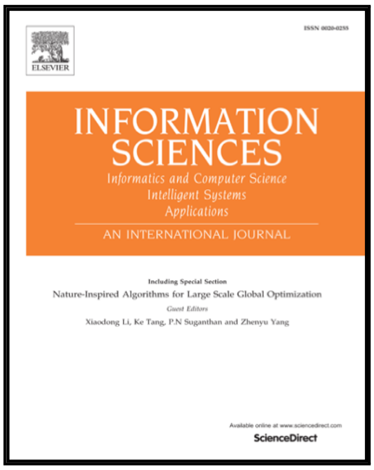
Accepted Manuscript



Handling Location Uncertainty in Probabilistic Location-Dependent Queries

Jorge Bernad, Carlos Bobed, Sergio Ilarri, Eduardo Mena

PII: S0020-0255(17)30134-2
DOI: 10.1016/j.ins.2017.01.029
Reference: INS 12714

To appear in: *Information Sciences*

Received date: 30 May 2016
Revised date: 12 December 2016
Accepted date: 9 January 2017

Please cite this article as: Jorge Bernad, Carlos Bobed, Sergio Ilarri, Eduardo Mena, Handling Location Uncertainty in Probabilistic Location-Dependent Queries, *Information Sciences* (2017), doi: 10.1016/j.ins.2017.01.029

Handling Location Uncertainty in Probabilistic Location-Dependent Queries

Jorge Bernad[a,1], Carlos Bobed[a,1,*], Sergio Ilarri[a], Eduardo Mena[a]

[a] *Dept. of Computer Science & Systems Engineering*
University of Zaragoza
50018 Zaragoza, Spain

Abstract

Location-based services have motivated intensive research in the field of mobile computing, and particularly on location-dependent queries. Existing approaches usually assume that the location data are expressed at a fine geographic precision (physical coordinates such as GPS). However, many positioning mechanisms are subject to an inherent imprecision (e.g., the cell-id mechanism used in cellular networks can only determine the cell where a certain moving object is located). Moreover, even a GPS location can be subject to an error or be obfuscated for privacy reasons. Thus, moving objects can be considered to be associated not to an exact location, but to an uncertainty area where they can be located.

In this paper, we analyze the problem introduced by the imprecision of the location data available in the data sources by modelling them using uncertainty areas. To do so, we propose to use a higher-level representation of locations which includes uncertainty, formalizing the concept of *uncertainty location granule*. This allows us to consider probabilistic location-dependent queries, among which we will focus on *probabilistic inside* (*range*) constraints. The adopted model allows us to develop a systematic and efficient approach for processing this kind of queries. An experimental evaluation shows that these probabilistic queries can be supported efficiently.

Keywords:
Probabilistic Range Queries, Location-Dependent Queries, Uncertainty Management.

1. Introduction

Location-Based Services [33, 37] (LBSs) have motivated intensive research in the field of mobile computing. These services provide value-added data by considering the locations of the mobile users and other moving objects to offer more customized information. As a basic building block of LBSs, the efficient processing of *location-dependent queries* [23] (queries whose answer depends on the location of certain moving objects) is a key issue. As a sample location-dependent query, consider a user with a smartphone that wants to locate the available taxi cabs that are near her while she is walking home on a rainy day.

Most proposals on location-dependent query processing implicitly assume GPS locations for the objects in a scenario (e.g., [8, 16, 17, 23, 30, 32, 34]). However, this is an unrealistic assumption because of several reasons:

- Positioning methods are subject to imprecision [42] due to different reasons. On the one hand, we can find that the positioning method might have an inherent precision error and provide an area where the object might be [29, 41], or directly have a location granularity coarser than required (i.e., cell-based location methods [38]). On the other hand, in the context of mobile objects, we could find that the location data is not updated as frequently as we would like, and therefore we have to assume some imprecision in the location of a moving object (e.g., providing an estimation taking into account its velocity and direction [40]).
- Locations can also be deliberately obfuscated to enhance privacy using different methods [3, 27]. For example, if the location method can only retrieve which street the user is in (e.g., due to his/her preferences), instead of his/her GPS location, querying for nearby objects must take into account that the user might be in any particular position within that street.

Note how, in all such situations and independently of the source of imprecision in the location mechanism, the different objects can be associated to an *uncertainty area* (i.e., the area where the object is probable to be located). In previous works [5, 21, 22], we introduced a location model which allowed us to represent locations at different granularities. In particular, we introduced the notion of *location-granule* as a set of physical locations with an associated semantics. As we will see in the following, we can leverage such definition to model location uncertainty dealing with imprecise locations as if they were modeled at different granularities.

*Corresponding author
Email addresses: jbernad@unizar.es (Jorge Bernad), cbobed@unizar.es (Carlos Bobed), silarri@unizar.es (Sergio Ilarri), emena@unizar.es (Eduardo Mena)

[1] The first two authors contributed equally to this work.

In this paper, we propose to model all the sources of imprecision under a unified model based on an extension of our location model, and to analyze the problem introduced by such imprecision of the available location data when it comes to processing location-based queries. Thus, we propose the concept of *uncertainty location granule* (or just *uncertainty granule*) to model situations where the uncertainty is part of the nature of the problem. Intuitively, an uncertainty granule is a place or an area where an object can be located, together with a probability density function (*pdf* from now on) to know the likelihood that the object was in any point of this area. This extension allows us to consider probabilistic location-dependent queries, among which we focus in this paper on queries with *inside constraints*, i.e., constraints that are satisfied by objects located within a certain range around a given moving object.

As an example, imagine that we want to monitor policemen that are located less than r meters from a place where a certain criminal is currently located with a probability of at least 70%; alternatively, for example, we might want to monitor all the policemen that *may* be within that radius and obtain the probability that they are actually within the radius. Let us note that we deal with *uncertainty places* (uncertain locations): 1) the place where the criminal (what we call the *reference object*) is located, and 2) the places where policemen (what we call the *target objects*) are located. We will call this type of queries *probabilistic inside queries*. This type of queries can be considered a specialization for the spatial domain in $\mathbf{R}^2$ of *fuzzy probabilistic range queries* [36] or *uncertain range queries* [44]; the concept of *probabilistic similarity join* defined in [25] is also similar as well.

While there have been many works on the processing of different types of queries in presence of uncertainty [6, 9, 13, 24, 28, 31, 45], there is a lack of an in-depth formal study of this type of queries, along with an efficient processing approach, where uncertainty is also considered for the queried position (as it can be the position of moving object) and which does not assume the availability of indexes of moving objects. Most of such works follow a similar processing schema as in [9], where the authors propose a three-step query processing approach, namely, *filtering* (i.e., prune out objects with zero probability of being in the answer set), *verification* (i.e., calculate sufficient conditions –probability bounds– for an object to be or not to be in the answer set), and *refinement* (i.e., calculate the exact probabilities); our work completely focuses on the second and third steps, not being bound to any particular approach for the first one. Thus, we can consider our approach as complementary to the others and could be used to improve them.

The main contributions of this paper are:

- We formally analyze a type of query, that we call probabilistic inside query from the point of view of our extended location model. Our results are general enough to be applied to other models in the literature.

- We provide a method to solve these queries efficiently when the underlying probability density function is the uniform distribution over a disk. Note that, in fact, the uniform distribution is the case where the least information we have about the position of the objects, and it has been used in many previous works [11, 12, 24, 28].

- We perform an extensive experimental evaluation where the efficiency of the described method for the uniform distribution can be appreciated. We show the feasibility of our approach even for mobile devices.

We would like to remark that the work presented here complements the work presented in [22], as their goals are quite different. In [22], the type of queries that we introduced and studied from a theoretical point of view involved using locations at different granularities and granule maps, while in the current paper we propose to use our extended location model to deal with imprecision, as well as a method to efficiently process the studied queries (which do not involve locations at different granularities, but just imprecise locations), and a thorough experimental evaluation. So, in this paper we provide and validate a complete and in-depth solution to process probabilistic inside queries, rather than analyze the basics of granule-based inside queries from a theorical perspective.

The structure of the rest of this paper is as follows. In Section 2, we present an overview of location granules and the extend model to incorporate uncertainty, and formally define the problem we tackle in this paper. Then, in Section 3, we present our study on probabilistic inside queries, and explain our approach to tag objects with the probabilities that they are part of the answer to the query. In Section 4, we focus on the case of uniform *pdfs*, providing a method to quickly determine whether an object belongs to the answer set for a given query. In Section 5, we present an experimental evaluation that shows that the probabilistic location-dependent queries studied in this paper can be supported efficiently. In Section 6, we present some related work. Finally, in Section 7, we draw our conclusions and present some ideas for future work.

2. Uncertainty Location Granules and Probabilistic Queries

In this section, we first provide a brief explanation of the notion of *location granule*. Then, we introduce the data model used to represent location granules and its extension to deal with uncertainty and imprecise locations. Finally, we formally define the type of range queries we deal with in this paper, to state the problem we address in latter sections. For better readability, we summarize in Table 1 the notations that will be used throughout the paper.

A *location granule* [5, 21] refers to one or more geographic areas which identify a set of GPS locations under a common name. As an example, it can be said that a given car is at a certain <x, y> location, but alternatively it can be stated, for example, that it is in the location granule *Los Angeles* (a location granule of type *city*), in the location granule *California* (a location granule of type *state*), or in the location granule *USA* (a location granule of type *country*), depending on the location granularity required (*GPS*, *city*, *state*, or *country*, respectively). This concept of *semantic location* is similar to the notion of *place* [18, 20] or *spatial granule* [4] proposed in the literature.

Table 1: Notations in the paper.

