Technical note

Uncertain lines and circles with dependencies

Yonatan Myers, Leo Joskowicz *

School of Engineering and Computer Science, The Hebrew University of Jerusalem, Israel

ARTICLE INFO

Keywords:
Geometric uncertainty
Tolerance

ABSTRACT

Classical computational geometry algorithms handle geometric constructs whose shapes and locations are exact. However, many real-world applications require the modeling of objects with geometric uncertainties. Existing geometric uncertainty models cannot handle dependencies among objects. This results in the overestimation of errors. We have developed the Linear Parametric Geometric Uncertainty Model, a general, computationally efficient, worst-case, linear approximation of geometric uncertainty that supports dependencies among uncertainties. In this paper, we present the properties of the uncertainty zones of a line and circle, defined using this model, and describe efficient algorithms to compute them. We show that the line's envelope has linear space complexity and is computed in low polynomial time. The circle's envelope has quadratic complexity and is also computed in low polynomial time.

1. Introduction

Geometric uncertainty is ubiquitous in CAD/CAM, robotics, wireless networks, and other fields. Measurement and manufacturing processes are intrinsically imprecise. In contrast, geometric models are usually exact. Modeling and computing geometric variability is thus of scientific, technical, and economic importance.

Geometric uncertainty has been addressed using different models in a variety of fields, e.g. mechanical engineering [1–3] and computational geometry [4]. A common approach is to bound point locations using simple shapes, such as rectangles [5], circles [1,2], convex polygons [6] and line segments [7]. The main drawback of these models is their inability to model mutually dependent uncertainties, which are common in practice [3]. Probabilistic models have been suggested [11,12], but they do not handle dependencies.

Assuming error variations are independent may lead to an overestimation of the uncertainty. For example, let c be a point located in the interval [−1,1], and let a and b be two points to the left and right of c, each at a distance of 4 ± 1 from c. a and b are thus in the intervals [−6,−2] and [2,6]. If we assume the uncertainties of a and b are independent, the maximum distance between them is 12. However, neither can be further than 5 from c, so they cannot be more than 10 apart. Ignoring the dependence of a and b overestimates the actual maximum distance by 2 (20%).

Uncertainty has been studied in the context of numerical computation stability. Interval arithmetic provides methods to represent and propagate real-valued number uncertainty intervals [8]. Since interval arithmetic does not account for dependencies, it often yields overestimated intervals. Affine arithmetic [9] improves interval estimation by tracking round-off errors, and allows for interdependence. However, its focus is robust computation, so it does not provide a description of geometric uncertainty. Furthermore, new uncertainty parameters are required for computing non-affine functions, e.g. multiplication. As a result, the estimates provided by this model are often not tight. Other techniques, such as robust, finite precision, and epsilon geometry are applicable only for small variations, and do not take account of interdependencies [10].

Our previous work addressed shape and position uncertainty in mechanical assemblies [3]. We identified the need for an effective geometric uncertainty representation and computation model that handles coupled uncertainties. We recently introduced the Linear Parametric Geometric Uncertainty Model (LPGUM) [13], a general, computationally efficient worst-case, linear approximation of geometric uncertainty that allows for coupling between uncertainties.

In the LPGUM, geometric features, such as points and lines are defined by parameters with uncertainty intervals. The sensitivity of the location of a feature to the parameters is defined by sensitivity matrices. The sensitivity matrices are derived from the functional relations between the design or measurement parameters, by calculating the partial derivative of the function for each parameter. We have developed efficient algorithms for computing the uncertainty zones of points and lines in the plane, and for answering relative position queries and range queries [13,14].

This paper presents an LPGUM line and circle. The main result is that the complexity of the envelope of an LPGUM line is linear, in the number of parameters needed to represent it.
2. The Linear Parametric Geometric Uncertainty Model

A parametric uncertainty model \( q, \bar{q}, \Delta \) is defined as follows. Let \( q = [q_1, \ldots, q_k]^T \) be a vector of \( k \) parameters over an uncertainty domain, \( \Delta \). Each parameter, \( q_i \in \mathbb{R} \), takes a value from a bound uncertainty interval, \( \Delta_i = [q_i^-, q_i^+] \), and is associated with a nominal value, \( \bar{q}_i \). The parameters' uncertainty domain is \( \Delta = \Delta_1 \times \cdots \times \Delta_k \). The nominal parameter vector, \( q = (q_1, \ldots, q_k) \), is the parameter vector values with no uncertainty. Without loss of generality, we assume that \( q_i^- = q_i^+ \), and that \( \bar{q}_i = 0 \).

