non axis-parallel minimum-volume bounding box of n points in R^3. The running time of their algorithm is $O(nlogn + n/\epsilon^3)$.

The authors in [1] presented MBR approximations for three uncertainty region models, namely, the *Cylinder* model [24], the *Cone* model proposed in [8] and the *Tornado* model presented in [29]. In [2], the authors proposed the *Tilted Minimum Bounding Box* approximation for the *Truncated Tornado* model.

4. Uncertainty Region Modeling

In this section we present the linear and non-linear uncertainty models. We first discuss and summarize three recently proposed uncertainty models, providing a unified representation of the models, then we propose a new efficient non-linear model called the *Truncated Tornado*. The notations of Table 1 will be used for the rest of this paper.

4.1 Linear Models

Linear uncertainty models take first degree values as input to calculate the uncertainty regions as described in this section.

4.1.1 The Cylinder Model

The *Cylinder* model (Fig. 1) is a simple uncertainty model that represents the uncertainty region as a cylindrical body [24]. Any two adjacent reported points, P_1 and P_2 of a trajectory segment, are associated with a circle that has a radius equal to an uncertainty threshold r. The value of r represents the maximum possible displacement (MD) from the reported point including the instrument and measurement error, e.

The *Cylinder* model quantifies the maximum displacement using the maximum velocity M_v of the object and the time interval between P_1 and P_2 such that $MD = M_v \cdot (t_2 - t_1)$. The *Cylinder* model has the following properties: The two cross-sections at t_1 and at t_2 are hyper-circles that are perpendicular to dimension d+1 (i.e., the time dimension) and centered at, respectively, P_1 and P_2, with their radius: $r = e + MD$ (see Fig. 1).

4.1.2 The Cone Model

The *Cone* model uses the maximum velocity M_v of the moving object to calculate the maximum displacement [8]. However, as displayed in Fig. 2, the maximum displacement, MD, is a function of time t in the *Cone* model. Each direction defines a funnel that has its top as a circle with radius equal to e centered at one of the reported points and its base as a circle that is perpendicular to the time dimension at the other reported point. The overlapping region of the two funnels generated between two adjacent reported positions defines the uncertainty region of *Cone* (see Fig. 2). The boundary of the uncertainty region is the maximum possible deviation of an object travelling between P_1 and P_2 during T. The maximum displacement at a given time t is $disp(t) = M_v \cdot t$, and the maximum displacement MD is defined as $MD = M_v \cdot (t_2 - t_1)$.

The *Cone* model defines the minimum and maximum future and past locations of the moving object at a given time t as follows:

The minimum possible future position along dimension i:

$$f^1_{min}(t) = (P_{1i} - e) - disp(t) \quad (1)$$

Table 1 Notations.

Notation	Meaning
P_{1i}	reported position of point 1 in the ith dimension
P_{2i}	reported position of point 2 in the ith dimension
t_1	time instance when P_1 was reported
t_2	time instance when P_2 was reported
t	any time instance between t_1 and t_2 inclusively
T	time interval between t_2 and t_1, $T = t_2 - t_1$
V_{1i}	velocity vector at P_1 in the ith dimension
V_{2i}	velocity vector at P_2 in the ith dimension
d	space dimensionality (time dimension is not included)
e	instrument and measurement error
M_v	maximum velocity of an object
M_a	maximum acceleration of an object
MD	maximum displacement of an object
$MBR_{Cylinder}$	MBR of *Cylinder*
MBR_{Cone}	MBR of *Cone*
$MBR_{Tornado}$	MBR of *Tornado*
$MBR_{EstTornado}$	MBR of *EstTornado*
$MBR_{Truncated\ Tornado}$	MBR of *Truncated Tornado*
low_i	lower bound of an MBR in the ith dimension
$high_i$	upper bound of an MBR in the ith dimension

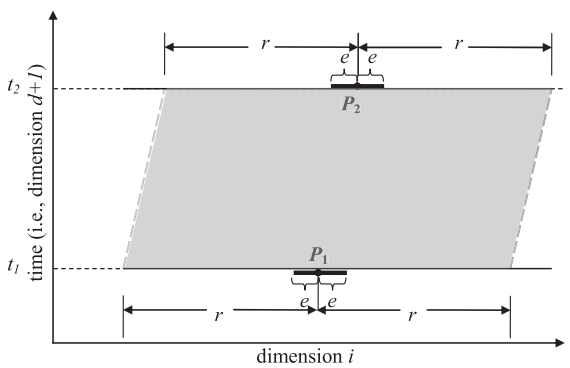


Fig. 1 Uncertainty regions in 2D (space-time) generated by the *Cylinder* model.

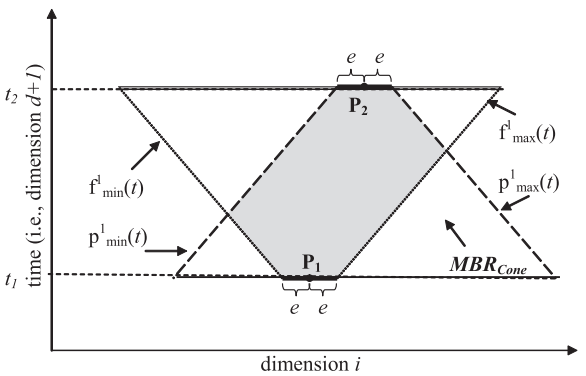


Fig. 2 Uncertainty regions in 2D (space-time) generated by the *Cone* model.

The maximum possible future position along dimension i:

$$f^1_{max}(t) = (P_{1i} + e) + disp(t) \quad (2)$$

The minimum possible past position along dimension i:

$$p^1_{min}(t) = (P_{2i} - e - MD) + disp(t) \quad (3)$$

The maximum possible past position along dimension i:

$$p^1_{max}(t) = (P_{2i} + e + MD) - disp(t) \quad (4)$$

The "f" stands for future position starting from P_1 while the "p" stands for past position starting from P_2.

4.2 Non Linear Models

Non-linear models consider second degree input values in their calculation of the uncertainty regions as discussed in the following models.

4.2.1 The Tornado Model

Both the *Cylinder* and the *Cone* models assume that a moving object can instantly reach the maximum velocity from the current velocity. However, moving objects in reality move with momentum, i.e., they need some time to change their velocities. Thus, many moving objects in a lot of cases (e.g. vehicles) move with a certain acceleration that is bounded by a maximum value. This provides the idea of a

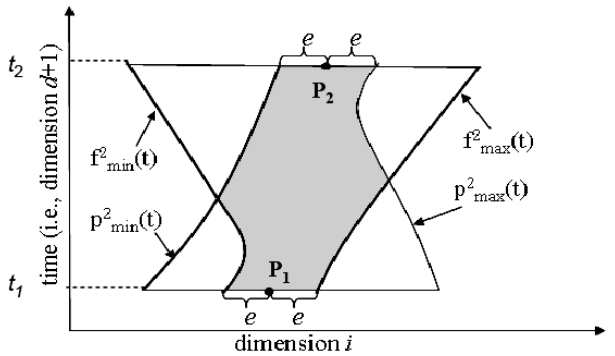


Fig. 3 Uncertainty regions in 2D (space-time) generated by the *Tornado* model.

2^{nd} degree uncertainty model called the *Tornado* model [29]. The *Tornado* model uses both the maximum velocity M_v and the maximum acceleration M_a of the moving object to calculate the maximum displacement, taking into consideration both directions: from P_1 to P_2 using V_1 as an initial velocity and from P_2 to P_1 using V_2 as an initial velocity (see Fig. 3).

Let $displ_1$ and $displ_2$ be, respectively, the first-degree and second-degree displacement functions defined as follows:

$$displ_1(V,t) = V \cdot t$$

$$displ_2(V,a,t) = \int_0^t (V + a \cdot x)dx \approx V \cdot t + (a/2) \cdot t^2$$

where V is velocity, a is acceleration and t is time. In addition, let t_{Mv} be the amount of time required to reach the maximum velocity M_v, given an initial velocity I_v and a maximum acceleration M_a, then we have: $t_{Mv} = (M_v - I_v)/M_a$, when $M_v > I_v$ (where $I_v = V_1$ for P_1 and $I_v = V_2$ for P_2).

The movement of an object is modeled as follows: an object accelerates its speed with the maximum acceleration until it reaches M_v (i.e., $displ_2$). Once it reaches M_v, it travels at Mv (i.e., $displ_1$). This is a realistic approximation of most moving objects. Then, given a location-time $< P, V, t >$ (i.e., a $d \geq 1$ dimensional location), where position P is associated with a time t and a velocity V, an object is changing its velocity towards M_v at a rate of M_a along dimension i when $D_1 = displ_1(M_v, t - t_{Mv})$ and $D_2 = displ_2(V, M_a, t_{Mv})$. The position of the object along dimension i after some time, t, from the start time t_s (i.e., t_1) is defined by the following function:

$$f\_pos(P_{1i}, V_{1i}, M_v, M_a, t, i) = \begin{cases} P_{1i} + D_2 + D_1 & \text{if } t_{Mv} < t \\ P_{1i} + D_2 & \text{otherwise} \end{cases}$$

Similarly, the position of the object before some time, t, from the start time t_s (i.e., t_2) is defined as follows:

$$p\_pos(P_{2i}, V_{2i}, M_v, M_a, t, i) = \begin{cases} P_{2i} - D_2 - D_1 \text{ if } t_{Mv} < t \\ P_{2i} - D_2 \quad \text{otherwise} \end{cases}$$

The first case (i.e., $t_{Mv} < t$) in the above functions is the case when the object reaches the maximum velocity M_v in a time period that is less than t. Hence, the position of the object is evaluated by a curve from t_s to t_{Mv} applying the maximum acceleration and by a linear function from t_{Mv} to t. However, the second case is the case when the object cannot reach M_v between t_s and t, thus the position of the object is only calculated using a non-linear function that uses the maximum acceleration M_a. When the range of velocity and acceleration of the object is $[-Mv, +Mv]$ and $[-Ma, +Ma]$, respectively, the *Tornado* model defines the future and past maximum displacements of the object as follows:

The minimum possible future position along dimension i:

$$f^2_{min}(t) = f\_pos(P_{1i}, V_{1i}, -Mv, -Ma, t, i) \quad (5)$$

The maximum possible future position along dimension i:

$$f^2_{max}(t) = f\_pos(P_{1i}, V_{1i}, +Mv, +Ma, t, i) \quad (6)$$

The minimum possible past position along dimension i:

$$p^2_{min}(t) = p\_pos(P_{2i}, V_{2i}, +Mv, +Ma, t, i) \quad (7)$$

The maximum possible past position along dimension i:

$$p^2_{max}(t) = p\_pos(P_{2i}, V_{2i}, -Mv, -Ma, t, i) \quad (8)$$

The funnel formed between $f^2_{min}(t)$ and $f^2_{max}(t)$ corresponds to all possible displacements from P_1 during T and the funnel formed between $p^2_{min}(t)$ and $p^2_{max}(t)$ corresponds to all possible displacements from P_2 during T. The area formed by the overlapping regions of the two funnels is the uncertainty region generated by the *Tornado* model (see Fig. 3).