G_i	location granule
$EA(G_i)$	extended area of a location granule
$O_i =< id, UG_i, pdf_i >$	uncertainty granule with associated uncertainty area UG_i and probability density function pdf_i
$P(O \in G)$	probability that the uncertainty granule O is located inside the location granule G
$inGrProb(O, G)$	same as $P(O \in G)$
$distProb(G, O)(R)$	probability that the distance between O and G is equal to R
$P(distProb(G, O) \leq R)$ $P(distProb(O', O) \leq R)$	probability that the distance between G (O') and O is less or equal to R
$probL^{O_t}_{O_r}(R)$	same as $P(distProb(O_t, O_r) \leq R)$
$B(\mathbf{l}, r)$	disk centered in $\mathbf{l} \in \mathbf{R}^2$ with radius $r \in \mathbf{R}$
$f^r_{\mathbf{l}}$	uniform density function over $B(\mathbf{l}, r)$
$f^{R_1,R_2}_{\mathbf{l}_1,\mathbf{l}_2}$	convolution product of $f^{R_1}_{\mathbf{l}_1}$ and $\bar{f}^{R_2}_{\mathbf{l}_2}$
$A^{R_2}_{R_1}(x)$	$R_1^2 \arccos(\dfrac{R_1^2 - R_2^2 + x^2}{2R_1 x})$
$T_{R_1,R_2}(x)$	$\sqrt{4R_1^2R_2^2 - (R_1^2 + R_2^2 - x^2)^2}$
$C^{R_1,R_2}_{\mathbf{l}}$	annulus centered at $\mathbf{l}$ and radius R_1 and R_2
$C^{R_1,R_2}_{\mathbf{l}}(\alpha, \beta)$	Portion of the annulus $C^{R_1,R_2}_{\mathbf{l}}$ between an angle α and β

2.1. Data Types: Uncertainty Location Granules

The basic data model that we consider for location-dependent query processing with location granules is composed of three datatypes (see Table 2 for a summary):

- *Objects* are characterized by an internal (system-managed) identifier, a name, a GPS location (*loc.x* and *loc.y*), a class, and probably other attributes specific to their class.
- A *location granule* (or simply a *granule*) G is a tuple $(id, name, G_A)$ where *id* is an internal (system-managed) identifier, *name* is the name of the granule, and $G_A \subseteq \mathbf{R}^2$ is the area of the location granule. It provides three main kinds of operators: the *inGr* operators (short for *inGranule*) returns a boolean indicating whether a certain GPS location or location granule is within a specified granule (or within a set of granules), the *dist* operator (*distanceGranule*) computes the distance between a provided GPS location and a granule.

Up to this point, no special provisions have been made to consider location uncertainty in the data model. So, to deal with imprecise locations and probabilistic queries, the model is extended by introducing the notion of *uncertainty granule*.

Definition 1. *Uncertainty Granule (UG). An element O_t of type Uncertainty Granule (in the following, type UG) is a tuple $O_t =< id, UG_t, pdf_t >$ composed by:*

- *an internal (system-managed) identifier id.*
- *$UG_t \subseteq \mathbf{R}^2$, which is an area or a discrete set of points; we will say that UG_t is the* uncertainty area *of O_t.*
- *a function $pdf_t \in PDF$, the set of all probability density functions, whose interpretation depends on whether UG_t is an area or a set of points: if UG_t is an area, pdf_t is a probability density function from $\mathbf{R}^2$ to $\mathbf{R}$ that represents the probability of an object being in each point of this area, and therefore, $\int_{UG_t} pdf_t = 1$; if UG_t is a discrete set of points, pdf_t is a probability mass function such that $\sum_{x \in UG_t} pdf_t(x) = 1$.*

Without loss of generality, an element $x \in UG_t$ is said to be an instance of the uncertainty granule O_t and we will denote it by $x \in O_t$.

Different probability density functions could be considered to define an uncertainty granule, depending on the information available as background knowledge. As an example, a uniform distribution could be assumed, which means that the object can be with the same probability in any location within the corresponding uncertainty granule. Of course, a specific probability density function could consist of different sub-functions to estimate the probability for each of the component areas (e.g., considering a uniform distribution within some of the areas and a Gaussian distribution in others).

An uncertainty granule can represent an imprecise location. For example, due to GPS errors, a car can be considered as an uncertainty granule $O_c =< id, UG_c, pdf_c >$, where UG_c is a disk of a given radius (the error radius), and pdf_c is the uniform distribution over the disk UG_c.

The model is also extended by adding new operators that relate location granules with uncertainty granules. The *inGr* operators turns into *inGrProb*, which takes as input a granule (or a set of granules) and an uncertainty granule, and returns the probability that the imprecise location represented by the uncertainty granule is inside the location granule, that is, if G is a location granule with associated area G_A, and O_t is an uncertainty granule with associated *pdf* pdf_t, then:

$$inGrProb(G, O_t) = \int_{G_A} pdf_t$$

We also use the notation $P(O_t \in G)$ (probability that O_t is in G) for $inGrProb(G, O_t)$.

Finally, other two operators are introduced: the distance between location granules and uncertainty granules, and the distance between uncertainty granules. The former operator takes as input a location granule G and an uncertainty granule O_t and returns a probability density function:

$$distProb(G, O_t) = f : \mathbf{R} \rightarrow \mathbf{R}$$

Table 2: Basic probabilistic data model: datatypes and main operators for location granules and uncertainty granules.

Datatype	*Tuple format*	*Operators*
Object (OB)	<id, name, loc, class, otherAttr>	representObject
Location Granule (LG)	<id, name, G_A>	inGr: LG x GPS → Boolean inGr: LG x OB → Boolean inGr: $\mathcal{P}$(LG) x GPS → Boolean dist: GPS x LG → Real
Uncertainty Granule (UG)	<id, UG_t, density function>	inGrProb: LG x UG → Real inGrProb: $\mathcal{P}$(LG) x UG → Real distProb: LG x UG → PDF distProb: UG x UG → PDF

such that $distProb(G, O_t)(R)$ represents the probability that the distance between the imprecise location represented by the uncertainty granule O_t and the location granule G is equal to R, that is:

$$distProb(G, O_t)(R) = P(\{x \in O_t \mid dist(x, G) = R\})$$

Similarly, the operator $distProb(O_1, O_2)$, where O_1 and O_2 are uncertainty granules, is defined by;

$$distProb(O_1, O_2) = f : \mathbf{R} \rightarrow \mathbf{R}$$

and $distProb(O_1, O_2)(R)$ is the probability that the imprecise locations modeled by the uncertainty granules O_1 and O_2 are at distance R. Note that the distances involving uncertainty granules do not return a real number, but a function representing a probability density function.

We denote by $P(distProb(G, O) \leq R)$ (probability that G and O are not further than R) the value:

$$P(distProb(G, O) \leq R) = \int_0^R distProb(G, O)(x)dx$$

Similarly, we denote by $P(distProb(O_1, O_2) \leq R)$ the probability that the distance between O_1 and O_2 is less or equal than R.

Last but not least, we want to remark that this data model that represents imprecise locations is equivalent to the models introduced in [9, 10, 11, 13].

2.2. Probabilistic Inside Queries: Problem Statement

Now that the data model to represent objects and location granules with uncertain locations is presented, we are able to define the probabilistic location-based constraints that we address in this paper.

Definition 2. *Let $\mathcal{D} = \{O_1, \ldots, O_n\}$ be a set of uncertainty granules, O_r be an uncertainty granule, and R and p be real numbers. A* probabilistic inside query*, $q(\mathcal{D}, O_r, R, p)$, returns the uncertainty granules of the set $\mathcal{D}$ that are no further than R from the reference object O_r with a probability greater or equal to p, that is:*

$$q(\mathcal{D}, O_r, R, p) = \{O_i \in \mathcal{D} \mid P(distProb(O_i, O_r) \leq R) \geq p\}$$

For simplicity, from now on we will say that the uncertainty granules in $\mathcal{D}$ are the target objects *(more precisely, the location granules of the target objects) and the uncertainty granule O_r is the* reference object *(more precisely, the location granule of the reference object).*

In the following section, we analyze this kind of constraints and present our approach to process them in an efficient way. The main problem with these queries is to calculate the probability that an object is in the answer set. Recalling the sample query in the introduction, an interesting query could be "What is the probability that a policeman is no further than r meters from the criminal's location?". As we will see in the next sections, the solution requires to numerically solve an integral for each of the objects in the scenario, and this calculus has a direct impact on the performance.

One way to address this issue is to define some kind of spatial index enriched with additional information. For example, in [36], the authors define a type of R^*-tree to maintain the objects, so they can filter some objects using the R^*-tree, and the numerical integral is only calculated for a few objects. Similarly, in [44], the authors use a quadtree for the same purpose. However, on the one hand, in a volatile and highly-uncertain environment, where the objects are moving continuously and we cannot foresee the next movement for each object, it is difficult to maintain an index structure such an R^*-tree or a quadtree, and, on the other hand, despite leading to few false positives, every single numerical integration which can be avoided matters (i.e., it can have a significant impact on the query processing overhead). That said, our method is not incompatible with the use of a spatial index: It could be used jointly with an index to speed up the processing of the query by avoiding the computation of many numerical integrals that slow down the process. In particular, as we will show in our experiments, we adopt an approach similar to [11] to show the benefits of our approach both in the presence and the absence of an index filtering the objects which are clearly out of the answer set.

3. Processing of Probabilistic Inside Queries

In this section, we tackle the processing of probabilistic inside queries using our model. We focus on the problem of tagging each target object with the probability that the object is part of the query answer, as applying a probability threshold specified in the constraint can be considered as a filtering step once these probabilities have been computed. For simplicity of exposition, we assume uncertainty with a continuous probability density function in all the situations and for all the objects. An analogous discussion would apply for a discrete probability mass function.

As an example of the queries to be processed, consider a query such as "Which taxis (class of *target objects*) are less than R meters from Anne (*reference object*) with a probability greater than p". In this case, while raw GPS locations are considered, the locations are subject to an uncertainty; therefore, the concept of *uncertainty granule* (see Definition 1) must be used to refer to those locations.

For illustrative purposes, let us take a look at Figure 1. In the figure, UG_t is the uncertainty area associated to the target object O_t, and UG_r is the uncertainty area associated to the reference object O_r; for simplicity, and without loss of generality, UG_r is a rectangle and UG_r is a circle. The area A_t are the points of UG_t which are not further than R (the radius of the *inside* constraint) from the uncertainty area UG_r associated to the reference object O_r. Formally:

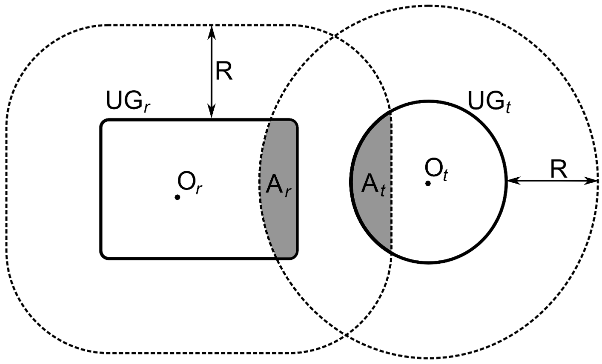


Figure 1: Areas involved in a probabilistic inside query.

$$\begin{aligned} A_t &= \{x \in UG_t \mid dist(x, UG_r) \le R\} \\ &= EA(UG_r) \cap UG_t \end{aligned}$$

where $EA(UG_r)$ is what we call *the extended area of* UG_r (the area inside the dotted line surrounding UG_r in Figure 1), i.e., the *Minkowski sum* of the area UG_r and a disk with radius R (the Minkowski sum of two sets in the Euclidean space is obtained by adding every element of one set to every element of the other [39]).