Asymmetric domains are transformed into symmetric ones by adjusting the nominal parameter values and uncertainty intervals:

\[
q_i^\text{mid} = (q_i^- + q_i^+)/2, \quad q_i^- = q_i^+ - q_i^\text{mid}, \quad q_i^+ = q_i^- + q_i^\text{mid}.
\]

Interdependent geometric objects are modeled over a common parametric uncertainty model. The object's relations to each parameter determines the dependence between uncertainties. The parameters' intervals and sensitivity matrices are hand-tailored to each application. For example, in mechanical engineering, they are derived from the parts' nominal and tolerance specifications. In distributed wireless sensor networks, they are derived from the sensors' specifications, and from a physical model of the environment. If a statistical model is available, a probability density function may be assigned to the parameters.

2.1. Uncertain dimension

The term ‘dimension’ refers to a quantity or number [15]. An uncertain dimension, \( d(q) \), is defined by a nominal value, \( \bar{d} \), and a \( k \)-dimensional sensitivity vector, \( A_{d0} \), over a parametric uncertainty model. Entry \( (A_{d0})_{ij} \) quantifies the sensitivity of the dimension to parameter \( q_i \): it is zero when the dimension is independent of \( q_i \). The LPGUM of \( d(q) \) is:

\[
d(q) = \bar{d} + A_{d0} q.
\]

The uncertainty zone of dimension, \( d(q) \), is the set of all dimension values for instances of parameter vector \( q \):

\[
Z(d(q)) = \{ d' \mid d' = \bar{d} + A_{d0} q, q \in \Delta \}.
\]

The uncertainty zone of a dimension is the interval \([\min_{q \in \Delta} d(q), \max_{q \in \Delta} d(q)] \subset \mathbb{R} \). The interval endpoints are directly computed by minimizing or maximizing each individual parameter.

We compute \( d^- = \min_{q \in \Delta} d(q) = \bar{d} + \sum_{j=1}^k d_j^+ (A_{d0})_{ij} \) and \( d^+ = \max_{q \in \Delta} d(q) = \bar{d} + \sum_{j=1}^k d_j^- (A_{d0})_{ij} \) by evaluating the expressions for each parameter individually.

Two dimensions, defined over the same parametric model, are dependent, iff they both depend on at least one common parameter. Otherwise, they are independent.

2.2. Uncertain point/vector

An uncertain point, \( v(q) \), is defined by a nominal location, \( \bar{v} \), and a \( 2 \times k \) sensitivity matrix, \( A_v \), over a parametric uncertainty model, \( q, \bar{q}, \Delta \). The uncertain location of the point is \( v(q) = (v_1(q), v_2(q))^T \), where \( v_1(q) \) and \( v_2(q) \) are uncertain dimensions; the nominal location of the point is \( \bar{v} = (\bar{v}_1, \bar{v}_2)^T \). Entry \( (A_v)_{ij} \) quantifies the sensitivity of coordinate \( i \) to parameter \( q_j \), \( i = 1 \) for \( x \), \( i = 2 \) for \( y \); it is zero, if coordinate \( i \) is independent of \( q_j \). When the entire column \( (A_v)_{i.} \) is zero, the point is independent of \( q_i \).

The LPGUM of point \( v(q) \) is:

\[
v(q) = \bar{v} + A_v q.
\]

The LPGUM of vector \( v(q) \) is defined similarly and is denoted using bold face. Hereafter, when we refer to a point, \( u(q) \), we implicitly assume the existence of a corresponding nominal location, \( \bar{u} \), a sensitivity matrix, \( A_u \), and a \( k \)-parameter parametric uncertainty model.