4.2.2 The Truncated Tornado Model

In this section we present our proposed model, the *Truncated Tornado*. The main idea behind the *Truncated Tornado* model, is the following observation: objects moving with momentum cannot make extreme changes in their velocity. Their positive and negative accelerations are limited, hence they need some time to change their velocities from one direction to the opposite direction, thus, the right and left corners P_{max} and P_{min} shown in Fig. 4 are impossible to be reached by the moving object, during the time interval t_1 to t_2 while traveling from P_1 to P_2, unless we assume infinite acceleration, which is unrealistic. However, these extreme points of the *Tornado* uncertainty region imply that there are still sub-regions within the calculated uncertainty region that can be removed. Hence, removing unreachable sub-areas of the uncertainty region by calculating the furthest point (C in Fig. 5) an object can reach given its maximum acceleration, and yet being able to go back and reach the interval of the other direction would greatly reduce the volume of the uncertainty region.

We calculate the furthest point C in the *Truncated Tornado* model as follows:

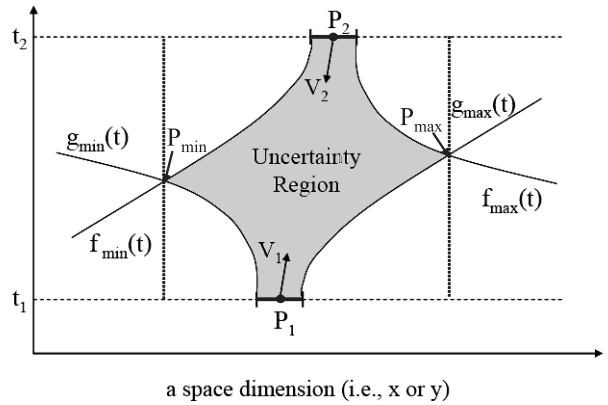


Fig. 4 Uncertainty region of the Tornado model.

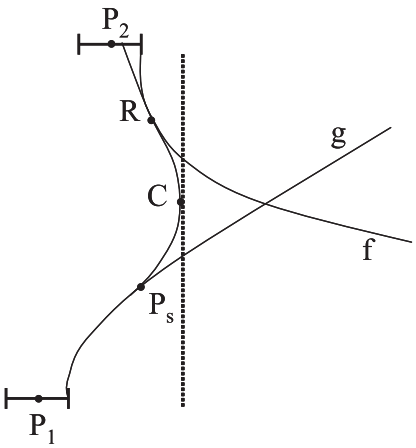


Fig. 5 Point of no safe return (case a).

Case a: Assume that the two intervals and trajectories are as shown in Fig. 5.
We say that P_s is the *point of no safe return* for g if P_s is the rightmost point on the trajectory g such that when the object (car) starts changing direction at P_s then it will touch the trajectory f (at point R), i.e., the object is within the boundary of the maximum possible deviation. Since any realizable trajectory between the two intervals must remain within the boundary defined by f and g, it is clear that the point C (which is the rightmost point on the decelerating trajectory started at P_s) can be used as a cut point for the right boundary of the MBR encompassing the uncertainty region. This boundary is indicated in the figure by the dotted line.

The question is how to calculate the points P_s and C. We show how to do this when both f, g are parabolas first (case a) and then we show how to modify the functions when P_s and R are on the linear part of g and f, respectively, i.e., when the maximal velocity is reached before P_s going from P_1 to P_2 and before R going from P_2 to P_1 (case b).

Upon turning the situation by 90 degrees counterclockwise, we see that f, g in Fig. 5 are parabolas given by $f(x) = ax^2 + b_1x + c_1$, $g(x) = ax^2 + b_2x + c_2$. (We use the same quadratic coefficient a since the maximal acceleration M_a is the same for f and g, namely $2a$.) Note that $b_1 \neq b_2$ since the parabolas f, g are not nested.

Let x_0 be the x-coordinate of the point P_s. The deceleration trajectory started at P_s is again a parabola, and it can be given by $h(x) = -ax^2 + ux + v$. We must determine u, v and x_0.

We want h to stay below f at all times, hence $-ax^2 + ux + v \leq ax^2 + b_1x + c_1$ for every x. Equivalently, $k(x) = 2ax^2 + (b_1 - u)x + (c_1 - v) \geq 0$ for every x. Since we want h to touch f, we want k to be a parabola that touches the x-axis. Equivalently, we want the discriminant $(b_1 - u)^2 - 4(2a)(c_1 - v)$ to be equal to 0. This yields

$$v = c_1 - (b_1 - u)^2/(8a) \tag{9}$$

Analogously, we want h to stay below g at all times, hence $-ax^2 + ux + v \leq ax^2 + b_2x + c_2$ for every x. Equivalently, $k(x) = 2ax^2 + (b_2 - u)x + (c_2 - v) \geq 0$ for every x. Since we want h to touch g, we want k to be a parabola that touches the x-axis. Equivalently, we want the discriminant $(b_2 - u)^2 - 4(2a)(c_2 - v)$ to be equal to 0. This yields

$$v = c_2 - (b_2 - u)^2/(8a) \tag{10}$$

The parabola h must satisfy both ((9)) and ((10)), therefore, we can set ((9)) = ((10)), eliminate v from the equation, solve for u and then substitute to find v. Solving for u we get

$$u = 4a\frac{c_2 - c_1}{b_1 - b_2} + \frac{1}{2}(b_1 + b_2) \tag{11}$$

(Here we use the observation that $b_1 \neq b_2$.)

It is now easy to find the cut point C, as this is the vertex of the parabola h. Notice that we only need to calculate u to find C since $C = \frac{u}{2a}$.

Finally, the reverse time problem (going from P_2 to P_1) is precisely the forward time problem: we are looking for a parabola that stays below and touches both f and g, hence the reverse time parabola coincides with the forward time parabola.

The same technique needs to be applied to find the cut point C' on the left boundary of the calculated MBR, i.e., the minimum extreme point of the uncertainty region. In this case, upon turning the situation by 90 degrees counterclockwise, we see that f, g are parabolas given by $f(x) = -ax^2 + b_1x + c_1$, $g(x) = -ax^2 + b_2x + c_2$ and h is a parabola given by $h(x) = ax^2 + ux + v$ and we need h to stay above f and g at all times and touches them.

Case b: The other case is when M_v is reached by the moving object going from P_1 to P_2 and going from P_2 to P_1. In this case, both P_s and R are on linear parts of g and f, respectively (see Fig. 6).

f and g are lines given by $f(x) = -M_vx + b_1$, $g(x) = M_vx + b_2$. (We use the same linear coefficient M_v since the maximal velocity is the same for f and g, namely M_v.)

Similarly, let x_0 be the x-coordinate of the point P_s. The deceleration trajectory started at P_s is again a parabola, and it can be given by $h(x) = -ax^2 + ux + v$.

Just like the first case, we want h to stay below f at all

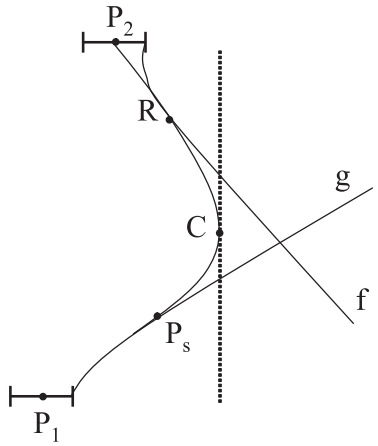


Fig. 6 Point of no safe return (case b).

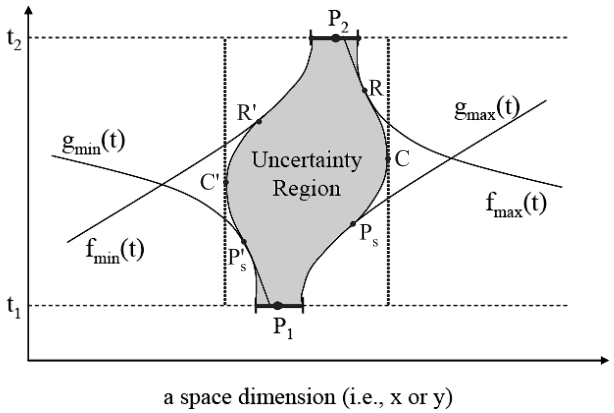


Fig. 7 Uncertainty region in 2D generated by the *Truncated Tornado* model.

times and hence, $-ax^2 + ux + v \leq -M_v x + b_1$ for every x. Equivalently, $k(x) = ax^2 - (u + M_v)x + (b_1 - v) \geq 0$ for every x. Since we want h to touch f, we want k to be a parabola that touches the x-axis. Equivalently, we want the discriminant $(u + M_v)^2 - 4a(b_1 - v)$ to be equal to 0. This yields

$$v = b_1 - \frac{(u + M_v)^2}{4a} \tag{12}$$

Analogously, we want h to stay below g at all times and hence $-ax^2 + ux + v \leq M_v x + b_2$ for every x. Equivalently, $k(x) = ax^2 + (M_v - u)x + (b_2 - v) \geq 0$ for every x. Since we want h to touch g, we want k to be a parabola that touches the x-axis. Equivalently, we want the discriminant $(M_v - u)^2 - 4a(b_2 - v)$ to be equal to 0. This yields

$$v = b_2 - \frac{(M_v - u)^2}{4a} \tag{13}$$

We can now solve for u. Setting ((12)) = ((13)), we get

$$u = \frac{a(b_1 - b_2)}{M_v} \tag{14}$$

We have now found u and hence C for the case when R and P_s are on the linear part of f and g respectively. Similar calculations can be done to find C' on the left boundary of the calculated MBR, i.e., the minimum extreme point of the uncertainty region.

Obviously, there are two other cases that need to be considered when calculating C and C', depending on the locations of P_s, R and P'_s, R', respectively. Note that the solution amounts to a quadratic equation in all cases.

The uncertainty region example shown in Fig. 7 is generated by this model when both P_s and R for the minimum and maximum calculations lie on the curved part of g and f, respectively.

Notice that both the *Tornado* and the *Truncated Tornado* models use the same maximum acceleration M_a in the positive and negative directions. For some straightforward modifications to the equations discussed above for calculating the extreme point C, one can incorporate the case where the maximum positive acceleration is different from the maximum negative acceleration. This is a very common case with many moving objects such as vehicles. For simplicity, we decided to only include the special case of having the same M_a for both the positive and negative directions.

5. Uncertainty Region Approximation

In this section, we discuss the *Minimum Bounding Rectangle* (MBR) approximation in 2D or the *Minimum Bounding Box* MBB in 3D of the uncertainty regions generated by the *Cylinder* model, the *Cone* model, the *Tornado* model and the *Truncated Tornado* model. Then, we propose a new uncertainty approximation of moving objects called the *Tilted Minimum Bounding Box* (TMBB).