In the same way, the area A_r contains the points in UG_r that are not further than R from UG_t:

$$\begin{aligned} A_r &= \{x \in UG_r \mid dist(x, UG_t) \le R\} \\ &= EA(UG_t) \cap UG_r \end{aligned}$$

If we consider a *may* constraint (probability threshold > 0), then there is no need to compute the actual probabilities of the objects (unless this information must be computed to show it to the user). In this case, a target object will be part of the answer as long as its area A_t is not empty.

As we will see in the next subsection, calculating the exact probability that the objects O_t and O_r are no further than R involves solving numerically a double integral. This may have a great impact on the computational cost when dealing with a large number of objects, even if we assume that the pdf functions of the uncertainty granules are simple. Section 4.1 is devoted to explain how objects can be sieved to decide if an object must be or cannot be in the answer set. With this sieve, we will limit the number of objects for which we have to compute the exact probability.

3.1. Calculating the Exact Probability

This section is devoted to describe how to calculate the probability that two objects represented by the uncertainty granules O_1 and O_2 are not further than R, that is, how to calculate the value $P(distProb(O_1, O_2) \le R)$ that appears in Definition 2. We first describe how to calculate the probability for the general case, i.e., making no assumption about the uncertainty area UG_i and the density function f_i of the uncertainty granules $O_i = \langle id, UG_i, f_i \rangle$. Then, we particularize these calculations when UG_i is a disk and f_i is the uniform probability density function over the disk UG_i.

So, we will describe how to obtain the function

$$probL_{O_1}^{O_2} \colon \mathbf{R} \to [0, 1]$$

such that $probL_{O_1}^{O_2}(R) = P(distProb(O_1, O_2) \le R)$. Note that $probL_{O_1}^{O_2}$ is the cumulative distribution function of the probability density function, $distProb(O_1, O_2)$ (see Table 2 in Section 2.1).

Let O_1 and O_2 be two imprecise objects (uncertainty granules) $O_1 = \langle id, UG_1, f_1 \rangle$ and $O_2 = \langle id, UG_2, f_2 \rangle$. We will denote by X the two-dimensional random variable that models where the object O_1 is situated in the plane. It is clear that f_1 is the probability density function of the random variable X. Similarly, we define Y as a two-dimensional random variable with probability density function f_2 for O_2 (see Figure 2). Let us denote by $\bar{Y}$, the random variable, $\bar{Y}(w) = -Y(w)$ with density function $\bar{f}_2$, where $\bar{f}_2(x, y) = f_2(-x, -y)$ (that is, $\bar{f}_2$ is symmetric to f_2 with respect to the origin). This is illustrated in Figure 2.

The random variable $X + \bar{Y}$ represents all difference vectors from a possible location of O_1 to a possible location of O_2. Intuitively, the probability that the distance from O_1 to O_2 is less than R, i.e., $probL_{O_1}^{O_2}(R)$, can be calculated dividing the number of these difference vectors whose norm is less than R (all the possible locations where the objects are not further than R) by

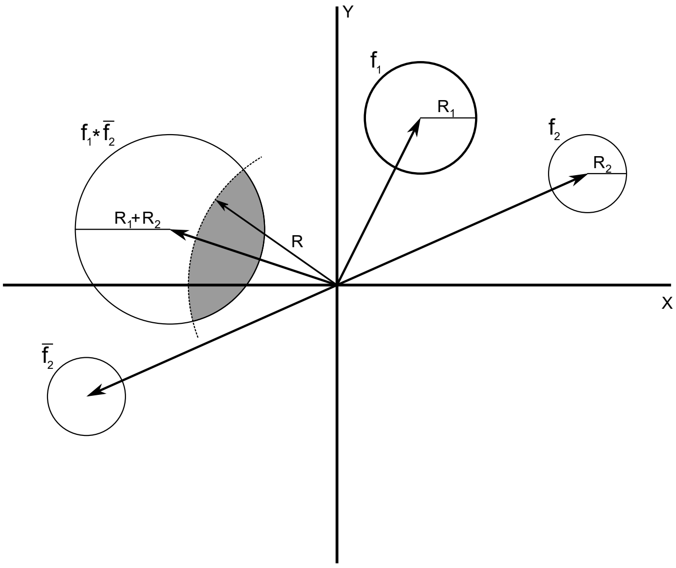


Figure 2: Probability density functions and random variables used for modeling the problem: f_1, f_2, $\bar{f}_2$, and $f_1 * \bar{f}_2$.

the number of all the possible difference vectors (all the possible locations of the objects). It is well known that the density function f of the sum of two random variables is the convolution product (denoted by $*$) of the density functions of each random variable. So, the density function of $X + \bar{Y}$ is:

$$f(\mathbf{t}) = (f_1 * \bar{f}_2)(\mathbf{t}) = \int_{\mathbf{R}^2} f_1(\mathbf{x})\bar{f}_2(\mathbf{t}-\mathbf{x})d\mathbf{x}$$

or equivalently:

$$f(\mathbf{t}) = \int_{\mathbf{R}^2} f_1(\mathbf{x}) f_2(\mathbf{x}-\mathbf{t})d\mathbf{x} \qquad (1)$$

since $\bar{f}_2(\mathbf{t}-\mathbf{x}) = f_2(\mathbf{x}-\mathbf{t})$.

Hence, the cumulative distribution function $probL_{O_1}^{O_2}$, representing the probability that the objects are no further than R, is the integral of f over the disk of center $(0,0)$ and radius R, $B(0,R)$:

$$probL_{O_1}^{O_2}(R) = \int_{B(0,R)} f(\mathbf{t})d\mathbf{t} = P(distProb(O_1,O_2) \leq R)$$

In general, the computation of this integral could have a great computational cost since we have to solve numerically two integrals in the plane: one integral to solve the convolution and another one to integrate this convolution over a disk.

4. Case of Study: Uniform Pdfs

In this section, we focus on the case of study where density functions f_1 and f_2 in Equation 1 are uniform *pdfs* over a disk, which is a common assumption in the field of moving objects. The uniform *pdf* is applied when there is no information about the location of an object. In that case, we consider the worst case: The object can be in any place of the uncertainty area with equal probability. Another reason to consider the uniform distribution is its simplicity and feasibility (see [11] for a extended discussion).

Thus, we will denote by $f_{\mathbf{l}}^R$ the uniform probability density function over a disk centered in a location $\mathbf{l} \in \mathbf{R}^2$ and radius $R \in \mathbf{R}$:

$$f_{\mathbf{l}}^R(\mathbf{t}) = \begin{cases} \dfrac{1}{\pi R^2}, & \mathbf{t} \in B(\mathbf{l}, R) \\ 0, & \mathbf{t} \notin B(\mathbf{l}, R) \end{cases}$$

In the formula, πR^2 obviously represents the area of the disk. In this case, we can give a closed formula for the convolution of two functions, $f_{\mathbf{l}_1,\mathbf{l}_2}^{R_1,R_2}(\mathbf{t}) = f_{\mathbf{l}_1}^{R_1} * \bar{f}_{\mathbf{l}_2}^{R_2}(\mathbf{t})$, doing some geometrical calculations:

$$f_{\mathbf{l}_1,\mathbf{l}_2}^{R_1,R_2}(\mathbf{t}) = \begin{cases} \dfrac{1}{\pi^2 R_1^2 R_2^2} A(\|\mathbf{t}\|) & \mathbf{t} \in B(\mathbf{l}_1 - \mathbf{l}_2, R_1 + R_2) \\ 0 & \text{otherwise} \end{cases}$$

where $A(\|\mathbf{t}\|)$ is the area of $B((0,0), R_2) \cap B((\|\mathbf{t}\|, 0), R_1)$ and $\|\mathbf{t}\|$ represents the norm of the vector $\mathbf{t}$, as shown in Figure 3.

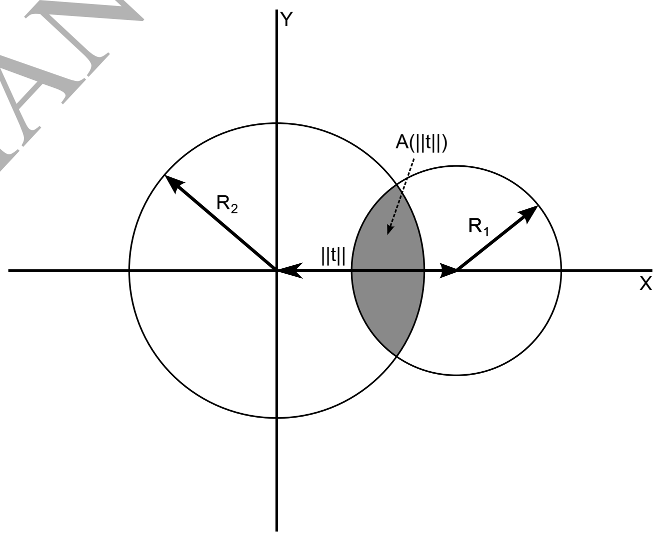


Figure 3: Inside constraint: $A(\|\mathbf{t}\|)$ for the uniform probability density function case.

If we calculate the area $A(\|\mathbf{t}\|)$, the function $f_{\mathbf{l}_1,\mathbf{l}_2}^{R_1,R_2}(\mathbf{t})$ is equal to:

$$\begin{cases} \dfrac{1}{\pi \max(R_1^2, R_2^2)}, & \|\mathbf{l}_1 - \mathbf{l}_2 - \mathbf{t}\| \leq |R_1 - R_2| \\ \dfrac{F_{R_1,R_2}(\|\mathbf{t}\|)}{\pi^2 R_1^2 R_2^2}, & |R_1 - R_2| \leq \|\mathbf{l}_1 - \mathbf{l}_2 - \mathbf{t}\| \leq R_1 + R_2 \\ 0, & \text{otherwise} \end{cases}$$

where $F_{R_1,R_2}(x) = A_{R_1}^{R_2}(x) + A_{R_2}^{R_1}(x) - T_{R_1,R_2}(x)/2$, and:

$$A_{k_1}^{k_2}(x) = k_1^2 \arccos(\frac{k_1^2 - k_2^2 + x^2}{2k_1 x}) \quad (2)$$

$$T_{R_1,R_2}(x) = \sqrt{4R_1^2R_2^2 - (R_1^2 + R_2^2 - x^2)^2} \quad (3)$$

In Figure 4, the shape of a function $f_{\mathbf{l}_1,\mathbf{l}_2}^{R_1,R_2}(\mathbf{t})$ can be seen. The top circle has radius $|R_1 - R_2|$, the bottom one has radius $R_1 + R_2$, and its center is situated in $\mathbf{l}_1 - \mathbf{l}_2$.