The uncertainty zone of a point is defined as the set of all locations of a point's instances for \( q \in \Delta \):

\[
Z(v(q)) = \{ v' \mid v' = \bar{v} + A_v q, q \in \Delta \}.
\]

The arrangement of lines perpendicular to the vectors \( A_{v0} \), rooted at the origin, is called the cone diagram. Fig. 1 shows an example of an LPGUM point and its cone diagram. In [3], an algorithm for computing the uncertainty zone of a point is provided and the following is proved.

**Theorem 1.** The uncertainty zone boundary of an LPGUM point, \( v(q) \), is a zonotope with at most \( 2k \) vertices, computed in optimal \( O(k \log k) \) time and \( O(k) \) space.

Two uncertain points are said to be dependent, iff they both depend on at least one common parameter.

3. Uncertain line/line segment

An uncertain line, \( l(q) \), is defined as the affine combination of an LPGUM point, \( u(q) \), and vector, \( v(q) \):

\[
l(q) = \bar{u} + A_u q + \alpha (v + A_v q), \quad \alpha \in \mathbb{R}.
\]

For an uncertain line segment \( \alpha \) is restricted to \( \alpha \in [0, 1] \). The point and the vector location uncertainties can be dependent or independent. A line can also be represented as \( y = a(q)x + b(q) \), where \( a(q) \) and \( b(q) \) are dimensions. This representation is less general, as it cannot be used to model vertical lines. A line represented this way can be transformed into an equivalent point-vector representation, by setting \( l(q) = b'(q) + \alpha a'(q) \), where \( b'(q) = (0, b(q))^T \) and \( a'(q) = (1, a(q))^T \).

The uncertainty zone of line \( l(q) \) (e.g. Fig. 2b) is defined as the set of all lines for instances of parameter vector, \( q \):

\[
Z(l(q)) = \{ u' \mid u' = \bar{u} + A_u q + \alpha (v + A_v q), q \in \Delta, \alpha \in \mathbb{R} \}.
\]
4. Line envelope

Let \( l(q) \) be a line, as defined above. Its uncertainty zone could cover the entire plane, in which case it does not have a boundary. This can be detected by considering the uncertainty zone of the vector \( w(q) \). If the origin lies inside the envelope, the uncertainty zone covers the entire plane. If the uncertainty zone does not cover the entire plane, its boundary consists of two curves. The curves are made up of straight line segments, and segments of parabolas. The complexity of the envelope is \( O(k) \).

4.1. Line events

To compute a line's uncertainty envelope, a sweep algorithm, in which \( \alpha \) is swept from \(-\infty \) to \( \infty \), is used. We first describe the events and how to find them. Every value of \( \alpha \) yields an LPGUM point, \( l_\alpha(q) \) (Fig. 2b) with nominal location, \( l_\alpha = \bar{u} + \alpha \vec{v} \), and sensitivity matrix, \( A_\alpha = A_0 + \alpha A_i \).

Every column of \( (A_\alpha)_{+,i} \), the sum \( (A_\alpha)_{+,i} + \alpha (A_\gamma)_{+,i} \). If \( (A_\alpha)_{+,i} \) and \( (A_\gamma)_{+,i} \) are not collinear then, as \( \alpha \) approaches \( -\infty \), the direction of \( (A_\alpha)_{+,i} \) approaches the direction of \( -(A_\gamma)_{+,i} \). For \( \alpha = 0 \), \( (A_\alpha)_{+,i} = (A_\gamma)_{+,i} \), and as \( \alpha \) approaches \( \infty \), the direction of \( (A_\alpha)_{+,i} \) approaches the direction of \( (A_\gamma)_{+,i} \). The transition from \( -(A_\gamma)_{+,i} \) through \( (A_\alpha)_{+,i} \) to \( (A_\gamma)_{+,i} \) is continuous and has a constant direction of rotation. We call such rotation monotonic. As \( (A_\gamma)_{+,i} \) turns monotonically in a range of \( \pi \) radians, the corresponding line in the cone diagram of \( l_\alpha(q) \) also rotates monotonically, in a range of \( \pi \) radians. Every line of the cone diagram may rotate at a different speed and direction, depending on the relative size and direction of \( (A_\gamma)_{+,i} \) and \( (A_\alpha)_{+,i} \). Let \( L_i \) and \( L_j \) be two lines of the cone diagram, perpendicular to \( (A_\gamma)_{+,i} \) and \( (A_\alpha)_{+,i} \). If \( L_i \) and \( L_j \) coincide at a given value, \( \alpha' \), then there exist two cones at \( \alpha' - \epsilon \) bounded by \( L_i \) and \( L_j \), which vanish at \( \alpha' \), and two new cones emerge at \( \alpha' + \epsilon \), which are supported by \( L_i \) and \( L_j \) in reverse order. This is called a switch event. Hereafter, \( l(q) \) is an LPGUM line, defined over a parametric uncertainty model \( (q, \bar{q}, \Delta) \).