5.1 Minimum Bounding Box (MBB) Approximation

To calculate the size of an MBR of each uncertainty model, we need to calculate the minimum value and maximum value of the uncertainty region generated by any two consecutive reported points in each dimension. We will use the notations *Cylinder*, *Cone*, *Tornado* and *Truncated Tornado* to refer to both, the uncertainty models and the corresponding MBR models for the rest of this paper.

5.1.1 The Cylinder Model in MBB

Recall that the *Cylinder* model quantifies the maximum displacement using the maximum velocity M_v of the object and the time interval between P_1 and P_2 such that $MD = M_v \cdot (t_2 - t_1)$. The two cross-sections at t_1 and at t_2 are hyper-circles that are perpendicular to dimension d+1 (i.e., the time dimension) and centered at, respectively, P_1 and P_2, with their radius: $r = e + MD$.

The two points that define $MBR_{Cylinder}$ in the ith dimension can be determined as follows:

$$\begin{aligned} low_i(MBR_{Cylinder}) &= min\{P_{1i}, P_{2i}\} - r \\ high_i(MBR_{Cylinder}) &= max\{P_{1i}, P_{2i}\} + r \end{aligned} \tag{15}$$

Note that the calculation of the MBR of the *Cylinder*

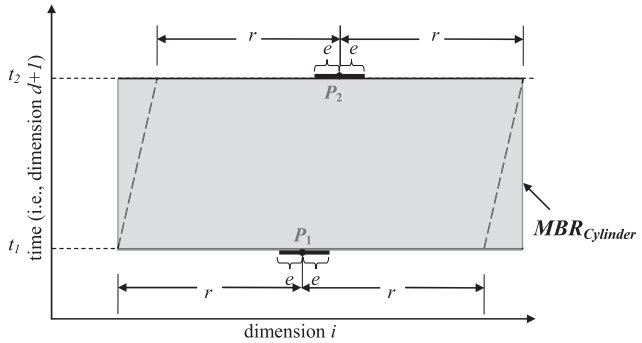


Fig. 8 MBR of the Cylinder model ($MBR_{Cylinder}$).

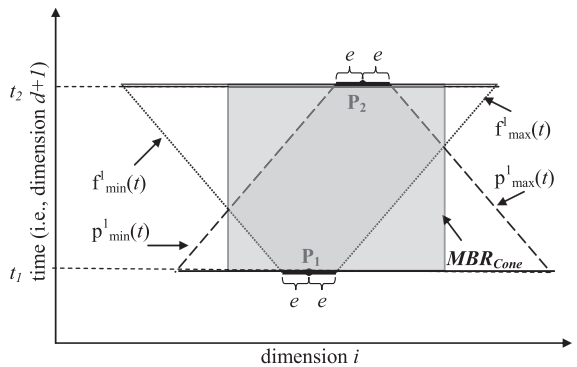


Fig. 9 MBR of the *Cone* model (MBR_{Cone}).

model (see Fig. 8) is simple and straightforward with little computational overhead.

5.1.2 The Cone Model in MBB

While the *Cylinder* model provides a simple and fast estimation of MBRs, its MBR includes large areas (volumes in 3D) that cannot be reached by the moving objects even in the worst case, i.e., the area which cannot be reached by an object moving with the maximum velocity from P_1 to P_2 during T.

As displayed in Fig. 9 and as discussed earlier, the maximum displacement, MD, is a function of time t in the *Cone* model. Moreover, the maximum displacement from both directions are considered in the calculation of MBR_{Cone}.

To calculate MBR_{Cone}, recall Eqs. (1) to (4). The cross point between $f^1_{min}(t)$ and $p^1_{min}(t)$ defines the theoretical lower bound of the MBR in the negative direction between P_1 and P_2. Similarly, the cross point between $f^1_{max}(t)$ and $p^1_{max}(t)$ defines the theoretical upper bound of the MBR in the positive direction. To find the two cross points, one can solve $f^1_{min}(t) = p^1_{min}(t)$ and $f^1_{max}(t) = p^1_{max}(t)$ for t to obtain:

$$low_i(MBR_{Cone}) = P_{1i} - e - \frac{P_{1i} - P_{2i} + MD}{2}$$
$$high_i(MBR_{Cone}) = P_{1i} + e + \frac{P_{2i} - P_{1i} + MD}{2} \quad (16)$$

Note that the calculation of MBR_{Cone} is also simple and straightforward with little computational overhead.

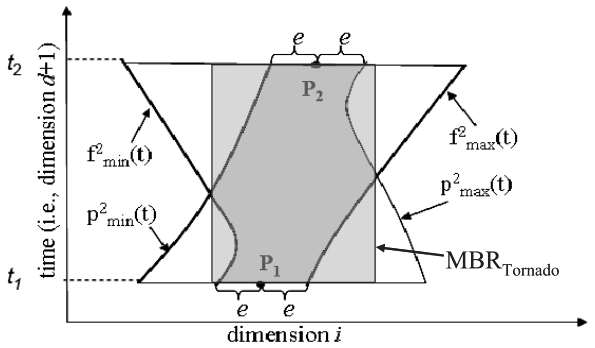


Fig. 10 MBR of the *Tornado* model ($MBR_{Tornado}$).

5.1.3 The Tornado Model in MBB

The estimation of $MBR_{Tornado}$ is basically similar to that of MBR_{Cone}, however, *Tornado* uses curves to represent the maximum displacement over time. Recall that the *Tornado* model defines the future and past maximum displacements of the object in Eqs. (5) to (8).

Figure 10 shows the output of these functions between P_1 and P_2. To calculate $MBR_{Tornado}$, one needs to determine the lower and upper bounds of the uncertainty region in every space dimension. In most cases, the lower bound can be the intersection of $f^2_{min}(t)$ and $p^2_{min}(t)$. The upper bound can be the intersection of $f^2_{max}(t)$ and $p^2_{max}(t)$. Hence, the following set of equations need to be solved for t to find the two cross points: $f^2_{min}(t) = p^2_{min}(t)$ for low_i and $f^2_{max}(t) = p^2_{max}(t)$ for $high_i$. However, in some cases, low_i can be still greater than $P_{1i} - e$ or $P_{2i} - e$ (similarly, $high_i$ can be smaller than $P_{1i} + e$ or $P_{2i} + e$). Thus, $MBR_{Tornado}$ is defined as:

$$low_i(MBR_{Tornado}) = min\{P_{1i} - e, P_{2i} - e, low_i\}$$
$$high_i(MBR_{Tornado}) = max\{P_{1i} + e, P_{2i} + e, high_i\} \quad (17)$$

To calculate the two cross points, one needs to solve a set of quadratic equations. The number of equations that need to be solved for each dimension is eight. For example, Fig. 11 shows the four possible cases of intersections between $f^2_{min}(t)$ and p^2_{min} for one dimension.

5.1.4 The Truncated Tornado Model in MBB

To calculate the MBR of the uncertainty region shown in Fig. 7, it is sufficient to choose $Max\{P_1 + e, P_2 + e, C\}$ for the maximum boundary, and to choose $Min\{P_1 - e, P_2 - e, C'\}$ for the minimum boundary, where C in the first set is calculated based on the maximum functions and C' in the second set is calculated based on the minimum functions discussed in the previous section. This should be applied for each space dimension to calculate the MBB that encloses the uncertainty volume.

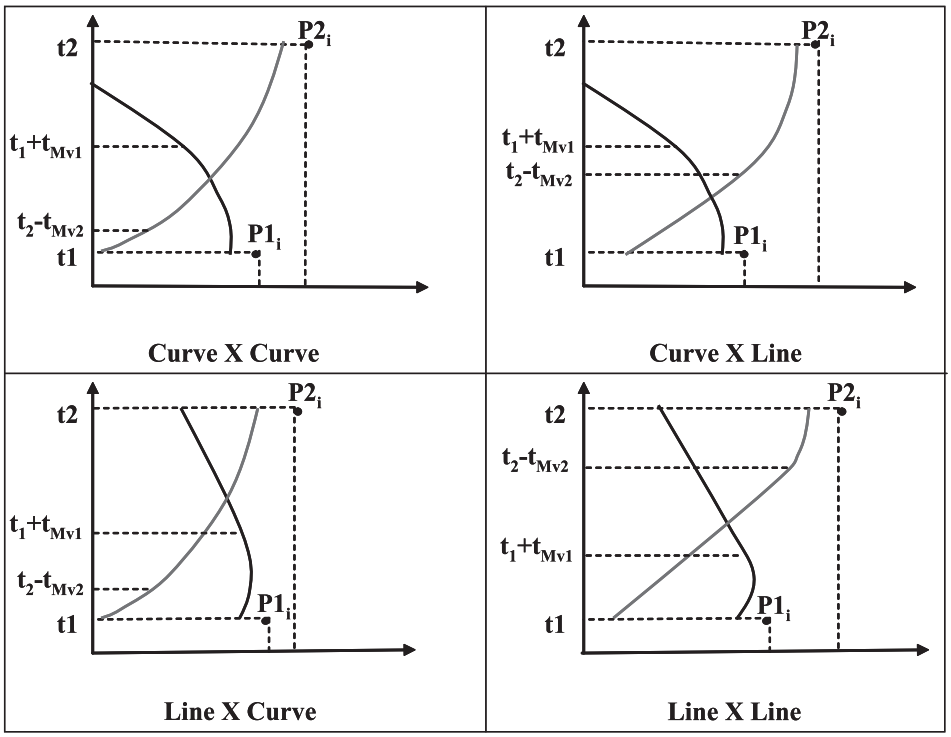


Fig. 11 Cases of intersection between $f^2_{min}(t)$ and $p^2_{min}(t)$.

5.2 Tilted Minimum Bounding Box (TMBB) Approximation

The MBB approximation greatly reduces the complexity of the geometry it represents. Another important feature of the MBB is the simple and fast execution of spatial operations such as the test for intersections. Consequently, the MBB is highly used as an approximation method. However, MBBs provide an inaccurate filter for the refinement step. The area of the MBB can differ a lot from the actual object being approximated which results in returning a large number of candidates to the query, which in turn increases the number of false hits.

The uncertainty regions generated by the *Truncated Tornado* model are rather "tilted" in shape in which traditional (axis-parallel) MBBs are most likely not close to the optimal approximations of the regions. The advantage of the *Truncated Tornado* model can be strengthened by a more accurate approximation that takes the tilted shape of the regions into account and not only the extreme points of the uncertainty region. We investigate the *Tilted Minimum Bounding Boxes* (TMBBs) as approximations of the uncertainty regions generated by the *Truncated Tornado* model. Figure 12 shows an example of the TMBR of an uncertainty region produced by the *Truncated Tornado* model vs. the axis-parallel MBR calculated in a 2D time-space dimension.