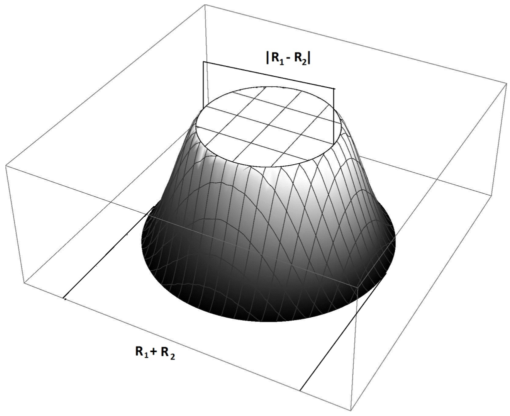


Figure 4: Shape of the function $f_{\mathbf{l}_1,\mathbf{l}_2}^{R_1,R_2}(t)$ for the uniform probability density function case. The center of the bottom circle is at $\mathbf{l}_1 - \mathbf{l}_2$.

It is easy to see that Figure 4 is the solid of revolution generated by the curve (see Figure 5) $g\colon \mathbf{R} \rightarrow \mathbf{R}$, defined as:

$$g(x) = \begin{cases} \frac{1}{\pi R_2^2}, & 0 \leq x \leq R_2 - R_1 \\ \frac{1}{\pi^2 R_1^2 R_2^2} F_{R_1,R_2}(x), & R_2 - R_1 \leq x \leq R_2 + R_1 \\ 0, & \text{otherwise} \end{cases}$$

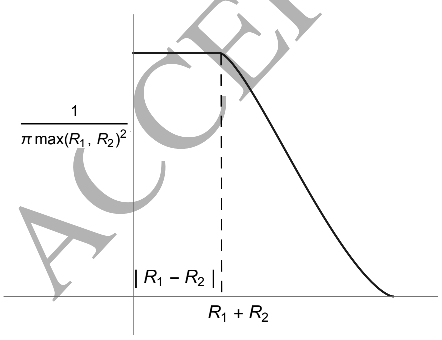


Figure 5: Curve that generates the function $f_{\mathbf{l}_1,\mathbf{l}_2}^{R_1,R_2}$ for the uniform probability density function case.

Therefore, the probability that the distance between objects O_1 and O_2 with associated density functions $f_{\mathbf{l}_1}^{R_1}$ and $f_{\mathbf{l}_2}^{R_2}$ is less than R can be computed by:

$$probL_{O_1}^{O_2}(R) = \int_{B(0,R)} f_{\mathbf{l}_1,\mathbf{l}_2}^{R_1,R_2}(\mathbf{t})d\mathbf{t} = 2\pi \int_0^R xg(x)dx \quad (4)$$

Due to the symmetry of the problem, the following proposition can be proved, that will be used to simplify the computation of $probL_{O_1}^{O_2}(R)$:

Proposition 1. *Let $f_{\mathbf{l}_1}^{R_1}$ and $f_{\mathbf{l}_2}^{R_2}$ be uniform density functions over the disks of center $\mathbf{l}_1$ and $\mathbf{l}_2$, and radius R_1 and R_2, respectively. If we denote by d_0 the distance from $\mathbf{l}_1$ to $\mathbf{l}_2$, $d_0 = \|\mathbf{l}_1 - \mathbf{l}_2\|$, then:*

$$\int_{B(0,R)} f_{\mathbf{l}_1,\mathbf{l}_2}^{R_1,R_2}(\mathbf{t})d\mathbf{t} = \int_{B(0,R)} f_{(0,0),(d_0,0)}^{R_1,R_2}(\mathbf{t})d\mathbf{t}$$

Hence, the value of $probL_{O_1}^{O_2}(R)$ for uncertainty granules with uniform probability density functions over disks depends only on the distance between the centers of the disks.

A geometrical interpretation of the value $probL_{O_1}^{O_2}(R)$ is shown in Figure 6: $probL_{O_1}^{O_2}(R)$ is the volume over the circumference of radius R and the function $f_{\mathbf{l}_1,\mathbf{l}_2}^{R_1,R_2}$.

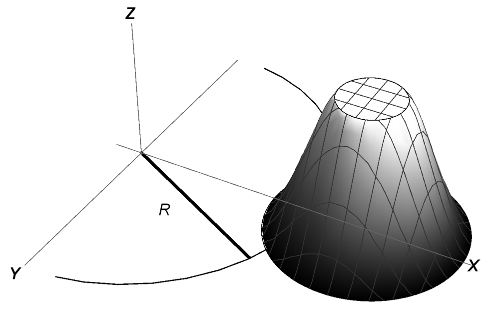


Figure 6: Interpretation of $probL_{O_1}^{O_2}(R)$ for uncertainty granules with uniform probability density functions.

It is not possible to give a closed formula to compute the value of $probL_{O_1}^{O_2}(R)$ when O_i has an arbitrary uniform *pdf* $f_{\mathbf{l}_i}^{R_i}$, so it could be necessary to compute it numerically. That has a great impact on the performance, as we will see in Section 5. To avoid this problem, we will sieve the objects that must be or cannot be in the answer set, so the integral will be computed for a small set of objects. In the next section, we will explain how to perform this sifting. We will see that we can find a closed formula to compute $probL_{O_1}^{O_2}(R)$ if $\mathbf{l}_1 = \mathbf{l}_2$ in Equation (4) (and so, in that case, we do not need to compute it numerically). Moreover, we will explain how we can obtain an upper and a lower bound for $probL_{O_1}^{O_2}(R)$ when $\mathbf{l}_1 \neq \mathbf{l}_2$, which allow us to sieve many of the candidates with less effort.

4.1. Optimization when a Minimum Probability is Required: Probability Threshold

As we mentioned in the previous section, the integral of Equation (4) has to be solved numerically. In a scenario with thousands of objects, if we want to retrieve the objects that are not further than R from a reference object with a given probability (*probability threshold*), the integral has to be calculated numerically thousands of times, which is computationally expensive. In this section, we will describe how to sift out the objects to avoid calculating the integral as many times as possible. We start with some notations that we will use in the rest of the paper.

Definition 3. *We denote by $C_{\mathbf{l}}^{d_1,d_2}$ the annulus formed by two circles centered at $\mathbf{l}$ and radius d_1 and d_2 ($d_1 < d_2$, see Figure 7.a) :*

$$C_{\mathbf{l}}^{d_1,d_2} = B(\mathbf{l}, d_2) \setminus B(\mathbf{l}, d_1)$$

We also denote by $C_{\mathbf{l}}^{d_1,d_2}(\theta_1, \theta_2)$ the portion of the annulus $C_{\mathbf{l}}^{d_1,d_2}$ between the lines passing through $\mathbf{l}$ with slopes equal to θ_1 and θ_2, that is (see Figure 7.b):

$$C_{\mathbf{l}}^{d_1,d_2}(\theta_1, \theta_2) = \{\mathbf{y} \in C_{\mathbf{l}}^{d_1,d_2} \mid \theta_1 \leq \widehat{\mathbf{y} - \mathbf{l}} \leq \theta_2\}$$

where $\widehat{\mathbf{y} - \mathbf{l}}$ is the angle between $\mathbf{y} - \mathbf{l}$ and the x-axis.

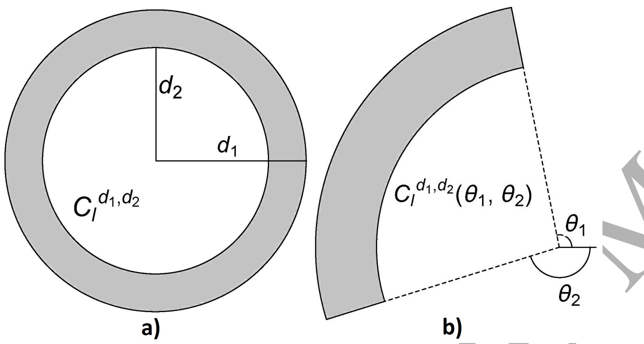


Figure 7: Geometric interpretation of an annulus, $C_{\mathbf{l}}^{d_1,d_2}$ (a); and a portion of an annulus, $C_{\mathbf{l}}^{d_1,d_2}(\theta_1, \theta_2)$ (b).

Definition 4. *We will denote by $B_{R_1,R_2}(x)$ the value of the integral:*

$$B_{R_1,R_2}(x) = \int_{B(0,x)} f_{\mathbf{l}_0,\mathbf{l}_0}^{R_1,R_2}(\mathbf{t})d\mathbf{t}$$

Note that $B_{R_1,R_2}(x)$ does not depend on the choice of $\mathbf{l}_0$ by Proposition 1. Geometrically, the value $B_{R_1,R_2}(x)$ is the volume of the intersection of a cylinder of radius x centered on the coordinate origin and $f_{\mathbf{l}_0,\mathbf{l}_0}^{R_1,R_2}$.

Using the notation of Equations (2) and (3), it can be shown that $B_{R_1,R_2}(x)$ is equal to:

$$\begin{cases} \dfrac{x^2}{\max(R_1^2, R_2^2)}, & 0 \leq x \leq |R_1 - R_2| \\ \dfrac{1}{\pi R_1^2 R_2^2}[x^2(A_{R_1}^{R_2}(x) + A_{R_2}^{R_1}(x)) + R_2^2 A_{R_1}^{x}(R_2) & \\ \quad -1/4(R_1^2 + R_2^2 + x^2)T_{R_1,R_2}(x)], & \\ & |R_1 - R_2| < x < R_1 + R_2 \\ 1, & x \geq R_1 + R_2 \end{cases} \tag{5}$$

The basic idea of the sifting process is that $B_{R_1,R_2}(x)$ can be calculated using the closed formula (5).