**Definition 1.** A switch event is a value of \( \alpha \) for which two lines of the cone diagram of the point \( l_\alpha(q) \) coincide. These are values of \( \alpha \) that satisfy the equation:

\[
\{(A_\alpha), (A_\gamma)_{+,i}^{-1}\} = 0.
\]

(4)

Switch events always involve neighboring lines. We therefore solve this equation only for such lines and new neighbors found in the course of the running algorithm.

When the vectors, \( (A_\alpha)_{+,i} \) and \( (A_\gamma)_{+,i} \), are collinear, there exists a value \( \alpha'' \) for which \( (A_\alpha)_{+,i} \) vanishes. When \( (A_\gamma)_{+,i} \) vanishes, the cones defined by it merge to form one cone.

**Definition 2.** A flip event occurs at a value of \( \alpha \), for which a column vector \( (A_\alpha)_{+,i} \) equals zero. These values of \( \alpha \) satisfy the equation:

\[
(A_\alpha)_{+,i} = 0.
\]

(5)

A third type of event occurs, when \( l_\alpha(q) \), or part of it, turns sufficiently to bring a different part of it in contact with the line’s envelope. This happens when two or more lines of the cone diagram rotate in the same direction, causing part (or all) of the zonotope to rotate. This is called a twist event.

**Definition 3.** A twist event is a pair of values of \( \alpha \) for which no topological event occurs, and the part of \( l_\alpha(q) \) tangential to the envelope changes. These are values of \( \alpha \) that satisfy

\[
l(q^{(i)}) = l(q^{(j)})
\]

(6)

where \( q^{(i)}, q^{(j)} \) are parameterizations of neighboring vertices on the zonotope of \( l_\alpha(q) \).

4.2. Line envelope properties

We now present the properties of the envelope of an LPGUM line.

**Lemma 4.** The two regions outside the uncertainty zone of \( l(q) \) are convex.

**Proof (Outline).** \( Z(l(q)) \) is the union of all instances of \( l(q) \). Hence, there is an instance tangential to every point on the boundary. This implies that the exterior areas are convex. \( \square \)

**Lemma 5.** The region on one side of the uncertainty zone of \( l(q) \) is unbounded.

**Proof (Outline).** All instances of \( l(q) \) encounter all the points on the line in the same order. We assume by contradiction that the region is bound. We then show that there must exist two instances of \( l(q) \) that encounter instances of \( l_\gamma(q) \) in different orders. \( \square \)

**Lemma 6.** Let \( l(q) \) be an LPGUM line. Its boundaries consist of parts of parabolic curves and straight line segments.

**Proof (Outline).** To trace out the line's envelope we compute a zonotope at every event. We then connect points on the boundaries of adjacent zonotopes with straight segments, so that the segments connect points that have identical parameterizations. Fig. 3 illustrates this. We call this action sweeping the zonotope edges. We show analytically that this can create only parabolic curves or straight segments. \( \square \)

**Lemma 7.** Every vertex on the uncertainty envelope of \( l(q) \) is the result of an event.

**Proof (Outline).** We note that between events, the zonotope vertices slide along line instances. Hence, changes which create vertices on the envelope may occur only at an event location, at which either the zonotope changes direction or flips. \( \square \)

An event is relevant, if it affects the envelope of the line.