When compared with axis-parallel MBBs in 3D, TMBBs, which are minimum volume bounding boxes that relax the axis-aligned property of the MBBs, generally allow geometries to be bounded more tightly with a fewer number of boxes [7]. Attempts to evaluate TMBBs on spatial data have resulted in proving that TMBB represent more accurate approximations of spatial datasets compared to MBBs [5]. Similar research has not been done yet (to the authors knowledge) on spatiotemporal datasets that deal with uncertainty regions. For the following reasons, we decide to investigate the TMBBs as an approximation method for the uncertainty regions generated by the *Truncated Tornado* model:

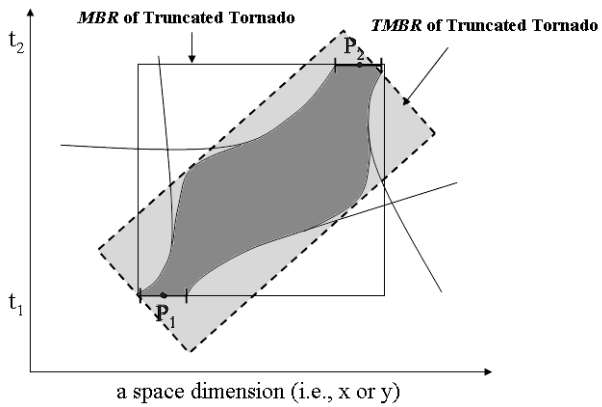


Fig. 12 MBR vs. TMBR of *Truncated Tornado*.

- Unlike spatial data, spatiotemporal data have movement directions associated with time. MOs continuously move over time, at least in one direction which provides a trend (direction) to the generated uncertainty

regions. This tilted direction in uncertainty suggests using a more general MBB enclosing the uncertainty region that accommodates the movement direction in order to more accurately approximate the uncertainty region.
- Unlike spatial data, spatiotemporal data can grow in dimension and hence, it is more important to reduce the volume of the uncertainty approximation since the advantage of reduction can be applied for every dimension. However, exact solutions to calculating the TMBB is only done for the 2D [22] and 3D cases [12].

When calculating the TMBB of the uncertainty region generated by the *Truncated Tornado* such as the one shown in Fig. 7, other points such as P_s and R might be the extreme points defining the TMBB. Hence we identify all the extreme points that need to be considered as follows:
R, C and P_s of the maximum direction (upper boundary) in the x-dimension need to be calculated and the corresponding y-values (at specific time instance when the x-values are calculated) are assigned to these points. Similarly, R, C and P_s in the y-dimension need to be calculated and the corresponding x-values (at specific time instance when the y-values are calculated) are assigned to these points. This results in six points calculated in 3D. The same calculation set needs to be done for the minimum direction (lower boundary) by calculating R', C' and P'_s in both the x and y dimensions which results in 6 other points. The other extreme points are $P_1 - e$, $P_1 + e$, $P_2 - e$ and $P_2 + e$.

Given six points in 3D calculated for the upper boundary, six points in 3D calculated for the lower boundary and finally four points in 3D (two top and two bottom), we calculate the TMBB enclosing the uncertainty region of the *Truncated Tornado* model using the approximation method of [3]. Given a set of points in 3D, their algorithm calculates an approximation of a minimal tilted MBB enclosing the set of points. The TMBBs of the uncertainty regions generated by the *Truncated Tornado* model resulted in orders of magnitude reductions in the volume compared to the axis-parallel MBBs as will be demonstrated in the next section.

6. Performance Evaluation

6.1 Analytical Evaluation

In this section we present an analytical evaluation of the *Truncated Tornado* approximated by the TMBB in order to provide an insight into the performance of the proposed uncertainty and approximation models before going into the detailed experimental evaluation. We first start by providing a general cost model for range queries that does not relate to a specific indexing structure and evaluate the approximation quality of TMBB vs. MBB. We then introduce a cost model proposed in [21] for range queries when using the R-tree as an index structure and dicuss the cost of our models.

6.1.1 Approximation Effect on Query Performance

The analysis in this section follows the approach of [7] and [25]. The authors provided a general cost model for the evaluation of range queries on objects indexed by MBBs and TMBBs. In our case, the objects are the uncertainty regions generated by the different models. The total cost of finding objects represented by general bounding volume (approximation) hierarchies that intersect a range query q can be represented by the following function: $Cost = Cost\ of\ filtering + Cost\ of\ refinement$, hence, the equation can be written as follows:

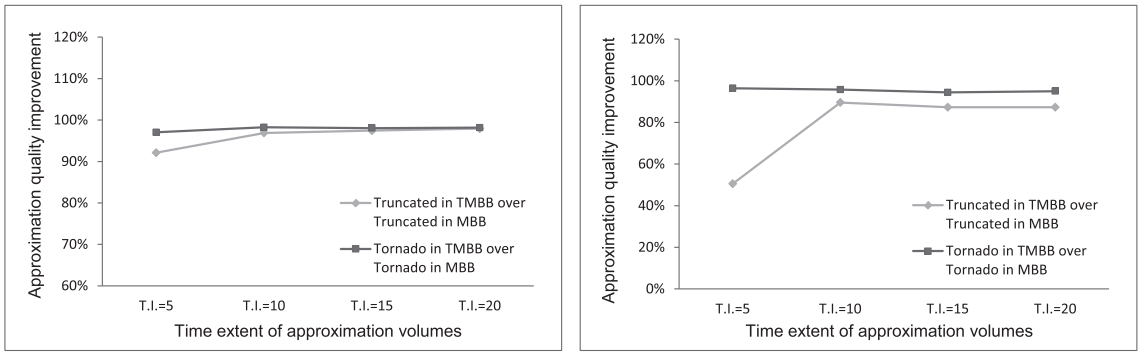
$$Cost = N_{BoundingVolume} \cdot C_{BoundingVolume} + N_{Object} \cdot C_{Object}$$

where, $N_{BoundingVolume}$: is the number of tests for bounding volumes intersecting q,
$C_{BoundingVolume}$: is the cost of testing for a bounding volume intersecting q,
N_{Object}: is the number of tests for data objects (uncertainty regions) intersecting q,
C_{Object}: is the cost of testing for a data object intersecting q

The time for refinement dominates the time for the filter step and the access frequency to the exact object by a range query depends greatly on the approximation quality [9]. The choice of the bounding volume whether it is MBB or TMBB affects $N_{BoundingVolume}$, $C_{BoundingVolume}$ and N_{Object}. The *Truncated Tornado* model's uncertainty regions when approximated by TMBB result in low values for $N_{BoundingVolume}$ by decreasing the number of intersections with q as well as low values for N_{Object} by decreasing the false-hits. However, they would result in high $C_{BoundingVolume}$ values compared to the *Cone* model and the *Tornado* model when approximated by MBB as they require higher number of operations to be evaluated [7], [25]. On the other hand, TMBBs have a much higher pruning power because of their higher approximation quality compared to MBBs [5].

Approximation quality of a bounding volume measures the "tightness" of the bounding volume with respect to the original object; the smaller the false area within the approximation, the higher the quality of the approximation. Figure 13 demonstrates the ratio of the approximation quality of the TMBB compared to the MBB using the synthetic and real datasets. Datasets generation and parameters are described in detail in Sect. 6.2.1. As TMBBs provide much higher approximation quality to uncertainty regions generated by the *Tornado* and *Truncated Tornado* models, they lead to a much smaller candidate set N_{Object} for the refinement step compared to MBBs, so the CPU cost during the refinements step will therefore be less.

Since the bottleneck in typical range queries on spatial and spatiotemporal data has been the I/O cost which is represented by the number of nodes accessed using an index structure, we present a cost model utilizing R-trees in the next section.



(a) Synthetic dataset 11 (b) Real dataset

Fig. 13 Approximation quality improvement of TMBB over MBB.

Table 2 Synthetic and real data sets and system parameters.

datasets		reported records			parameters	
		AVG Vel.	MAX Vel.	MAX Acc.	M_v	M_a
synthetic	Group 1	17.69 - 45.99	21.21 - 49.49	7.09 - 7.13	55	8
	Group 2	12.52 - 32.51	15.00 - 35.00	5.02 - 5.04	55	8
	Dataset 11	17.76	20.61	6.41	55	8
real	San Diego	11.44	36.25	6.09	38.89	6.5

6.1.2 Query Performance Using R Tree

One of the most recent cost models for range queries on uncertainty regions proposed in [21] is to estimate the number of node accesses which corresponds to the number of I/O operations when performing a range query on R tree indexed uncertainty regions generated by the proposed models.

Assuming that the entire data space is a unit workspace in 3D, the cost model is defined as follows: given an R tree R and a query q, let h_R be the height of the tree, $N_{R,L}$ be the number of nodes at level L, and $S_{R,L}$ be their average sizes. Then, the expected number of node accesses to answer q is defined as follows:

$$N_{total} = \sum_{L=1}^{h_R-1} intersect(N_{R,L}, S_{R,L}, q) \quad (18)$$

where $intesect(N, S, q) = N \cdot \prod_{k=1}^{3}(S_k + q_k)$ and S_i, q_i are the average extent of N MBBs and query q, respectively.

From the equation, the number of nodes accessed at a certain level L of the R tree depends on the average extent of the node rectangles at level L ($S_{R,L}$) which in turn depends on the average extents of the rectangles it encloses. It should be clear now that reducing the sizes of the MBBs of the uncertainty regions directly result in reducing the number of nodes accessed, and hence, reducing the total I/O cost. As will be demonstrated in our experimental evaluation, Sect. 6.2, the *Truncated Tornado* model succeeds in substantially reducing the uncertainty regions associated with each object, and therefore, reducing their MBB approximation, yielding low I/O cost compared to the other models.

6.2 Experimental Evaluation

6.2.1 Datasets and Experimental Methodology

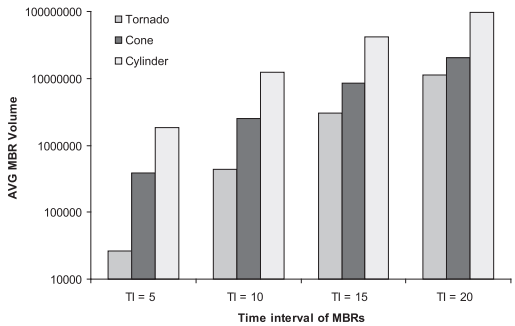
We evaluated the proposed models with both synthetic and real datasets. In all experiments, we assumed vehicles as moving objects. However, the proposed models can be applied to any moving objects.

Our synthetic datasets were generated using the "Generate_Spatio_Temporal_Data" ($GSTD$) algorithm [20] with various parameter sets such as varied velocities and different directional movements (see Table 2). On top of $GSTD$, we added a module to calculate the velocity values at each location. Each group consisted of five independent datasets (datasets $1-5$ in group 1 and datasets $6-10$ in group 2). Each dataset in group 1 was generated by 200 objects moving towards Northeast with a rather high average velocity. Each dataset in group 2 was generated by 200 objects moving towards East with a lower average velocity than the datasets in group 1. For the datasets in group 1, we varied the average velocity between 17.69 m/s and 45.99 m/s, and for the datasets in group 2, the average velocity varied between 12.52 m/s and 32.51 m/s. As a specific example, dataset 11 was generated by 200 objects moving towards Northeast but more towards East (i.e., the velocity in the x direction is greater than the velocity in the y direction) with an average velocity equal to 17.76 m/s. Each object in the eleven synthetic datasets reported its position and velocity every second for an hour.