Let $S = \{O_{t_i} \mid 1 \leq t \leq m\}$ be a set of target objects and O_r be a reference object. The process of sifting the set S is based on obtaining a lower and an upper bound for the value of the integral corresponding to $probL_{O_r}^{O_{t_i}}(R)$. If we want to retrieve the objects of S that are not further than R from O_r with a probability greater than p, and it is known that $probL_{O_r}^{O_{t_i}}(R) \in [l_i, u_i]$, then the objects such that $p < l_i$ are in the answer set and the objects with $u_i < p$ are not in the answer set. So, we have to calculate the integral only for the objects where $p \in [l_i, u_i]$.

Now, we will explain in detail how to get the lower and the upper bound, $[l, u]$, for $probL_{O_r}^{O_t}(R)$, where O_r and O_t are the uncertainty granules with associated density functions $f_{\mathbf{l}_1}^{R_1}$ and $f_{\mathbf{l}_2}^{R_2}$, respectively. By Proposition 1, we can assume that $\mathbf{l}_1 = (0, 0)$ and $\mathbf{l}_2 = (d_0, 0)$, where d_0 is the distance from $\mathbf{l}_1$ to $\mathbf{l}_2$. We want to give an upper and a lower bound of the volume of $f_{\mathbf{l}_1,\mathbf{l}_2}^{R_1,R_2}$ under the area $A = B((0, 0), R) \cap B((d_0, 0), R_1 + R_2)$ (gray area in Figure 8):

$$probL_{O_r}^{O_t}(R) = \int_A f_{\mathbf{l}_1,\mathbf{l}_2}^{R_1,R_2}$$

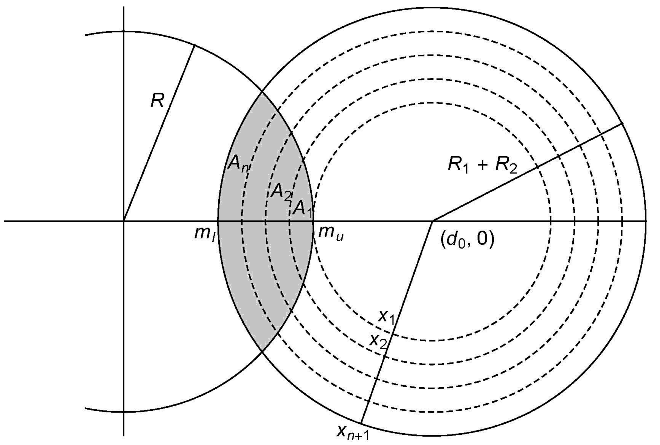


Figure 8: Annuli used to give a lower and upper bound for the sieve.

Let us denote by $[m_l, m_u]$ the interval that is the intersection of A and the x-axis. We divide the interval $[d_0 - m_u, d_0 - m_l]$ in n equal parts:

$$[d_0 - m_u, d_0 - m_l] = \cup_{i=1}^{n}[x_i, x_{i+1}]$$

where obviously $x_1 = d_0 - m_u$ and $x_{n+1} = d_0 - m_l$. If we denote by $C_i = C_{(d_0,0)}^{x_i,x_{i+1}}$ the annulus centered at $(d_0, 0)$ and radius x_i and x_{i+1}, we can split the area A into n parts, $A = A_1 \cup \cdots \cup A_n$ with $A_i = C_i \cap A$ (see Figure 8).

Let α_i be the maximum angle such that $L_i = C_{(d_0,0)}^{x_i,x_{i+1}}(\alpha_i, -\alpha_i)$ holds that $L_i \subseteq A_i$ (see Figure 9.a), and β_i be the minimum angle such that $A_i \subseteq U_i = C_{(d_0,0)}^{x_i,x_{i+1}}(\beta_i, -\beta_i)$ (see Figure 9.b). It is clear that $L = \cup_{i=1}^{n} L_i \subseteq A \subseteq U = \cup_{i=1}^{n} U_i$. Therefore:

$$l = \int_L f_{\mathbf{l}_1,\mathbf{l}_2}^{R_1,R_2} \leq \int_A f_{\mathbf{l}_1,\mathbf{l}_2}^{R_1,R_2} \leq \int_U f_{\mathbf{l}_1,\mathbf{l}_2}^{R_1,R_2} = u$$

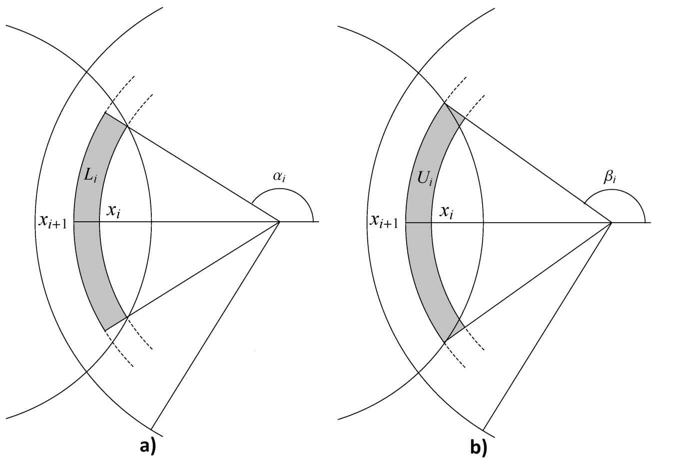


Figure 9: Minimum bound for A_i (a); and maximum bound for A_i (b).

Hence, l and u are bounds satisfying that $probL_{O_r}^{O_t} \in [l, u]$.

Now, let us note that the value l can be calculated using $B_{R_1,R_2}(x)$ (see Definition 4):

$$l = \int_L f_{\mathbf{l}_1,\mathbf{l}_2}^{R_1,R_2} = \sum_{i=1}^{n} \int_{L_i} f_{\mathbf{l}_1,\mathbf{l}_2}^{R_1,R_2} =$$
$$= \sum_{i=1}^{n} \frac{(\pi - \alpha_i)}{\pi} (B_{R_1,R_2}(x_{i+1}) - B_{R_1,R_2}(x_i)) =$$
$$= \sum_{i=1}^{n+1} \frac{M_i}{\pi} B_{R_1,R_2}(x_i)$$

where $M_i = \pi - \alpha_i$ if $i = 1$ or $i = n + 1$, and $M_i = \alpha_i - \alpha_{i-1}$ if $i = 2, \ldots, n$. In a similar way, we can deduce that the upper bound u is equal to:

$$u = \sum_{i=1}^{n+1} \frac{N_i}{\pi} B_{R_1,R_2}(x_i)$$

where $N_i = \pi - \beta_i$ if $i = 1$ or $i = n + 1$, and $N_i = \beta_i - \beta_{i-1}$ if $i = 2, \ldots, n$.

Finally, the angles α_i and β_i can be also computed with a closed formula since, basically, their computation is equivalent to finding the point where $B((0, 0), R)$ and $B((d_0, 0), x_i)$ intersect (see the notation of Equation 2):

$$\alpha_i = A_{R_1+R_2}^{R}(x_i)/(R_1 + R_2)^2$$
$$\beta_i = A_{R_1+R_2}^{R}(x_{i+1})/(R_1 + R_2)^2$$

Therefore, we can quickly calculate a lower and an upper bound for $probL_{O_r}^{O_t}$, since we have a closed formula. In Section 5, we will see empirically the benefits of these bounds.

5. Experimental Evaluation

To validate our proposal in terms of performance and scalability, we have performed an extensive experimental evaluation. In this section, we first specify the implementation details of our prototype and the experiment settings (Section 5.1). Then, we present some experimental results in two different sections: Section 5.2 is dedicated to tests focused on measuring the filtering capabilities of our approach, while Section 5.3 is dedicated to tests focusing on evaluating the performance over both synthetic and real datasets.

5.1. Experimental Settings

We have developed a prototype to show the feasibility of our approach, capable of processing probabilistic location-based queries in the presence of location uncertainty. Our prototype has been implemented using Java 1.7 as the programming language, which has allowed us to test it on Desktop PCs and Android devices (see specifications later).

In the experiments, we have used both synthetic and real datasets. We have generated five sets of 1,000,000 objects each, which are randomly placed in a 4000x4000 grid, as synthetic datasets, which we have made available at `http://sid.cps.unizar.es/projects/ProbabilisticQueries/datasets/`. As real dataset, we have used the LB dataset, a dataset containing the minimum bounding rectangles (MBRs) of Long Beach county roads. It contains 53K pairs of points, but we have considered each of the points in the tuples as a possible location for an object, leading to a 106K point dataset. This dataset has been produced by the Tiger project of the US Census Bureau, and it is downloadable at the R-tree portal, `http://www.rtreeportal.org`.

We focus on evaluating the query processing where a minimum probability threshold has been established in the query. We take as ground-truth probability values the ones obtained using the Simpson's rule[2] in 2D for integration to calculate the integrals using 100 steps for each dimension. Moreover, we consider that the uncertainty area of all the reference and the target objects is a disk and the *pdf* associated is the uniform distribution over the disk[3]. The specific uncertainty and query range radii varies from experiment to experiment depending on the particular aspect of our approach we are evaluating, and it will be specified at the beginning of each of the subsequent experiment sections.

In order to effectively assess the benefits of our approach in presence of possible spatial indexing techniques, our prototype is able to use a filter based on a PR-tree [2] to index and prefilter the objects in the scenario using the minimum bounding boxes (MBBs) of the uncertainty areas of the objects[4]. The main idea behind this prefiltering step is that whenever the MBB of an object does not overlap the MBB of the uncertainty area of the reference object extended by the range radius of the query, the probability that such an object belongs to the answer set is 0,

[2] Simpson's rule is a well known integration method. A brief introduction to this method can be found in `http://en.wikipedia.org/wiki/Simpson's_rule` or in `http://spikedmath.com/336.html`.

[3] Similar to the error given by a GPS position, assuming no further information on the probabilities.

[4] We have used the implementation by Robert Olofsson, which can be obtained at http://www.khelekore.org/prtree/, last accessed 8th December, 2016.

and can be directly discarded without performing further calculations. Thus, when this index-based filter is used, our sieve is applied to the set of objects that are not pruned by it. In our experiments, whenever this index-based filter is used, it will be explicitly stated; otherwise, the evaluation is performed over all the objects in the scenario without using any index.