**Lemma 8.** Let \( l(q) \) be an LPGUM line. On \( l(q) \), there are at most 2k relevant topological events.

We note that, because of the monotonic rotation of the lines, \( l_\alpha \), of the cone diagram, and of the orientation of the vector normal to the line’s envelope, the envelope can move from one side of \( l_\alpha \) to the other only once. This is proved by exploring all possible outcomes of topological events.

**Lemma 9.** Let \( l(q) \) be an LPGUM line. No more than \( O(k) \) twist events affect the line’s uncertainty zone.

We note that twist events occur where there are no topological events, so the neighbor relations between lines of the cone diagram are preserved. We note also that twist events involve only

Please cite this article as: Myers Y, Joskowicz L. Uncertain lines and circles with dependencies. Computer-Aided Design (2012), doi:10.1016/j.cad.2012.10.040
neighboring vertices of the point’s zonotope. As neighbor relations between lines of the cone diagram are preserved, only \( k \) twist events are possible.

Every event creates not more than two vertices on the uncertainty envelope. This is because, at a given value of \( \alpha \), the envelope cannot undergo more than one change. Topological events occur at a single value of \( \alpha \), while twist events are divided between two values of \( \alpha \). This generates at most two vertices on the envelope.

**Theorem 2.** Let \( l(q) \) be an LPGUM line. There are \( O(k) \) vertices on the line’s upper envelope.

**Proof.** This is a direct consequence of the above lemmas. \( \square \)

**5. Line envelope computation**

To compute the uncertainty envelope of an LPGUM line, we use a sweep algorithm. We sweep the values of \( \alpha \) from a small value “before” any event, to a large value “after” the last event. As the values of \( \alpha \) change, the zonotope \( \Omega(\alpha) \) slides along the line, tracing out the line’s envelope (Fig. 2b). The sweep stops at events. The algorithm iterates over all the events, tracing out the uncertainty envelope segments, and finding new events. When the event queue is empty, the segments are combined.

To find an initial value for \( \alpha \), we compute all flip events, and add them to the event queue. We set \( \alpha \) at a value smaller than the smallest value found. We then repetitively compute the values of switch events among all neighboring lines, and set \( \alpha \) at a value smaller than the smallest value found. This is repeated until there is no need to change the value of \( \alpha \). The same procedure is now followed for twist events. Finding the initial value will take \( O(k^2) \) time, as there are no more than \( k \) flip events and \( k^2 \) switch events, and, in the absence of topological events, there are no more than \( k \) twist events.

For flip events, the value of the parameter corresponding to the flipped line in the cone diagram is negated for all vertices of the point’s envelope. For switch events, the values of the parameters corresponding to the lines involved are negated for the two vertices corresponding to the cones bound by both lines, and the values of new switch events are computed for the new neighboring lines in the cone diagram. For twist events, no updating action is necessary. For every type of event, we sweep the zonotope, from the previous value of \( \alpha \) to the current one.

We collect the segments of the boundary of all the sweeps, in a set called the set of segments. Switch events cause new lines of the cone diagram to become neighbors. Thus, we solve Eq. (4) for new events, and add them to the queue. Once the event has been handled, all parts of the envelope that are covered by the line, at values smaller than \( \alpha \), will have been added to the set of segments, and all new switch events will have been found.

When the event queue is empty, we compute the line’s envelope by finding the upper and lower envelope of the segments, in the set of segments.

**Theorem 3.** The uncertainty envelope of an LPGUM line, \( l(q) \), can be computed in \( O(k^2 \log k) \) time and \( O(k^2) \) space.