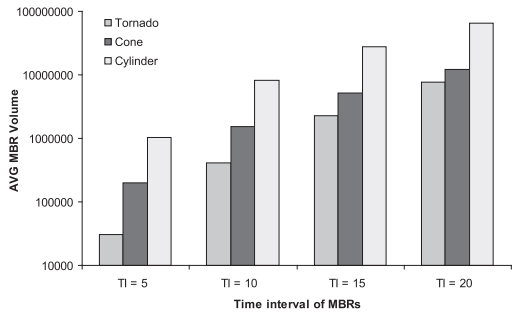
The real data set was collected using a GPS device while driving a car in the city of San Diego in California, U.S.A. The actual position and velocity were reported every

one second for a total period of 20 minutes. The average of the recorded velocities was 11.44 m/s. Table 2 shows the average velocity of the moving objects, the maximum recorded velocity and the maximum acceleration of the moving object. The last two columns show the maximum velocity and maximum acceleration values that were used in the calculation of the MBRs for each model.

Our evaluation of the models are based on two measurements. First, we quantified the volume of MBRs using each model. Second, given a range query, we measured the number of overlapping MBRs. This indicates how efficiently a range query can be evaluated using each model. In our experiments, we varied the size of the range queries. The range queries were generated randomly by choosing a random point in the universe, then appropriate x, y extents (query area) and t extent (query time) were added to that point to create a random query region (volume) in the MBR universe. The MBRs of each model for both the synthetic and real datasets were calculated using time interval TI (time interval of MBRs) equal to 5, 10, 15 and 20 seconds to investigate the impact of the update interval on the query performance using each model. We performed all our experiments on an Intel based computer running MS XP operating system, 1.66 GHz CPU, 1 GB main memory space, using Cygwin/Java tools.



(a) Synthetic dataset 11



(b) Real dataset

Fig. 14 Average MBR volume of each model.

6.2.2 Uncertainty Models in MBB Approximation

In this section, we evaluate the proposed MBR models for the *Cylinder*, *Cone*, *Tornado* and *Truncated Tornado* models using both synthetic and real data sets.

(1) A. Cylinder Vs. Cone Vs. Tornado using MBB

Figure 14 (a) shows the average volume of the MBRs generated by each model for synthetic dataset 11. The x-axis represents the time interval (TI) used to calculate the MBRs. The y-axis (logarithmic scale) represents the average volume of the MBRs. Regardless of the TI value, the *Cone* model resulted in much smaller MBRs than the *Cylinder* model. The *Tornado* model resulted in even much smaller MBRs compared to the *Cone* model. Another observation is that the larger the TI value is, the less advantage we gain from the *Tornado* model compared to the *Cone* model. When TI is large, all calculated MBRs are very large because the maximum velocity is assumed during most of the time interval (T) between any two reported points, regardless of the model.

The results using the real dataset showed the same trends compared to the synthetic data results. Figure 14 (b) shows the average volumes of the MBRs generated by each model for the San Diego dataset. The x-axis represents the time interval TI that is used to calculate the MBRs. The y-axis (logarithmic scale) represents the average volume of the MBRs in cubic meters. Regardless of the TI value, the *Cone* model resulted in much smaller MBRs than the *Cylinder* model. Also, the *Tornado* model resulted in even much smaller MBRs compared to the *Cone* model.

Next, we generated and evaluated 4000 random queries to synthetic dataset 11. We varied the query area between 0.004% and 0.05% of the area of the universe and varied the time extent of the query between 2 minutes and 8 minutes. Figure 15 shows the average number of intersecting MBRs per 1000 queries that each model resulted in while varying TI.

In all cases, the *Tornado* model resulted in an order of magnitude less number of intersections than the *Cone* model. This is because the *Tornado* model produced much smaller MBRs than the *Cone* model. Notice that the *Tornado* model has more advantage over the *Cone* model for smaller values of TI as explained in the previous result. For the same reason, the *Cone* model outperformed *Cylinder* resulting in much less number of intersections.

In Fig. 16, we generated 4000 random queries for the real dataset. We varied the query area between 0.004% and 0.05% of the area of the universe and varied the time extent of the query between 2 minutes and 8 minutes. Figure 16 shows the average number of intersections per 1000 query that each model resulted in when varying TI. All observations on the synthetic dataset results hold with the real dataset.

In addition to comparing the number of intersections each MBR model resulted in, we performed experiments on synthetic dataset 11 indexed by R* tree to compare the actual false hits. The buffer was set to 10% of the number of nodes of the tree. Figure 17 shows the number of false hits each MBR model resulted in, using 1000 randomly generated queries varying TI. The query area varied between 0.004% and 0.05% and the time extent between 2 and 8 min-

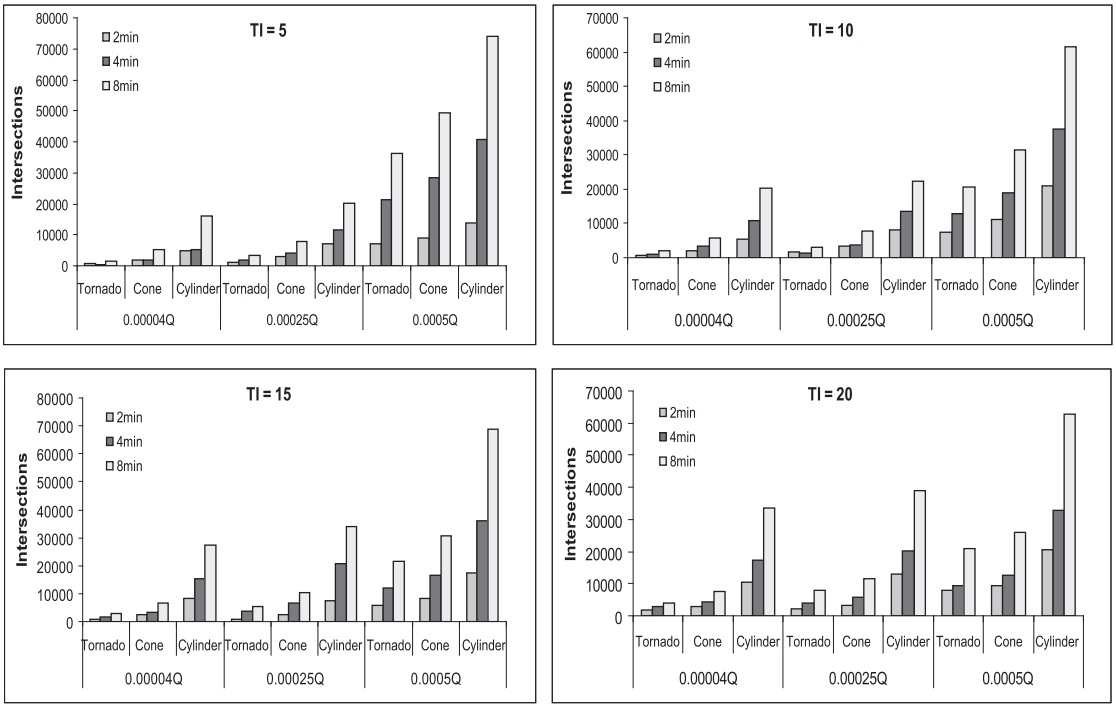


Fig. 15 Average number of intersections per 1000 query for synthetic dataset 11.

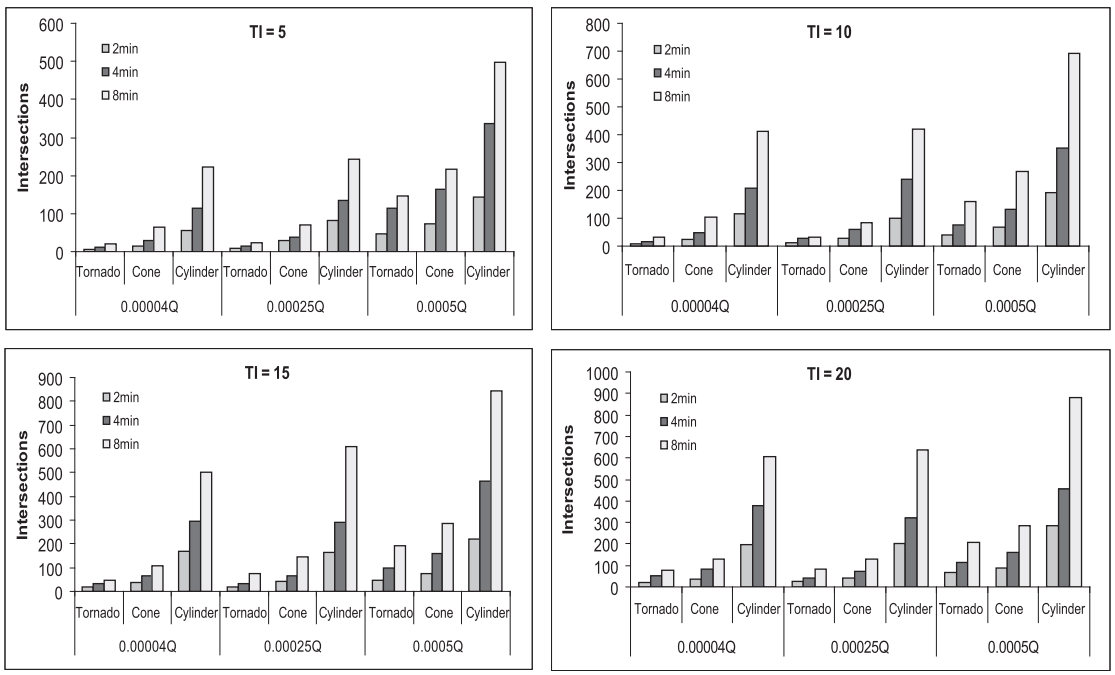


Fig. 16 Average number of intersections per 1000 query for the real dataset.

utes. In all cases, the *Tornado* model resulted in less number of false hits than the *Cone* model since the *Tornado* model produced much smaller MBRs than the *Cone* model. For the same reason, the *Cone* model outperformed the *Cylinder* model resulting in much less number of false hits.