Finally, the experiments were run on a computer with an Intel Core i5-2320 processor running at 3.00 GHz and 16 GB of RAM memory, and on two different Android smartphones, a Galaxy Nexus (Android 4.2.1, Texas Instruments OMAP 4460 dual-core 1.2 GHz, 1 GB RAM, released in 2011), and a BQ M5 (Android 5.1.1, Qualcomm Snapdragon 615 octa-core 1.5 GHz, 2 GB RAM, released in 2015).

5.2. Evaluating Filtering Capabilities

The first set of tests was focused on evaluating the filtering capabilities of the method presented in Section 4.1, that is, the percentage of objects that are correctly evaluated to be part or not of the answer without having to calculate the exact probability. The filtering efficiency was measured in two different settings: with no previous object indexing (i.e., the calculations are performed against all the objects in the scenario, regardless their position), and using a PR-tree to prefilter all the objects that could never be part of the answer set. This latter prefiltering is performed taking into account the minimum bounding box of the uncertainty area of each object in the scenario, and the minimum bounding box of the query relevance area (defined by the uncertainty radius plus the range radius): if such minimum bounding boxes do not overlap the object is discarded, as the probability that it belongs to the answer set is 0.

In these experiments, we set high levels of uncertainty (uncertainty radius range 50-500 distance units), along with large inside radius (500-1500 distance units, which is from 12.5% to 37.5% of the length of the scenario) to involve as many objects as possible. Thus, we evaluate the performance in difficult scenarios. The values that we have considered for the parameters in a query $q(\mathcal{D}, O_r, R, p)$, as described in Definition 2, can be seen in Table 3. We have run queries for five locations of the reference object in the 4000x4000 grid, and all the possible combinations of the other parameters.

Table 3: Parameters for the filtering experiments of a query $q(\mathcal{D}, O_r, R, p)$.

Total target objects in $\mathcal{D}$	10^2, 10^3, 10^4, 10^5
Locations of the reference object (O_r)	(2000, 2000) (2000±1500, 2000±1500)
Radius for uncertainty area of reference and target objects	50, 100, 250, 500
R (inside radius)	500, 1000, 1500
p (probability threshold)	0.25, 0.50, 0.75 1.0
Number of annuli	16, 32, 64

The results of the first filtering experiment (no preindexing) are shown in Figure 10, which displays the mean of the filtering rate when 16, 32, and 64 annuli are used to obtain the filtering bounds. For instance, the mean of the filtering rate for all the possible queries combining the parameters of Table 3 with 100 objects and using 64 annuli is 99.92%. We can observe that the numerical integral to obtain the exact probability has to be calculated for less than 1% of the target objects.

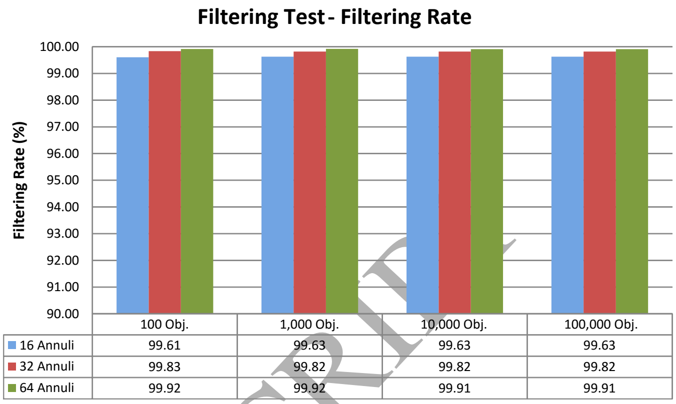


Figure 10: Filtering rate varying the number of annuli without indexing.

In Figure 11, the results of applying our approach after prefiltering the objects using a PR-tree based filter are shown. The graph should be read as follows:

- The first column of each group displays the mean percentage of target objects that the PR-tree based filter has been able to filter out. For example, for 100,000 objects, it was able to detect and remove 44.86% of the objects that were not going to be part of the answer set (probability 0) thanks to the use of the index.
- Each of the associated columns is the mean of the filtering rate of our approach over the results of the prefiltering step. Following with the previous example, for 100,000 objects and using 32 annuli, a 55.14% of the objects in the scenario were passed to our filtering approach, which was able to successfully determine whether the objects belonged or not to the answer set while avoiding the calculation of the integral in 99.32% of the cases.

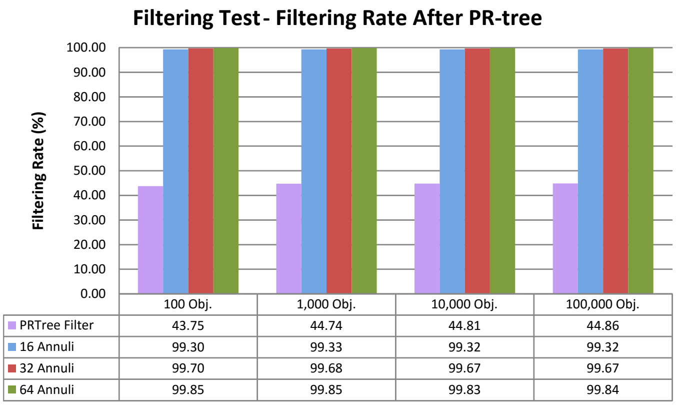


Figure 11: Filtering rate using the PR-tree-based filter and varying the number of annuli.

In all the filtering experiments, we can see how the amount of annuli used in the approximation has a direct impact on the filtering rate. However, there is a balance between the precision of the approximation and the speedup obtained by the filtering, as we will see in the following section. For completeness's sake, we include in Figure 12 the performance graph of these experiments, which displays the mean execution times grouped by the number of objects in the scenario. There, the performance of the base approach using indexes (i.e., a PR-tree based approach without our sieve step) is compared to our approach without using an index (i.e., calculating the query against all the objects in the scenario).

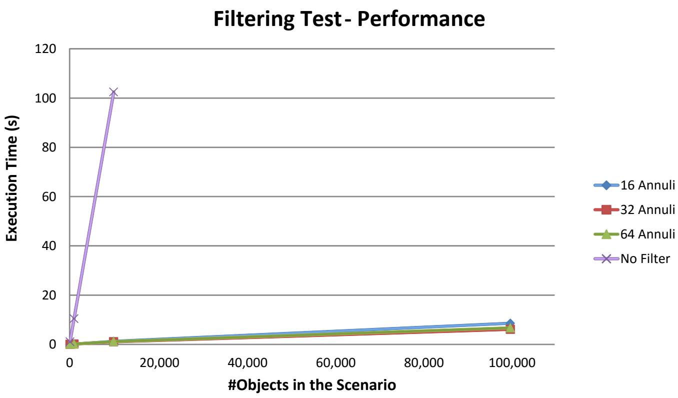


Figure 12: Performance of our approach in the filtering tests.

5.3. Evaluating the Performance and Scalability

The second set of tests was focused on testing the performance and scalability of our algorithm using both synthetic and real datasets.

Using Synthetic Datasets

In this set of experiments, we reduced the uncertainty (the uncertainty radius ranged from 10 to 50 units) and the radius for the range queries (from 100 to 500 units), but we increased the number of objects in the scenario up to 1,000,000 objects (see Table 4).

Table 4: Parameters for the performance experiments of a query $q(\mathcal{D}, O_r, R, p)$.

Total target objects in $\mathcal{D}$	5×10^4, 10^5, 5×10^5, 10^6
Locations object reference (O_r)	(2000, 2000) (2000±1500, 2000±1500)
Radius for uncertainty area of reference and target objects	10, 30, 50
R (inside radius)	100, 300, 500
p (probability threshold)	0.25, 0.50, 0.75
Number of annuli	16, 32, 64

We measured the performance both not using the index-based filtering and using it, whose mean results grouped by the number of objects in the scenario can be seen in Figures 13 and 14, respectively. In the graphs, we can see the balance between the precision of the approximation (number of annuli) and the performance achieved. As the number of annuli increases, our approximation is more expensive in terms of execution time and, although it is linear in the number of objects in the scenario, such cost becomes more significant than the cost of calculating the avoided numeric integrals[5].

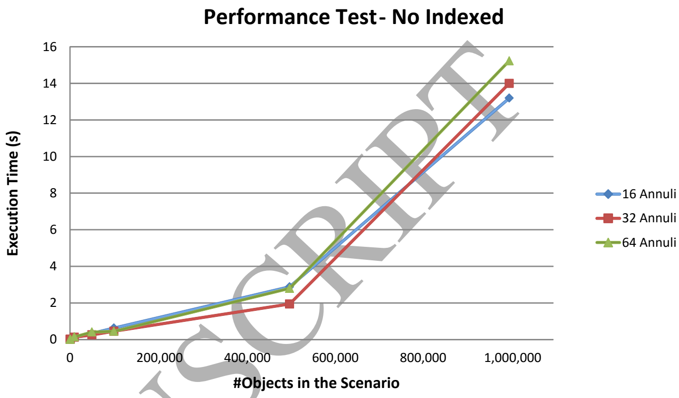


Figure 13: Performance and influence of the number of annuli in our approach when no index is used.

Figure 14: Performance and influence of the number of annuli in our approach using a PR-tree based indexing.

This can be seen in both figures, where the times spent for the tests with 64 annuli are slightly above the times spent for the tests with 32 annuli. Moreover, in Figure 13, note how for 1,000,000 objects, the times for the tests with 32 annuli are no longer below the performance for 16 annuli due to this balance. This does not happen for the indexed tests as, thanks to the use of the PR-tree, our approach is not applied to all the objects in the scenario.

Needless to say, if we did not use the filter bounds, the execution time would be huge, which clearly shows the interest and the benefits of the proposed filtering technique.

[5]This depends directly on the numeric method used to calculate the integrals. In our case, the Simpson's algorithm is quadratic in the number of steps.

Using a Real Dataset

In order to evaluate our approach in a real scenario, we conducted a set of performance experiments on the LB dataset (described in Section 5.1). To do so, we took the same parameters used in [36] with the difference that we forced the reference object of the query to follow a trajectory of 25 steps which crossed the cloud of points of the dataset (selecting the position of the reference object randomly led to many queries with no objects in the answer and biased the results, so we chose a more challenging scenario). The parameters used for these tests can be found in Table 5. Based on the results shown in the previous set of experiments, we set the number of annuli to 32 as it is the optimal value to achieve the best performance in a scenario with such number of objects. Besides testing our approach with a real spatial dataset, we also wanted to evaluate our approach on mobile devices to determine whether it could be used in devices with limited capabilities. Thus, we implemented an Android version of the tests, and ran it on two different devices (described in Section 5.1).