**Proof (Outline).** That the algorithm is correct is confirmed by Lemma 7. To prove the time and space complexity, we show that the overall number of events handled by the algorithm is \( O(k^2) \), there being no more than \( O(k) \) in the queue at one time. Maintaining the queue takes \( O(\log k) \) for every event. Finally, we show, based on [16], that the envelope can be computed in \( O(k^2 \log k) \) time. \( \square \)

**6. Uncertain circle**

Let \( o(q), u(q), v(q) \) and \( w(q) \) be four LPGUM points, and let \( d(q) \) be an LPGUM dimension, all defined over a parametric uncertainty model \( (q, \bar{q}, \Delta) \). A circle can be defined in one of four ways: (a) center-radius: a center point \( o(q) \) and radius \( d(q) \); (b) center-point: a center point \( o(q) \) and a circumference point \( u(q); (c) antipodal-points: two antipodal circumference points \( u(q), v(q) \), and (d) three-points: three circumference points \( u(q), v(q), w(q) \). In the nominal case, all four representations are equivalent. In the LPGUM model, the first three representations are cases of the center-vector circle defined below, while the last one is different.

**Definition 10.** Uncertain center-vector circle — let \( o(q) \) be the center and \( r(q) \) be the radius, where \( o(q) \) and \( r(q) \) are an LPGUM point and vector, both defined over the same parametric uncertainty model \( (q, \bar{q}, \Delta) \). The center-vector circle is defined as \( c(q) = o(q) + R(\theta)r(q) \), where \( R(\theta) \) is a \( 2 \times 2 \) rotation matrix.

The uncertainty zone of a center-vector LPGUM circle, \( c(q) \), is the set of all circles, \( c(q) \), for instances of \( q \in \Delta \):

\[
Z(c(q)) = \{ c' \mid c' = o + A_\theta q + R(\theta)(r + A_\theta q) \}, \quad q \in \Delta, \quad \theta \in [0, 2\pi].
\]

**Lemma 11.** The center-radius, center-point and antipodal-points LPGUM representations of a circle are special cases of the center-vector representation.

To prove the lemma, we describe a construction of a center-point circle from the center-radius, center-point and antipodal-points representations, and show how a center-vector circle can be transformed into any one of the three representations. \( \square \)

The three-point representation is not equivalent to the point and vector representation. To show this consider the following three points, \( a(q) = [0, -10]^T + [0, 0]^T q, b(q) = [10, 0] + [0, 1]^T q, c(q) = [0, 10] + [0, 0]^T q \), defined over a one parameter uncertainty model \( (q, \bar{q}, \Delta) \), where \( \Delta = [-1, 1] \). When \( q = 0 \), the center should attain the value, \( [0, 0]^T \), and it should attain the value, \( [\bar{q}, 0] \), for both \( q = 1 \) and for \( q = -1 \). It is impossible to create an LPGUM point with such properties.

**Definition 12.** Uncertain three-point circle — let \( u(q), v(q) \) and \( w(q) \) be three LPGUM points defined over the same parametric uncertainty model \( (q, \bar{q}, \Delta) \). The three-point LPGUM circle is:

\[
c(q) = circle(u(q), v(q), w(q)), \quad q \in \Delta.
\]

The three-point LPGUM circle’s uncertainty zone is the set of all circles, \( c(q) \), for instances of parameter vector \( q \):

\[
Z(c(q)) = \{ c' \mid c' = circle(u(q), v(q), w(q)) \}, \quad q \in \Delta.
\]

**7. Center-vector circle envelope**

Let \( c(q) = o(q) + R(\theta)r(q) \) be an LPGUM circle. The uncertainty zone of a center-vector circle, Eq. (7), is an annulus-like area, enclosed by an outer envelope and a possibly empty inner envelope (Fig. 4). The outer envelope consists of line segments and circular arcs; the inner envelope consists of circular arcs. The outer envelope is the union of all circle instances; the inner envelope is the intersection of all circle instances, and is thus convex. The geometric complexity of the envelopes is \( O(k^2) \).

**7.1. Center-vector circle events**

To compute the circle’s uncertainty zone, we follow the same method as for an LPGUM line 5. Hence, we have similar events.
Every value of $\theta$ yields an LPGUM point, $c_\theta(q)$. The nominal location of $c_\theta(q)$ is $\bar{c}_\theta = \bar{o} + R(\theta)\bar{r}$, and its sensitivity matrix is $A_\theta = A_\bar{\theta} + R(\theta)A_r$. Every column of the uncertainty sensitivity matrix, $(A_\theta)_i$, is the sum of $(A_{\bar{\theta}})_i$ and $R(\theta)(A_r)_i$. Thus, every value of $\theta$, $c_\theta(q)$ yields a different cone diagram. We now characterize the circle's events.