(2) B. Cone Vs. Tornado Vs. Truncated Tornado using MBB

In this section, the performance of the *Truncated Tornado* model is illustrated in terms of the volume of MBBs approximating the uncertainty regions produced by this model. We compare the results to the *Tornado* model, the most recently

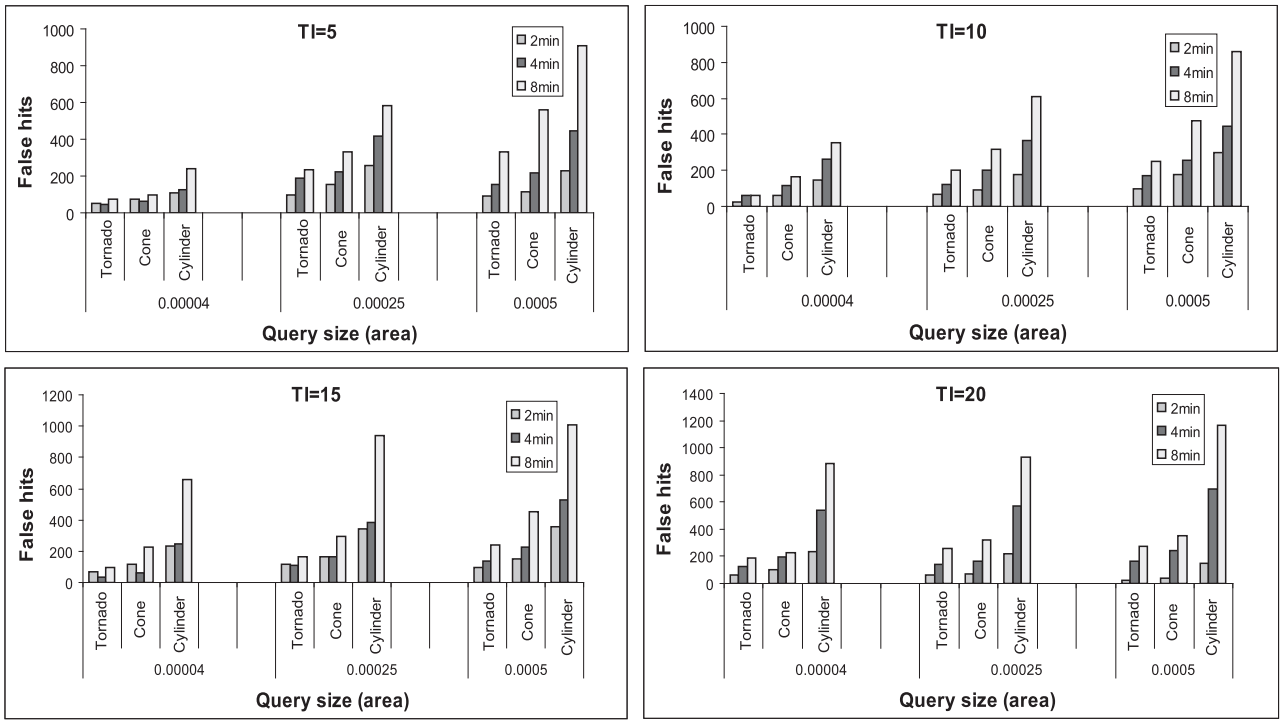
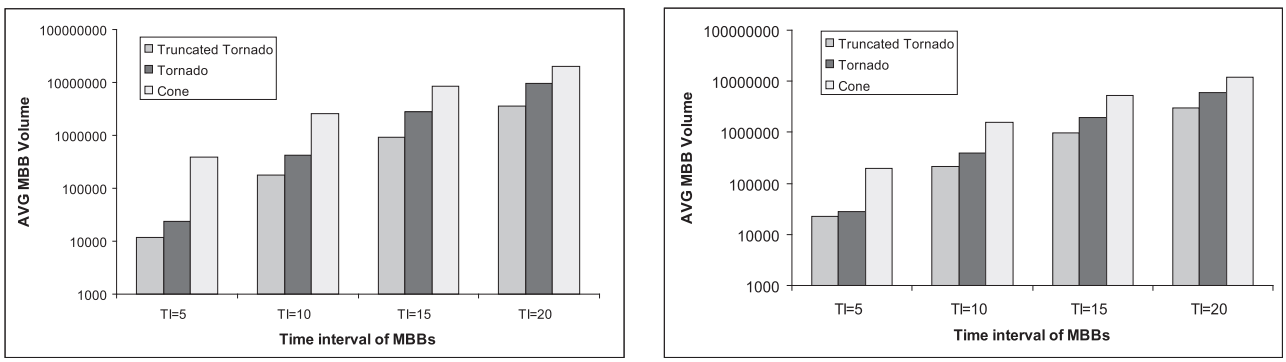


Fig. 17 Number of false hits for synthetic dataset 11.



(a) Synthetic dataset 11

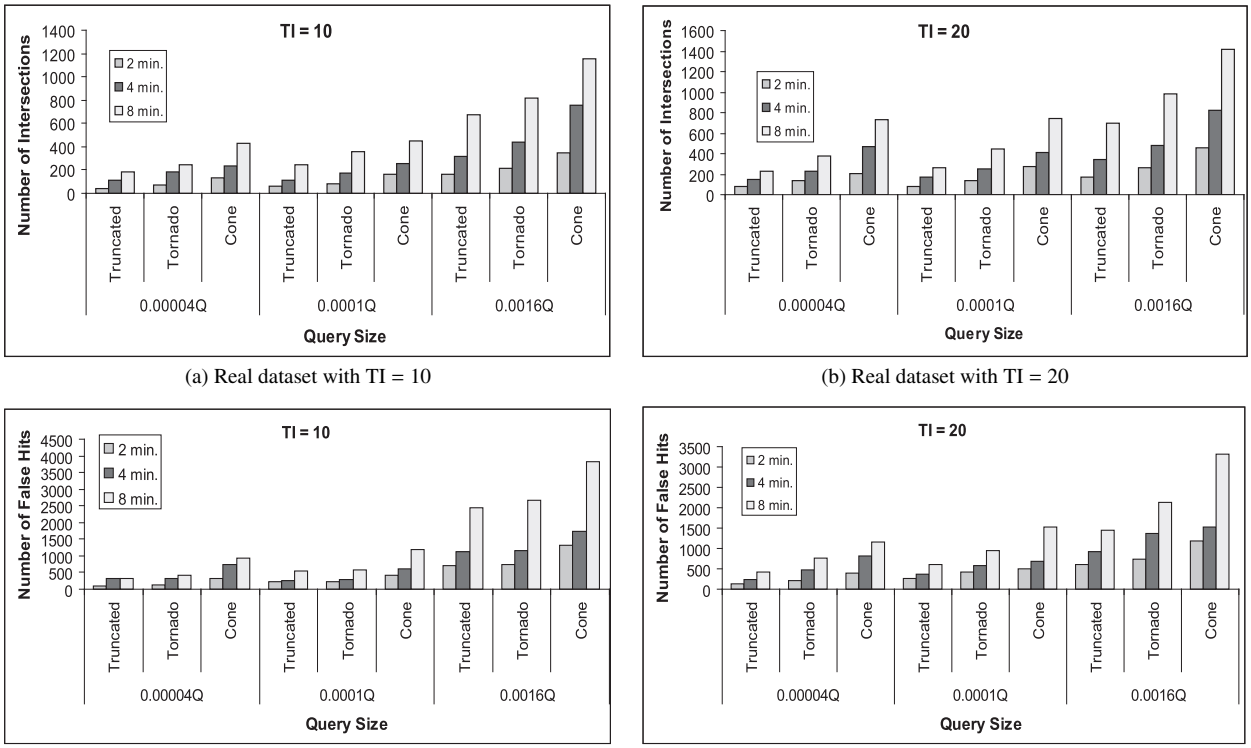
(b) Real dataset

Fig. 18 MBB volume comparison of the *Truncated Tornado*, *Tornado* and *Cone* models.

proposed uncertainty model as well as to the standard *Cone* model proposed in [8].

Figure 18 (a) and (b) show the average volume of the MBBs generated by the the *Truncated Tornado*, the *Tornado* and the *Cone* models using both the real and synthetic datasets. The average reduction in the volume of the *Truncated Tornado* MBBs for the synthetic data set over all TI values was 60% over the *Tornado* model and 90% over the *Cone* model. The average reduction in the volume was 45% over the *Tornado* MBBs and 82% over the MBBs of the *Cone* model, using the real dataset. The reduction rate in the MBB volume for both datasets was obtained based on the reduction rate in the uncertainty region volume that is produced by the *Truncated Tornado* model. The *Truncated Tornado* model produced uncertainty regions with extreme points that are included within the extreme points of the uncertainty regions generated by the *Tornado* model, this results in smaller MBB approximations of the uncertainty regions of the *Truncated Tornado* model for the same TI value used to calculate the MBBs of the *Tornado* uncertainty regions. Since most moving objects move with average velocities less than their maximum velocities, the *Tornado* and *Truncated Tornado* models outperformed the *Cone* model, which assumes that the maximum velocity is reached at all times which therefore creates larger uncertainty regions, and hence, larger MBB volumes.

Next, we generated and evaluated 5000 random range queries using the MBBs calculated by the *Truncated Tornado*, the *Tornado* and the *Cone* models for the real dataset. We varied the query area between 0.004% and 0.16% of the area of the universe and varied the time extent of the query between 2 minutes and 8 minutes.



(a) Real dataset with TI = 10

(b) Real dataset with TI = 20

(c) Synthetic dataset 11 with TI = 10

(d) Synthetic dataset 11 with TI = 20

Fig. 19 *Truncated Tornado* vs. *Tornado* vs. *Cone* with 5000 queries using MBBs.

Figure 19 (a) and (b) show the number of intersecting MBRs that each model resulted in for the real data set when TI=10 and TI=20, respectively. In both cases, the *Truncated Tornado* model resulted in much less intersections compared to the *Tornado* model. The reduction of the number of intersections is more significant when comparing the *Truncated Tornado* model to the *Cone* model. The average reduction of the *Truncated Tornado* model over the *Tornado* model in terms of number of intersections over all query sizes was 26% for TI=10 and 34% for TI=20. The reduction over the *Cone* was 51% and 60% for TI=10 and TI=20, respectively. Results when using other TI values showed similar trends.

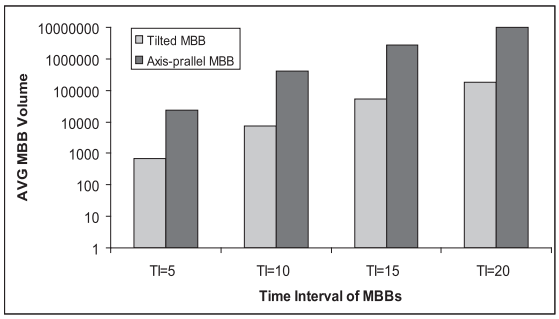
Finally, we performed the same range query experiments using R* trees to simulate practical implementation. We generated and evaluated 5000 random range queries to the MBBs calculated by the *Truncated Tornado*, the *Tornado* and the *Cone* uncertainty models for the synthetic dataset. The evaluation metric in this experimental part is the number of false-hits caused by random queries performed on the R* trees. The cache size was set to 10% of the data size and we varied the query area between 0.004% and 0.16% of the area of the universe and varied the time extent of the query between 2 minutes and 8 minutes.