Table 5: Parameters for the LB dataset experiments of a query $q(\mathcal{D}, O_r, R, p)$.

Total target objects in $\mathcal{D}$	106×10^3
Locations of the reference object (O_r)	25 steps within the LB cloud of points
Radius for uncertainty area of reference and target objects	5, 100, 250
R (inside radius)	250, 500, 750
p (probability threshold)	0.25, 0.50, 0.75
Number of annuli	32

The filtering rates and performance results of these tests can be seen in Figures 15 and 16, respectively. There, the results are grouped by uncertainty radius, e.g., the label LB-5 represents the aggregated results of the tests using the scenario LB with an uncertainty radius of 5 distance units.

In Figure 15, we can see how the filtering rates achieved by our approach are consistent with those achieved in the synthetic scenarios. The figures should be read as follows. *PRTree Prefilter* is the mean filtering rate achieved by the PRTree prefilter, *Filter* is the mean filtering rate of our approach when applied to the objects that the prefiltering step has not discarded (similar to Figure 11) and, finally, *Total* is the global mean filtering of our proposal independently of using the index-based prefilter or not (similar to Figure 10). Note how the efficiency of the PRTree-based prefilter decreases as the uncertainty and query radii increase. This is due to the fact that the difference between the area of relevance (i.e., the area where the objects must lay to be part of the answer set) and its minimum bounding box increases as the sum of the uncertainty radius and the inside range radius grows, leading to more false positives (i.e., objects whose probability of being part of the answer set is 0 but are not discarded in the prefiltering step using the PRTree); however, all of those false positives are quickly removed by our sieve. Of course, these filtering rates are exactly the same for both PC and Android devices.

Regarding the performance, we can see the benefits of using our approach even not having indexed the objects in the scenario, as the calculations needed to sieve the results are faster than calculating their exact probabilities. Besides, the results obtained for Android devices suggest that this technique could be carried out by the mobile devices, without having to rely on a back-end server to perform the calculations.

6. Related Work

Inside queries [8, 17] have been studied in the literature of spatio-temporal and moving object databases (see [23] for a survey). However, existing works on location-dependent query processing usually implicitly assume GPS locations for the objects in a scenario. Although some proposals acknowledge the importance of considering different location resolutions (e.g., [19]), the processing of classical constraints such as *inside* is not considered in that context.

The importance of dealing with the uncertainty of location information is emphasized in the literature. Probabilistic queries, even though not in the context of moving objects, were introduced in [10]. For moving objects, probabilistic queries are usually computed by estimating the locations of the objects through a probability density function that models the uncertainty, in a way that the probability that an object is within a certain region can be computed by integration (see [11]). As solving these integrals is frequently expensive (numerical methods are usually required), an index-based filtering step is introduced to prune the search space. Different relevant proposals exist in the literature. For example, probabilistic range queries are the focus of [36], and probabilistic nearest neighbor queries are studied in [7, 26]. In [44], the authors present a model to index uncertain objects and solve probabilistic queries.

In particular, as we mentioned in Section 2.2, the processing of probabilistic inside query has been studied previously, mainly, in [1, 36, 43, 44]. In all these works, the bottleneck in the performance is the calculus of numerical integrals. So, the fewer integrals have to be calculated, the better is the performance. In [36], the authors propose an index, called U-tree, that avoids doing some of these integrals. The U-tree is an R*-tree augmented with some information about the *pdf* of the uncertain objects. This information is based on PCRs (probabilistically constrained rectangles), an extension of the concept of minimum bounding rectangles for uncertain areas. The filter capabilities using the U-tree behave quite well when the range query is a rectangle, but it has worse results when the range query is a circle, as we have in our experiments. Trying to overcome this problem, the authors in [43] construct an index, a UI-tree, which is also a modified R-tree. In this case, the UI-tree holds some disjoint partitions of the uncertain area of the objects together with the cumulative *pdf* in each of these partitions. A different method is proposed in [1], where the authors define an index, called UP-tree, based on the selection of some pivot points that are used to prune (not validate) objects by the reverse triangle inequality. Finally, in [44], the authors define a U-Quadtree, a Quadtree with additional information about the *pdf* of objects. The main difference with the other methods is

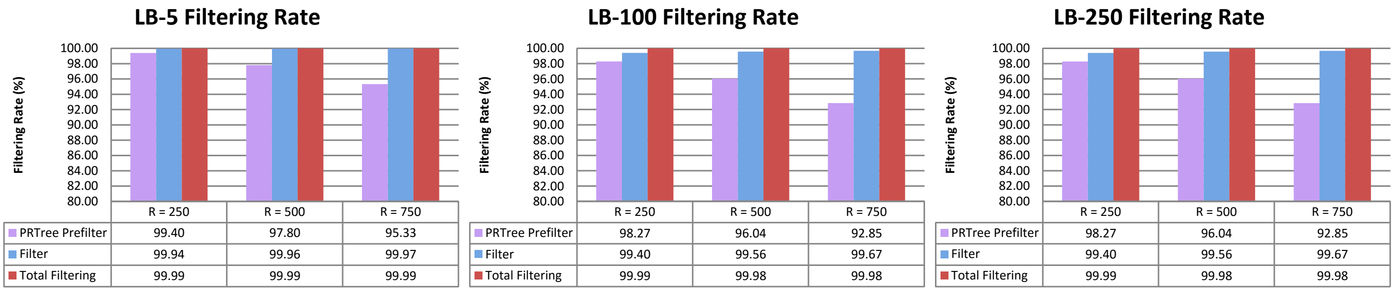


Figure 15: Filtering rate for the different uncertainty values (LB-5,LB-100,LB-250) in the LB dataset experiments.

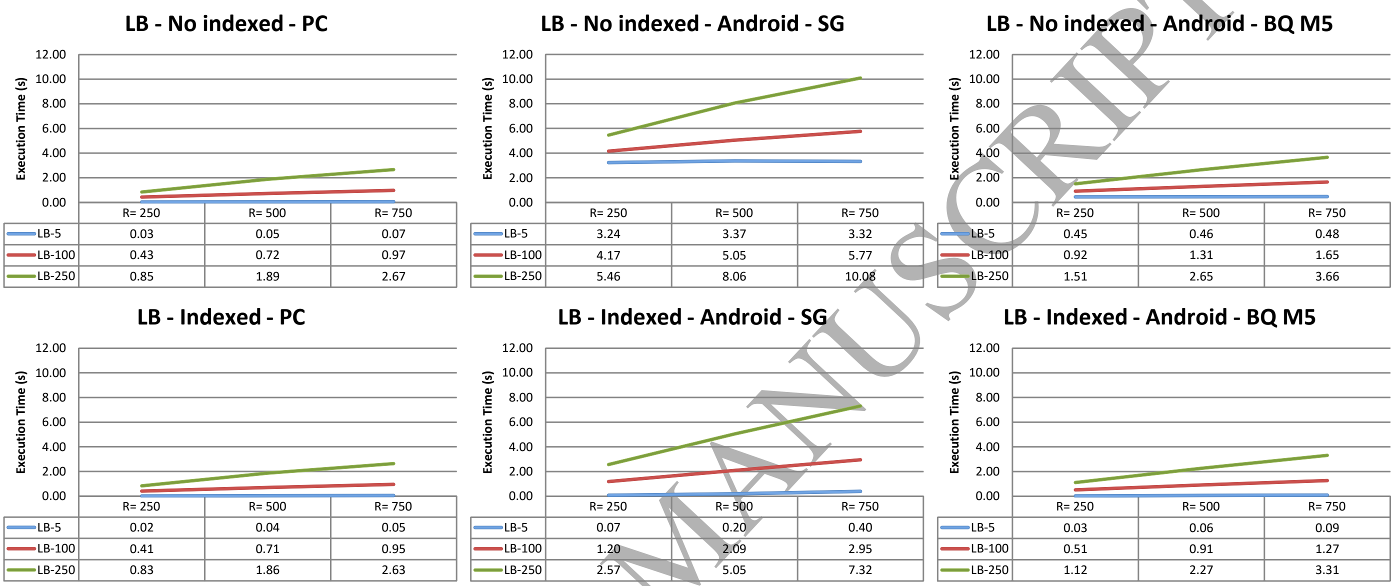


Figure 16: Performance of our approach for the different uncertainty values (LB-5,LB-100,LB-250) on a PC and two Android devices.

that the summary information about the *pdf* saved on the index depends on where the *pdf* has more information.

The previous approaches do not constrain the used *pdfs*; however, they rely on the assumption that pre-indexing the objects is always feasible, which might not be the case in a mobile and volatile environment. Moreover, as shown in the experimental section, our approach would be complementary to their works in order to lower the number of numeric integrals to be solved.

There are several works [6, 9, 13, 24, 28, 31, 45] which consider different types of probabilistic queries (e.g., different variants of nearest neighbour queries and range queries within temporal ranges and trajectories). However, none of them, but for [6], consider that the uncertainty can also affect the queried position (e.g., the query could be expressed in terms also of a moving object, and thus, be subject of uncertainty). The work in [6] focuses on calculating the similarity domination rather than the type query that we are dealing with in our work.

Thus, as opposed to existing related work, in this paper we have presented a model of uncertainty based on the concept of location granule, analyzed the problem of probabilistic inside queries using it, and proposed a different (and complementary to other related works) approach to speed up the calculations needed to solve probabilistic inside queries, where the uniform distribution is considered and which filters a large percentage of the objects of the search space by just performing a few operations.

7. Conclusions and Future Work

Location-dependent queries are the main building block of location-based services, and therefore we have to provide efficient methods to solve them. However, as we have seen, they usually do not take into account the imprecision of the locations (inherent to the mechanism or introduced to force it, for example, for privacy issues). When you consider imprecise locations, the uncertainty in object locations also introduce a heavy-weight problem when solving the already troublesome location-dependent constraints.

In this paper, we have used our extended location model to deal with location imprecision using uncertainty location granules. In particular:

- We have formally analyzed probabilistic inside queries from the point of view of our extended location model.

Our analysis is general enough to be applied to other models in the literature.

- We have presented a method to process efficiently these queries when the underlying probability density function is the uniform distribution over a disk, a challenging problem which is the subject of study in many previous works.
- We have performed an extensive experimental evaluation with both synthetic and real datasets where the efficiency of the described method for the uniform distribution can be appreciated. The results obtained showed the interest of our proposal, even in non-indexed and mobile scenarios (where the calculations could be performed in mobile devices).