Definition 13. Switch event — let $c(q)$ be a center-vector LPGUM circle. A switch event occurs at a value of $\theta$, at which two lines of the cone diagram of $c_\theta(q)$ coincide:

$$\begin{vmatrix} (A_\theta)_{1i} + (R(\theta)(A_r)_{1i}) & (A_\theta)_{2i} + (R(\theta)(A_r)_{2i}) \\ (A_\theta)_{1j} + (R(\theta)(A_r)_{1j}) & (A_\theta)_{2j} + (R(\theta)(A_r)_{2j}) \end{vmatrix} = 0$$

that is, when $a \cos(\theta) + b \sin(\theta) + c = 0$ where

$$a = (A_{\bar{\theta}})_{1i} (A_r)_{2i} + (A_{\bar{\theta}})_{1j} (A_r)_{2j}$$
$$b = (A_{\bar{\theta}})_{1i} (A_r)_{1i} + (A_{\bar{\theta}})_{1j} (A_r)_{1j}$$
$$c = (A_{\bar{\theta}})_{1i} (A_r)_{1j} + (A_{\bar{\theta}})_{1j} (A_r)_{1i}$$

The resulting trigonometric equation in one unknown can have zero, one, or two solutions in the range $[0, 2\pi]$. When $a = b = c = 0$, there are infinitely many solutions, but no event occurs, as the cone diagram topology remains the same.

Definition 14. Flip event — let $c(q)$ be a center-vector LPGUM circle. A flip event occurs at values of $\theta$ for which column vector $(A_{\bar{\theta}})_i$ is zero.

This occurs when $\| (A_{\bar{\theta}})_i \| = \| (A_{\bar{\theta}})_j \|$ at $\theta = \pi - \alpha$, where $\alpha \in (-\pi, \pi)$ is the angle between $(A_{\bar{\theta}})_i$ and $(A_{\bar{\theta}})_j$.

Definition 15. Twist events — let $c(q)$ be a center-vector LPGUM circle. A twist event is a pair of values of $\theta$, at which no topological event occurs, and the part of $c_\theta(q)$ supporting the envelope changes. These are values of $\theta$ for which

$$c(q^{(i)}) = c(q^{(j)})$$

where $q^{(i)}, q^{(j)} \in \Delta$ are parameterizations of neighboring vertices on thezonotope of $c_\theta(q)$.

Lemma 16. Let $c(q)$ be a center-vector LPGUM circle. The outer boundary of its uncertainty zone consists of circular arcs and straight line segments and the inner boundary, if it exists, consists of circular arcs only.

Proof (Outline). We show that, in sweeping the edges of thezonotopes between event points on one zonotope are connected to points on the next one by circular arcs, in such a way that the outer boundary has the desired properties.

The inner boundary of the uncertainty zone is the outer boundary of the intersection of all the discs defined by $c(q)$ for any $q \in \Delta$. Thus it consists of circular arcs only. □

Theorem 4. Let $c(q)$ be a center-vector LPGUM circle. The complexity of the uncertainty zone envelope of $c(q)$ is $O(k^2)$.

Proof (Outline). As with the line, we show that segments are added to the boundary, only at events, and that there is a constant number of edges added at every one of the $O(k^2)$ events. The cell complexity in the resulting arrangement is shown to be $O(\lambda_d(n)) = O(n^{2\alpha(10)})$, where $\lambda_d(n)$ is the length of the Davenport–Schinzel sequence, $DS(n, s)$ [17]. Thus, the envelope’s complexity is $O(k^2 \cdot 2^{\alpha(10)}) = O(k^2 \log^2 k)$. □

8. Circle uncertainty zone computation

8.1. Independent uncertainties

When the center point and vector uncertainties are independent, the circle uncertainty zone is computed from the center point zonotope and the annulus centered on the origin. The annulus, $a$, is defined by $r_{\min} = \min \| v(q) \|$ and $r_{\max} = \max q \| v(q) \|$ for $q \in \Delta$. The uncertainty zone is the Minkowski sum of the zonotope of $o(q)$, and the annulus $a$. The outer envelope is computed in $O(k)$ time. When the diameter of the zonotope of $o(q)$ is greater than $2r_{\min}$, the inner envelope is empty. Otherwise, it contains $\bar{o}$ and is computed in $O(k \log^2 k)$ [17], which dominates the computation time complexity.