Figure 19 (c) and (d) show the number of false-hits that each model resulted in when TI=10 and TI=20, respectively. In both cases, the *Truncated Tornado* model resulted in much less number of false hits compared to the the *Tornado* and to the *Cone* models. The reduction in the number of false hits was 10% over the *Tornado* model and 47% over the *Cone* model when TI=10, and 34% over the *Tornado* model and 55% over the *Cone* model when TI=20. This demonstrates that the reduction achieved in the MBB volume directly resulted in the reduction of false-hits.

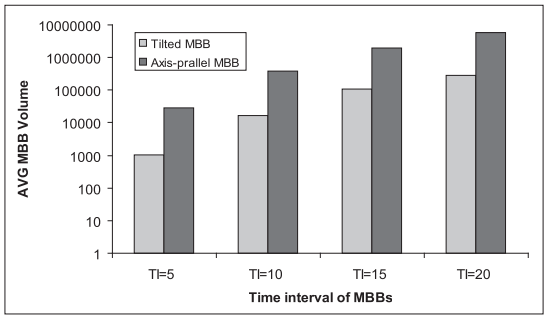
6.2.3 The Effect of TMBB Approximation

In this section, the TMBB approximation is evaluated by comparing the average volume of the TMBBs of the uncertainty regions generated by the *Tornado* and the *Truncated Tornado* models to the average volume of the axis-parallel MBBs in order to demonstrate the high accuracy of the TMBB compared to the MBB when approximating the same objects, namely, the uncertainty regions.

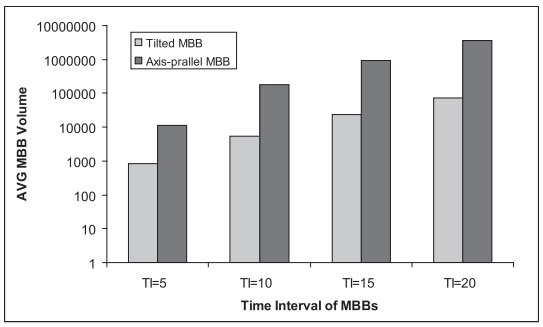
We first compared the volume of the TMBBs approximating the *Tornado* and *Truncated Tornado* uncertainty regions to that of the axis-parallel MBBs of the same regions using the real and synthetic datasets. Figure 20 (a) and (b) show the average volume of the TMBBs and the MBBs generated by the *Tornado* model for the synthetic and real datasets, respectively. Figure 20 (c) and (d) show the same for the *Truncated Tornado* model. The x-axis represents the time interval (TI) used to calculate the TMBBs and MBBs. The y-axis represents the average volume of the calculated TMBBs and MBBs (logarithmic scale). The TMBB resulted in an average reduction of 98% and 95% over the axis-parallel MBB for the synthetic and real dataset, respectively using the *Tornado* model. The reduction was 95% for the synthetic and 78% for the real dataset using the *Truncated Tornado* model. This high reduction in volume is due to the tighter approximation of the TMBB compared to the



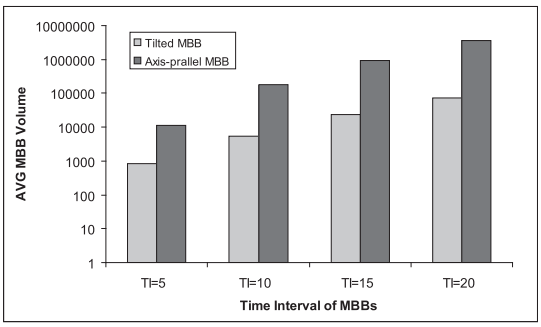
(a) *Tornado* with Synthetic dataset 11



(b) *Tornado* with Real dataset

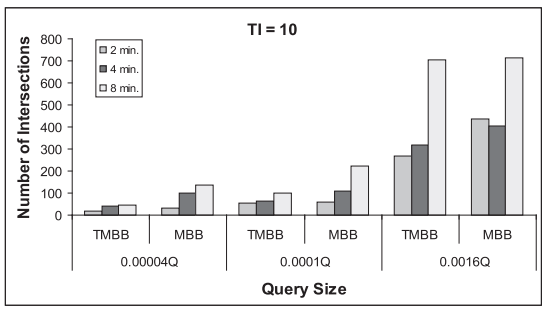


(c) *Truncated Tornado* with Synthetic dataset 11

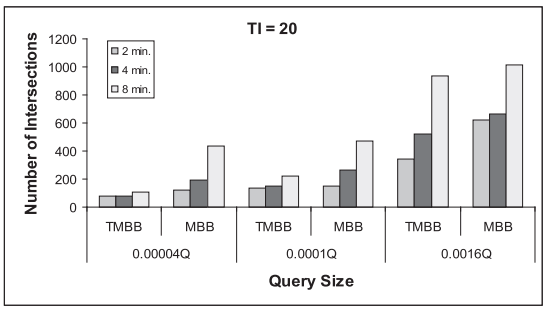


(d) *Truncated Tornado* with Real dataset

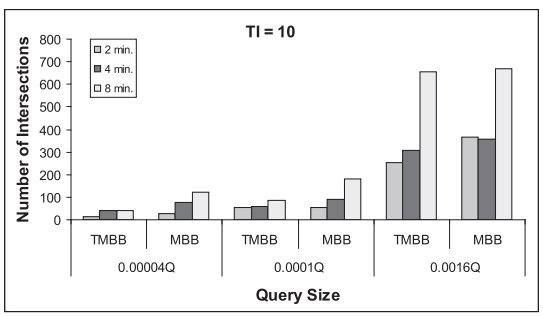
Fig. 20 TMBB volume vs. MBB volume.



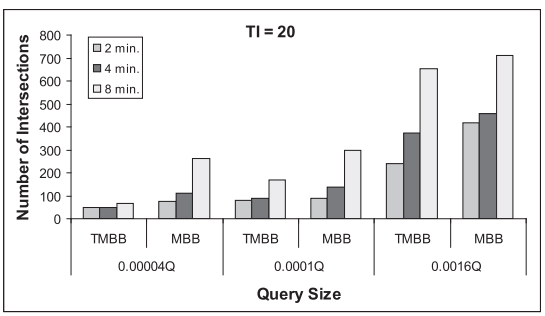
(a) *Tornado* with TI = 10



(b) *Tornado* with TI = 20



(c) *Truncated Tornado* with TI = 10



(d) *Truncated Tornado* with TI = 20

Fig. 21 TMBB vs. MBB with 5000 queries for real dataset.

MBB for this "tilted" shape of the uncertainty region. The "tilted" and "elongated" shape of the uncertainty regions generated by the *Tornado* and *Truncated Tornado* models is more accurately captured by the TMBB as apposed to the parallel-axis MBB since the TMBB can follow the direction of movement of the object and, hence, reduce the volume of the uncertainty regions' approximations.

Next, we generated and evaluated 5000 random range queries to the TMBBs and MBBs of the *Tornado* and *Truncated Tornado* models used in the previous experiment. We varied the query area between 0.004% and 0.16% of the area of the universe and varied the time extent of the query between 2 minutes and 8 minutes. Figure 21 (a) and (b) show the number of intersecting TMBBs and MBBs that the

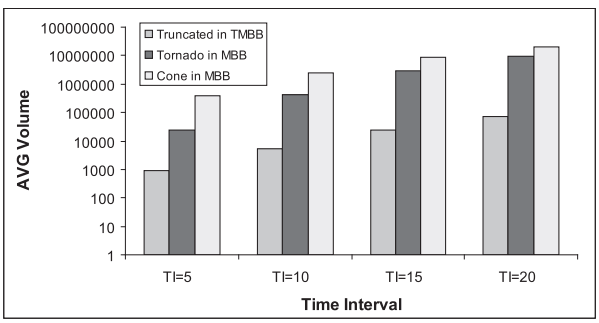
Tornado model resulted in when TI=10 and TI=20, respectively. In both cases, the TMBB resulted in much less intersections compared to the axis-parallel MBBs since TMBBs result in smaller average volumes compared to MBBs as was illustrated in earlier results. The reduction in the number of intersections of TMBB compared to MBB was 27%, 34% when TI=10 and TI=20, respectively. Figure 21 (c) and (d) show the number of intersections with the TMBBs and MBBs of the *Truncated Tornado* model. The reduction of TMBB over MBB in this case was 23% when TI=10 and 31% when TI=20. This illustrates the direct effect of the higher accuracy of the TMBB on the selectivity of the range queries.

6.2.4 Truncated Tornado + TMBB Evaluation

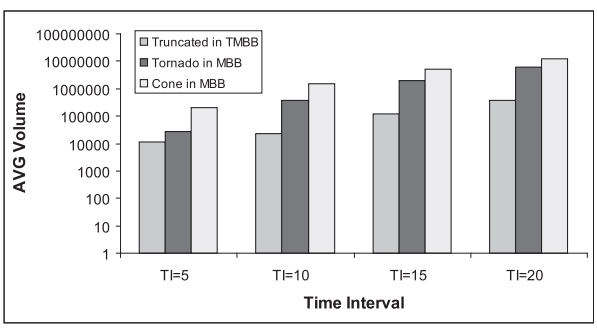
In this section, we show the effect of combining the *Truncated Tornado* and TMBB approximation method. This demonstrates the accuracy of representing uncertainty regions using the *Truncated Tornado* uncertainty model combined with a "good" approximation method, say the TMBB.

We first compared the volume of the TMBBs approximation of the *Truncated Tornado* uncertainty regions to the volume of the axis-parallel MBBs of the *Tornado* and *Cone* models using the real and synthetic datasets. Figure 22 (a) and (b) show the average volume of the TMBBs generated by the *Truncated Tornado* and the average volume of the MBBs generated by the other two models using the synthetic and real datasets, respectively. The x-axis represents the time interval (TI) used to calculate the MBBs. The y-axis represents the average volume of the TMBBs and MBBs (logarithmic scale). The *Truncated Tornado* combined with TMBB resulted in an average reduction of 93% and 97% over the axis-parallel MBB of the *Tornado* and *Cone* models, respectively, using the real dataset. The reduction when using the synthetic dataset was 99% over both the *Tornado* and *Cone* models. This reduction is due to the accuracy and efficiency in modeling the uncertainty regions generated by the *Truncated Tornado* model using the TMBB approximation.

Next, we generated and evaluated 5000 random queries to the TMBBs and MBBs calculated in the previous result for the real dataset. We varied the query area between 0.004% and 0.16% of the area of the universe and varied the time extent of the query between 2 minutes and 8 minutes. Figure 23 (a) and (b) show the number of intersecting TMBBs of the *Truncated Tornado* model and the number of intersecting MBBs of the *Tornado* and *Cone* models when TI=10 and TI=20, respectively. In both cases, the TMBBs of the *Truncated* model resulted in much less intersections compared to the *axis-parallel* MBBs of the other models since the *Truncated Tornado* results in much smaller uncertainty regions compared to the *Tornado* and *Cone* models. Also, TMBBs result in significantly smaller average volumes compared to MBBs as they more accurately approximate the uncertainty regions. The reduction in the number of intersections of the *Truncated Tornado* TMBBs was 42% over *Tornado* MBBs and 62% over *Cone* MBBs when TI=10. When TI=20, the reduction over *Tornado* MBBs was 47% and was 68% over *Cone* MBBs. Similar results were obtained using other query sizes and TI values.

