*International Journal of Geographical Information Science*
Vol. 00, No. 00, Month 200x, 1–26

# An analytic solution to the alibi query in the space-time prisms model for moving object data

Bart Kuijpers, Rafael Grimson and Walied Othman
Theoretical Computer Science Group
Hasselt University & Transnational University of Limburg, Belgium

(*August 2008*)

Moving objects produce trajectories, which are stored in databases by means of finite samples of time-stamped locations. When also speed limitations in these sample points are known, space-time prisms (also called beads) (Egenhofer 2003, Miller 2005, Pfoser and Jensen 1999) can be used to model the uncertainty about an object's location in between sample points. In this setting, a query of particular interest, that has been studied in the literature of geographic information systems (GIS), is the *alibi query*. This boolean query asks whether two moving objects can have physically met. This adds up to deciding whether the chains of space-time prisms (also called necklaces of beads) of these objects intersect. This problem can be reduced to deciding whether two space-time prisms intersect.

The alibi query can be seen as a constraint database query. In the constraint database model, spatial and spatio-temporal data are stored by boolean combinations of polynomial equalities and inequalities over the real numbers. The relational calculus augmented with polynomial constraints is the standard first-order query language for constraint databases and the alibi query can be expressed in it. The evaluation of the alibi query in the constraint database model relies on the elimination of a block of three existential quantifiers. Implementations of general purpose elimination algorithms, such as provided by QEPCAD, Redlog and Mathematica, are, for practical purposes, too slow in answering the alibi query for two specific space-time prisms. These software packages fail completely to answer the alibi query in the parametric case (that is, when it is formulated in terms of parameters representing the sample points and speed constraints).

The main contribution of this paper is an analytical solution to the parametric alibi query, which can be used to answer the alibi query on two specific space-time prisms in constant time (a matter of milliseconds in our implementation). It solves the alibi query for chains of space-time prisms in time proportional to the sum of the lengths of the chains. To back this claim up we implemented our method in MATHEMATICA alongside the traditional quantifier elimination method. Our solutions we propose are based on geometric argumentation and they illustrate the fact that some practical problems require creative solutions, where at least in theory, existing systems could provide a solution.

*Keywords:* uncertainty; space-time prism; alibi query; spatio-temporal; constraint databases; moving objects

## 1 Introduction and summary

The research on spatial databases, which started in the 1980s from work in geographic information systems, was extended in the second half of the 1990s to deal with spatio-temporal data. In this field, one particular line of research, concentrates on *moving object databases* (MODs) (Güting and Schneider 2005, Wolfson 2002), a field in which several data models and query languages have been proposed to deal with moving objects whose position is recorded at discrete moments in time. Some of these models are geared towards handling uncertainty that may come from various sources (measurements of locations, interpolation, ...) and several query formalisms have been proposed (Su et al. 2001, Geerts 2004, Kuijpers and Othman 2007). For an overview of models and techniques for MODs, we refer to the book by Güting and Schneider (Güting and Schneider 2005).

In this paper, we focus on the trajectories that are produced by moving objects and which are stored in a database as a collection of tuples $(\mathsf{t}_i, \mathsf{x}_i, \mathsf{y}_i)$, $i = 0, ..., N$, together with an object identifier. That is, trajectories are given as a finite sample of time-stamped locations in the plane. These samples may have been obtained by GPS-measurements or from other location aware devices. To reconstruct the trajectory of a moving object from such a sample, typically, linear interpolation is used. This model assumes that the object moves at constant minimal speed between the sample points and it does not cover uncertainty of the object's location.

One particular model for the management of the uncertainty of the moving object's position in between sample points is provided by the *space-time prism* model. Space-time prisms are also referred to as *beads* by some authors (Egenhofer 2003, Pfoser and Jensen 1999). In this model, it is assumed that besides the time-stamped locations of the object also some background knowledge, in particular a (e.g., physically or law imposed) speed limitation $\mathsf{v}_i$ at location $(\mathsf{x}_i, \mathsf{y}_i)$ is known. The space-time prism between two consecutive sample points is defined as the collection of time-space points where the moving objects could have passed, given the speed limitation (see Figure 1 for an illustration). The chain of space-time prisms connecting consecutive trajectory sample points is called a *lifeline necklace* (Egenhofer 2003). Whereas space-time prisms were already conceptually known in the time geography of Hägerstrand in the 1970s (Hägerstrand 1970), they were introduced in the area of GIS by Pfoser (Pfoser and Jensen 1999) and later studied by Egenhofer and Hornsby (Egenhofer 2003, Hornsby and Egenhofer 2002), and Miller (Miller 2005), and in a query language context by the present authors (Kuijpers and Othman 2007). Earlier, Wolfson proposed a cylinder model to deal with uncertainty (Wolfson 2002), but space-time prisms occupy only one third of space and therefore are a more efficient model for managing uncertainty, reducing the uncertainty of an objects location by two thirds.

In this setting, a query of particular interest that has been studied, mainly by Egenhofer and Hornsby (Egenhofer 2003, Hornsby and Egenhofer 2002), is the *alibi query*. This boolean query asks whether two moving objects, that are given by samples of time-space points and speed limitations, could have physically met. This question adds up to deciding whether the necklaces of space-time prisms of these moving objects intersect or not. This problem can be considered solved in practice, when we can efficiently decide whether two space-time prisms intersect.