8.2. Dependent uncertainties

When the center point and vector are dependent, we use a sweep algorithm similar to that used in computing the line. We perform an angular sweep of $\theta$, starting at $\theta = 0$ in Eq. (7). Each value of $\theta$ yields a zonotope, $c_\theta(q)$. As $\theta$ changes, $c_\theta(q)$ traces out the circle envelope. The values of $\theta$ for which the topology of the cone diagram of $c_\theta(q)$ changes correspond to events. Starting at the initial zonotope $c_\theta(q)$, we sweep the zonotope from its previous location to the current one, and compute new events. When the event queue is empty, we sweep the last zonotope to $c_\theta(q)$, and combine the parts to obtain the envelope.

To move the zonotope from one event to the next, we compute the circle, $c(q)$, for every point on the boundary of the first zonotope, and connect the resulting zonotopes with arc segments on the circle between the event angles. To compute this zonotope edge sweep, we compute the circular arc, only for vertices of the zonotope, $c_\theta(q)$. When the tangent to the two circles in the interval is valid, it is added.

If the swept area contains a tangent to the circular segments, the outer part of that area consists of the tangent segment, and the two arcs leading from the end of the tangent to the end of the zonotopes’ edges. The rest of the boundary is the inner part. If there is no tangent, the outer part is the arc that bounds the area, so that the swept area is on the inner side of the arc and the inner part is the other arc. The outer and inner parts of the swept areas are collected in separate sets. When no more events are left, the outer and inner envelopes are computed from their corresponding sets.

The algorithm for computing the circle’s envelope is identical to that of the line envelope, with the following two differences. First, the sweep from one event to the next is adapted for the circle as described above. Second, we do not compute or handle twist events, as they are handled implicitly by the addition of the tangent segment in the sweep step of the algorithm. The algorithm time complexity is dominated by the computation of the outer and inner cells of the two arrangements. As there are $O(k^2)$ events and every zonotope edge is swept at every event, each of the sets has, at most, $O(k^2)$ arcs. The arrangement can be computed with a standard sweepline algorithm in $O(k^2 \log k)$ time. When a point in the interior of each cell to be computed is known, the time is reduced to $O(\lambda_d(n) \log^2(n))$. Since $n = k^2$, $O(\lambda_d(k^2) \log^2(k^2)) \approx O(k^2 \log^2(k^2)) = O(k^4 \log^2 k)$. □
9. Conclusion

Classical computational geometry algorithms handle geometric constructs whose shapes and locations are exact. However, many real-world applications require the modeling of objects with geometric uncertainties. Existing geometric uncertainty models cannot handle dependencies among objects. This results in the overestimation of errors. We have developed the Linear Parametric Geometric Uncertainty Model, a general, computationally efficient, worst-case, linear approximation of geometric uncertainty that supports dependencies among uncertainties.

We define and describe the properties of a line and a circle under LPGUM, and shown that the complexity of a line’s uncertainty envelope is $O(k)$, and that of a center-vector circle in $O(k^2)$. We have also presented efficient algorithms for computing them.

The space necessary to store these basic entities is relatively modest, and the running time for computing them is low. Therefore, we believe that this model could be used to handle more complex problems with coupled geometric uncertainties, with polynomial run times. In fact our current work addresses relative positioning problems involving uncertain points and uncertain range queries [14,18].

In the future, we plan to consider computing the convex hull of uncertain points, the topological stability of an uncertain point set, and Voronoi diagrams of uncertain points. We further plan to explore the properties of more complex geometric entities, such as conics and splines, as well as geometric objects in 3D, i.e. points, lines and curves in 3D, as well as surfaces. We plan to apply some of our algorithms in mechanical engineering applications, distributed sensor localization in wireless sensor networks, and trajectory planning in neurosurgery [19].

References