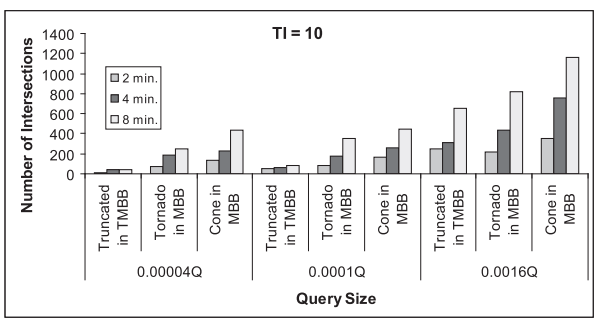


(a) Synthetic dataset 11

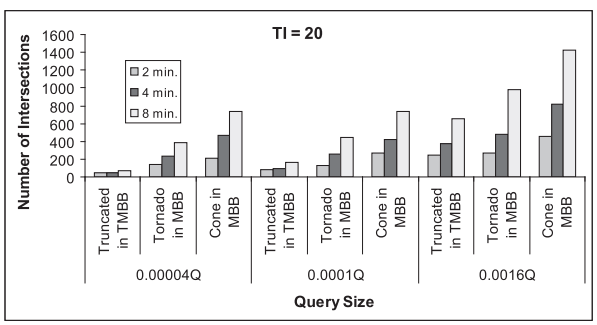


(b) Real dataset

Fig. 22 *Truncated Tornado* in TMBB vs. *Tornado* and *Cone* in MBB.



(a) TI = 10



(b) TI = 20

Fig. 23 Intersections of *Truncated Tornado* in TMBB vs. *Tornado*, *Cone* in MBB.

This result demonstrates the high efficiency obtained by the *Truncated Tornado* model in reducing the uncertainty regions, combined with the TMBB, an accurate approximation of the uncertainty regions generated by the proposed uncertainty model.

7. Conclusions

The primary focus of this paper has been to design and implement spatiotemporal models that are able to efficiently manage, quantify and query uncertainty regions associated with *Moving Objects* (MOs). In particular, this work investigated recently proposed uncertainty models, namely, the *Cylinder*, the *Cone*, the *Tornado* and the *Truncated Tornado* models.

The *Tornado* model for managing the uncertainty of continuously moving objects was particularly of interest as it utilizes higher-degree input compared to the Cylinder and Cone models that only take first degree values as input. In the two-phase (filtering and refinement) query processing, the minimality (or selectivity) of the approximation determines the false-hit ratio of the filtering phase, which translates into the cost of the refinement step; the minimality of the uncertainty model determines the final false-hit ratio of the query.

A practical framework called the *Truncated Tornado* model was proposed which managed to identify and eliminate unreachable object locations from the uncertainty regions calculated by the *Tornado* model, thus significantly reducing uncertainty region size and hence, reducing false hits.

To improve the filtering step in the query process, Sect. 5 covered *Minimum Bounding Rectangle* (MBR) approximations for the uncertainty regions generated by each of the following models: the *Cylinder*, the *Cone* the *Tornado* and the *Truncated Tornado*. By doing so, we were able to use R*-trees to index the irregularly shaped uncertainty regions.

Experiments on real and synthetic datasets showed that the *Tornado* model resulted in orders of magnitude reduction in the average volume compared to the *Cylinder* and *Cone* models which, in turn, resulted in reducing the number of false hits by 69% over the *Cylinder* model and 29% over the *Cone* model on average. We then showed how to combine the *Truncated Tornado* model with an efficient uncertainty region approximation, the *Tilted Minimum Bounding Box* (TMBB), that represented the "*tilted*" shape of the uncertainty regions more accurately. The *Truncated Tornado* model combined with TMBB resulted in an average volume reduction of 96% and 98% over the axis-parallel MBBs of the *Tornado* and *Cone* models, respectively. This resulted in reducing the number of intersections with randomly generated range queries by 42% over the *Tornado* MBBs and 62% over the *Cone* MBBs.

References

[1] S. Alkobaisi, W.D. Bae, S.H. Kim, and B. Yu, "MBR models for uncertainty regions of moving objects," Proc. Int. Conf. on Database Systems for Advanced Applications, 2008.
[2] S. Alkobaisi, P. Vojtěchovský, W.D. Bae, S.H. Kim, and S.T. Leutenegger, "The truncated tornado in TMBB: A spatiotemporal uncertainty model for moving objects," Proc. Database Expert and Systems Applications, pp.33–40, 2008.
[3] G. Barequet and S. Har-Peled, "Efficiently approximating the minimum-volume bounding box of a point set in three dimensions," J. Algorithms, vol.38, no.1, pp.91–109, 2001.
[4] N. Beckmann, H.-P. Kriegel, R. Schneider, and B. Seeger, "The R*-tree: An efficient and robust access method for points and rectangles," Proc. ACM SIGMOD, pp.322–331, 1990.
[5] T. Brinkoff, H.-P. Kriegel, and R. Schneider, "Comparison of approximations of complex objects used for approximation-based query processing in spatial database systems," Proc. Int. Conf. on Data Engineering, pp.40–49, 1993.
[6] H. Freeman and R. Shapira, "Determining the minimum-area encasing rectangle for an arbitrary closed curve," Commun. ACM, vol.18, no.7, pp.409–413, 1975.
[7] S. Gottschalk, M.C. Lin, and D. Manocha, "OBB-tree: A hierarchical structure for rapid interference detection," Proc. ACM SIGGRAPH, pp.171–180, 1996.
[8] K. Hornsby and M.J. Egenhofer, "Modeling moving objects over multiple granularities," Annals of Mathematics and Artificial Intelligence, vol.36, no.1-2, pp.177–194, 2002.
[9] H.-P. Kriegel, H. Horn, and M. Schiwietz, "The performance of object decomposition techniques for spatial query processing," Proc. Second International Symposium on Advances in Spatial Databases, pp.257–276, 1991.
[10] J. Ni and C.V. Ravishankar, "PA-tree: A parametric indexing scheme for spatio-temporal trajectories," Proc. Int. Symposium on Spatial and Temporal Databases, pp.254–272, 2005.
[11] J. Ni and C.V. Ravishankar, "Indexing spatio-temporal trajectories with efficient polynomial approximations," IEEE Trans. Knowl. Data Eng., vol.19, no.5, pp.663–678, 2007.
[12] J. O'Rourke, "Finding minimal enclosing boxes," Int. J. Parallel Program., vol.14, no.3, pp.183–199, 1985.
[13] D. Pfoser and C.S. Jensen, "Capturing the uncertainty of moving-objects representations," Proc. Int. Symposium on Advances in Spatial Databases, pp.111–132, 1999.
[14] D. Pfoser, C.S. Jensen, and Y. Theodoridis, "Novel approaches to the indexing of moving object trajectories," Proc. VLDB, pp.395–406, 2000.
[15] S. Rasetic, J. Sander, J. Elding, and M.A. Nascimento, "A trajectory splitting model for efficient spatio-temporal indexing," Proc. VLDB, pp.934–945, 2005.
[16] M. Schneider, Research Trends in Geographic Information Science, Part 2, Springer Berlin Heidelberg, 2009.
[17] P.A. Sistla, O. Wolfson, S. Chamberlain, and S. Dao, "Querying the uncertain position of moving objects," Temporal Databases: Research and Practice, vol.1399, pp.310–337, 1998.
[18] M. Stüwe, J.B. Abdul, M.N. Burhanuddin, and C.M. Wemmer, "Tracking the movements of translocated elephants in malaysia using satellite telemetry," Oryx, vol.32, no.1, pp.68–74, 1998.
[19] California Dept. of Transportation, New Technology and Research Program, Advanced transportation systems program plan, 1995.
[20] Y. Theodoridis, J.R.O. Silva, and M.A. Nascimento, "On the generation of spatiotemporal datasets," Proc. Int. Symposium on Advances in Spatial Databases, pp.147–164, 1999.
[21] Y. Theodoridis, E. Stefanakis, and T. Sellis, "Efficient cost models for spatial queries using R-trees," IEEE Trans. Knowl. Data Eng., vol.12, no.1, pp.19–32, 2000.
[22] G.T. Toussaint, "Solving geometric problems with the rotating

calipers," Proc. IEEE MELECON, pp.A10.02/1–4, 1983.
[23] G. Trajcevski, O. Wolfson, K. Hinrichs, and S. Chamberlain, "Managing uncertainty in moving objects databases," ACM Trans. Databases Syst., vol.29, no.3, pp.463–507, 2004.
[24] G. Trajcevski, O. Wolfson, F. Zhang, and S. Chamberlain, "The geometry of uncertainty in moving object databases," Proc. Int. Conf. on Extending Database Technology, LNCS 2287, pp.233–250, 2002.
[25] G. Van Den Bergen, "Efficient collision detection of complex deformable models using AABB trees," journal of graphics, gpu, and game tools, vol.2, no.4, pp.1–14, 1997.
[26] O. Wolfson, P.A. Sistla, S. Chamberlain, and Y. Yesha, "Updating and querying databases that track mobile units," Distributed and Parallel Databases, vol.7, no.3, pp.257–387, 1999.
[27] H. Wu, E. Rietzel, G. Chen, D. Kaeli, and B. Salzberg, "Characterizing tumor motion using 4-D computed tomography," CenSSIS Research and Industrial Collaboration Conference (RICC), 2003.
[28] B. Yu, "A spatiotemporal uncertainty model of degree 1.5 for continuously changing data objects," Proc. ACM Int. Symposium on Applied Computing, Mobile Computing and Applications, pp.1150–1155, 2006.
[29] B. Yu, S.H. Kim, S. Alkobaisi, W.D. Bae, and T. Bailey, "The Tornado model: Uncertainty model for continuously changing data," Proc. Int. Conf. on Database Systems for Advanced Applications, LNCS 4443, pp.624–636, 2007.

Shayma Alkobaisi is currently an assistant professor at the College of Information Technology in the United Arab Emirates University. She received her Ph.D. in Computer Science from the University of Denver in 2008. Dr. Alkobaisi's research interests include uncertainty management in spatiotemporal databases, online query processing in spatial databases, Geographic Information Systems and computational geometry.

Wan D. Bae is currently an assistant professor in the Department of Mathematics, Statistics and Computer Science at the University of Wisconsin-Stout. She received her Ph.D. in Computer Science from the University of Denver in 2007. Dr. Bae's current research interests include online query processing, Geographic Information Systems, digital mapping, multidimensional data analysis and data mining in spatial and spatiotemporal databases.

Sada Narayanappa is currently an advanced computing technologist at Jeppesen. He received his Ph.D. in Computer Science from the University of Denver in 2006. Dr. Narayanappa's primary research interests include computational geometry, graph theory, algorithms, design and implementation of databaes.