We are currently studying how to extend our method for normal *pdfs* and histograms (i.e., how to calculate rapidly an approximation of the integral using annuli), and how our approach could be applied and integrated with other approaches based on indexing. We also plan to study how the use of different location granularities would affect the semantics of probabilistic inside constraints and their efficient calculation. Last but not least, we will study other popular location-dependent constraints (such as *nearest-neighbor queries*, *closest-pairs* [14, 15] and *similarity joins* [14], as well as reverse kNN queries [35]) .

Acknowledgments

This work was supported by the CICYT project TIN2013-46238-C4-4-R and DGA-FSE.

References

References

[1] F. Angiulli, F. Fassetti, Indexing uncertain data in general metric spaces, IEEE Transactions on Knowledge Data Engineering 24 (9) (2012) 1640–1657.

[2] L. Arge, M. D. Berg, H. Haverkort, K. Yi, The priority R-tree: A practically efficient and worst-case optimal R-tree, ACM Transactions on Algorithms 4 (1) (2008) 9:1–9:30.

[3] P. Bellavista, A. Corradi, C. Giannelli, Efficiently managing location information with privacy requirements in Wi-Fi networks: a middleware approach, in: 2nd International Symposium on Wireless Communication Systems (ISWCS'05), Siena, Italy, 2005.

[4] A. Belussi, C. Combi, G. Pozzani, Formal and conceptual modeling of spatio-temporal granularities, in: 13th International Database Engineering & Applications Symposium (IDEAS'09), Cetraro, Calabria, Italy, 2009.

[5] J. Bernad, C. Bobed, E. Mena, S. Ilarri, A formalization for semantic location granules, International Journal of Geographical Information Science 27 (6) (2013) 1090–1108.

[6] T. Bernecker, T. Emrich, H.-P. Kriegel, N. Mamoulis, M. Renz, A. Züfle, A novel probabilistic pruning approach to speed up similarity queries in uncertain databases, in: IEEE 27th International Conference on Data Engineering (ICDE'11), Hannover, Germany, 2011.

[7] G. Beskales, M. A. Soliman, I. F. IIyas, Efficient search for the top-k probable nearest neighbors in uncertain databases, Proceedings of the VLDB Endowment 1 (1) (2008) 326–339.

[8] Y. Cai, K. A. Hua, G. Cao, T. Xu, Real-time processing of range-monitoring queries in heterogeneous mobile databases, IEEE Transactions on Mobile Computing 5 (7) (2006) 931–942.

[9] R. Cheng, J. Chen, M. Mokbel, C. Y. Chow, Probabilistic verifiers: Evaluating constrained nearest-neighbor queries over uncertain data, in: IEEE 24th International Conference on Data Engineering (ICDE'08), Cancun, Mexico, 2008.

[10] R. Cheng, D. V. Kalashnikov, S. Prabhakar, Evaluating probabilistic queries over imprecise data, in: ACM SIGMOD International Conference on Management of Data (SIGMOD'03), San Diego, California, USA, 2003.

[11] R. Cheng, D. V. Kalashnikov, S. Prabhakar, Querying imprecise data in moving object environments, IEEE Transactions on Knowledge and Data Engineering 16 (9) (2004) 1112–1127.

[12] C.-Y. Chow, M. F. Mokbel, H. V. Leong, On efficient and scalable support of continuous queries in mobile peer-to-peer environments, IEEE Transactions on Mobile Computing 10 (10) (2011) 1473–1487.

[13] B. S. Chung, W.-C. Lee, A. L. Chen, Processing probabilistic spatio-temporal range queries over moving objects with uncertainty, in: 12th International Conference on Extending Database Technology (EDBT'09), Saint-Petersburg, Russia, 2009.

[14] A. Corral, Y. Manolopoulos, Y. Theodoridis, M. Vassilakopoulos, Closest pair queries in spatial databases, in: ACM SIGMOD International Conference on Management of Data (SIGMOD'00), Dallas, Texas, USA, 2000.

[15] A. Corral, M. Vassilakopoulos, On approximate algorithms for distance-based queries using R-trees, The Computer Journal 48 (2) (2005) 220–238.

[16] H. Ding, G. Trajcevski, P. Scheuermann, Efficient maintenance of continuous queries for trajectories, Geoinformatica 12 (3) (2008) 255–288.

[17] B. Gedik, L. Liu, MobiEyes: A distributed location monitoring service using moving location queries, IEEE Transactions on Mobile Computing 5 (10) (2006) 1384–1402.

[18] J. Hightower, From position to place, in: 2003 Workshop on Location-Aware Computing, Seattle, Washington, USA, 2003.

[19] C. Hoareau, I. Satoh, A model checking-based approach for location query processing in pervasive computing environments, in: On the Move to Meaningful Internet Systems 2007: OTM 2007 Workshops (PerSys'07), Vilamoura, Algarve, Portugal, 2007.

[20] C. Hoareau, I. Satoh, From model checking to data management in pervasive computing: a location-based query-processing framework, in: 2009 International Conference on Pervasive Services (ICPS'09), London, United Kingdom, 2009.

[21] S. Ilarri, C. Bobed, E. Mena, An approach to process continuous location-dependent queries on moving objects with support for location granules, Journal of Systems and Software 84 (8) (2011) 1327–1350.

[22] S. Ilarri, A. Corral, C. Bobed, E. Mena, Probabilistic granule-based inside and nearest neighbor queries, in: 13th East-European Conference on Advances in Databases and Information Systems (ADBIS'09), Riga, Latvia, 2009.

[23] S. Ilarri, E. Mena, A. Illarramendi, Location-dependent query processing: Where we are and where we are heading, ACM Computing Surveys 42 (3) (2010) 12:1–12:73.

[24] Y. Jin, R. Cheng, B. Kao, K.-Y. Lam, Y. Zhang, A filter-based protocol for continuous queries over imprecise location data, in: 21st ACM International Conference on Information and Knowledge Management (CIKM'12), Maui, Hawaii, USA, 2012.

[25] H.-P. Kriegel, P. Kunath, M. Pfeifle, M. Renz, Probabilistic similarity join on uncertain data, in: 11th International Conference on Database Systems for Advanced Applications (DASFAA'06), Singapore, 2006.

[26] H.-P. Kriegel, P. Kunath, M. Renz, Probabilistic nearest-neighbor query on uncertain objects, in: 12th International Conference on Database Systems for Advanced Applications (DASFAA'07), Bangkok, Thailand, 2007.

[27] J. Krumm, A survey of computational location privacy, Personal and Ubiquitous Computing 13 (6) (2009) 391–399.

[28] E.-Y. Lee, H.-J. Cho, T.-S. Chung, K.-Y. Ryu, Moving range k nearest neighbor queries with quality guarantee over uncertain moving objects, Information Sciences 325 (2015) 324 – 341.

[29] C.-Y. Lin, Y.-C. Tseng, Y.-C. Liu, Imprecision-tolerant location management for object-tracking wireless sensor network, The Computer Journal 53 (3) (2010) 351–364.

[30] M. F. Mokbel, X. Xiong, M. A. Hammad, W. G. Aref, Continuous query processing of spatio-temporal data streams in PLACE, Geoinformatica

9 (4) (2005) 343–365.

[31] J. Niedermayer, A. Züfle, T. Emrich, M. Renz, N. Mamoulis, L. Chen, H.-P. Kriegel, Probabilistic nearest neighbor queries on uncertain moving object trajectories, Proceedings of the VLDB Endowment 7 (3) (2013) 205–216.

[32] S. Prabhakar, Y. Xia, D. V. Kalashnikov, W. G. Aref, S. E. Hambrusch, Query indexing and velocity constrained indexing: Scalable techniques for continuous queries on moving objects, IEEE Transactions on Computers 51 (10) (2002) 1124–1140.

[33] J. H. Schiller, A. Voisard, Location-Based Services, Morgan Kaufmann, San Francisco, California, USA, 2004.

[34] A. P. Sistla, O. Wolfson, S. Chamberlain, S. Dao, Modeling and querying moving objects, in: IEEE 13th International Conference on Data Engineering (ICDE'97), Birmingham, United Kingdom, 1997.

[35] D. Taniar, M. Safar, Q. T. Tran, W. Rahayu, J. H. Park, Spatial network RNN queries in GIS, The Computer Journal 54 (4) (2011) 617–627.

[36] Y. Tao, X. Xiao, R. Cheng, Range search on multidimensional uncertain data, ACM Transactions on Database Systems 32 (3) (2007) 15:1–15:54.

[37] Y. Theodoridis, Ten benchmark database queries for location-based services, The Computer Journal 46 (6) (2003) 713–725.

[38] E. Trevisani, A. Vitaletti, Cell-ID location technique, limits and benefits: An experimental study, in: 6th IEEE Workshop on Mobile Computing Systems and Applications (WMCSA'04), Washington, DC, USA, 2004.

[39] M. van Kreveld, Geo-information and computational geometry, chap. 1. Computational geometry: Its objectives and relation to GIS, Nederlandse Commissie voor Geodesie (NCG), 2006, pp. 1–8.

[40] O. Wolfson, L. Jiang, A. P. Sistla, S. Chamberlain, N. Rishe, M. Deng, Databases for tracking mobile units in real time, in: 7th International Conference on Database Theory (ICDT'99), Jerusalem, Israel, 1999.

[41] T. Yoshimura, H. Hasegawa, Comparing the precision and accuracy of GPS positioning in forested areas, Journal of Forest Research 8 (3) (2003) 147–152.

[42] V. Zeimpekis, G. M. Giaglis, G. Lekakos, A taxonomy of indoor and outdoor positioning techniques for mobile location services, SIGecom Exchanges 3 (4) (2002) 19–27.

[43] Y. Zhang, X. Lin, W. Zhang, J. Wang, Q. Lin, Effectively indexing the uncertain space, IEEE Transactions on Knowledge and Data Engineering 22 (9) (2010) 1247–1261.

[44] Y. Zhang, W. Zhang, Q. Lin, X. Lin, H. T. Shen, Effectively indexing the multidimensional uncertain objects, IEEE Transactions on Knowledge and Data Engineering 26 (3) (2014) 608–622.

[45] K. Zheng, G. Trajcevski, X. Zhou, P. Scheuermann, Probabilistic range queries for uncertain trajectories on road networks, in: 14th International Conference on Extending Database Technology (EDBT'11), Uppsala, Sweden, 2011.