Although approximate solutions to this problem have been proposed (Egenhofer 2003), also an exact solution is possible. We show that the alibi query can be formulated in the constraint database model by means of a first-order constraint database query (Kuijpers and Othman 2007, Paredaens et al. 2000). It is well-known that first-order constraint queries can be effectively evaluated and there exists implementations of quantifier-elimination algorithms for first-order logic over the real numbers that can be used to evaluate queries (Paredaens et al. 2000). Experiments with software packages such as QEPCAD (Hong 1990) and Mathematica (Wolfram 2007) on a variety of space-time prisms show that deciding if two concrete space-time prisms intersect can be computed on average in 2 minutes (running Windows XP Pro, SP2, with a Intel Pentium M, 1.73GHz, 1GB RAM). This means that evaluating the alibi query on the lifeline necklaces of two moving objects that each consist of 100 beads would take, if we test intersection of space-time prisms in the two necklaces pairwise, around $100 \times 100 \times 2$ minutes, which is almost two weeks. If we would first check whether the time domains of the space-time prisms in the two necklaces overlap, we could reduce the computation time to $(100 + 100) \times 2$ minutes, or almost 7 hours. Clearly, both amounts of time are unacceptable from a practical point of view.

Another solution within the range of constraint databases is to find a formula, in which the apexes and limit speeds of two space-time prisms appear as *parameters*, that parametrically expresses that two beads intersect. We call this problem the *parametric alibi query*. A quantifier-free formula for this parametric version could, in theory, also be obtained by eliminating one block of three existential quantifiers using existing quantifier-elimination software packages. We have attempted this approach using Mathematica and QEPCAD, but after several days of running (with the above processor), we have interrupted the computation, without successful outcome. It is known that these implementations fail on complicated, higher-dimensional problems. The benefit of having a quantifier-free first-order formula that expresses that two beads intersect is that the alibi query on two beads can be answered in constant time. The problem of deciding whether two lifeline necklaces intersect can then be done in time proportional to the sum of the lengths of the two necklaces of beads (if we first check if the time domains of the prisms overlap).

The main contribution of this paper is the description of an analytic solution to the alibi query. We give a quantifier-free formula, that contains square roots, however, and that expresses the (non-)emptiness of the intersection of two parametrically given space-time prisms. Although, in a strict sense, this formula cannot be seen as quantifier-free first-order formula (due to the roots), it still gives the above mentioned complexity benefits. At the basis of our solution is a geometric theorem that describes three exclusive cases in which space-time prisms can intersect. These three cases can then be transformed into an analytic solution that can be used to answer the alibi query on the lifeline necklaces consisting of 100 space-time prisms each in less than a minute. This provides a practical solution to the alibi query.

To back up our claim that the execution time of our method requires milliseconds or less we implemented this in MATHEMATICA and compared it to using traditional quantifier elimination to decide this query. We have included this implementation in the appendix and used it to perform numerous experiments which only confirm our claims.

This paper is organized as follows. In Section 2, we describe a model for trajectory (or moving object) databases with uncertainty using beads. In Section 3, we discuss the alibi query. An analytic solution to this query is given in Section 5.

## 2 A model for moving object data with uncertainty

In this paper, we consider moving objects in the two-dimensional $(x, y)$-space $\mathbf{R}^2$ and describe their movement in the $(t, x, y)$-space $\mathbf{R} \times \mathbf{R}^2$, where $t$ is time (we denote the set of the real numbers by $\mathbf{R}$).

In this section, we define trajectories, trajectory samples, space-time prisms and trajectory (sample) databases. Although it is more traditional to speak about moving object databases, we use the term trajectory databases to emphasize that we manage the trajectories produced by moving objects.

### 2.1 *Trajectories and trajectory samples*

Moving objects, which we assume to be points, produce a special kind of curves, which are parameterized by time and which we call *trajectories*.

**Definition 2.1** A *trajectory* $T$ is the graph of a mapping $I \subseteq \mathbf{R} \to \mathbf{R}^2 : t \mapsto \alpha(t) = (\alpha_x(t), \alpha_y(t))$, i.e.,

$$T = \{(t, \alpha_x(t), \alpha_y(t)) \in \mathbf{R} \times \mathbf{R}^2 \mid t \in I\},$$

where $I$ is the *time domain* of $T$. □

In practice, trajectories are only known at discrete moments in time. This partial knowledge of trajectories is formalized in the following definition. If we want to stress that some $t, x, y$-values (or other values) are constants, we will use sans serif characters.

**Definition 2.2** A *trajectory sample* is a finite set of time-space points $\{(\mathsf{t}_0, \mathsf{x}_0, \mathsf{y}_0), (\mathsf{t}_1, \mathsf{x}_1, \mathsf{y}_1), ..., (\mathsf{t}_N, \mathsf{x}_N, \mathsf{y}_N)\}$, on which the order on time, $\mathsf{t}_0 < \mathsf{t}_1 < \cdots < \mathsf{t}_N$, induces a natural order. □

For practical purposes, we may assume that the $(\mathsf{t}_i, \mathsf{x}_i, \mathsf{y}_i)$-tuples of a trajectory sample contain rational values.

A trajectory $T$, which contains a trajectory sample $\{(\mathsf{t}_0, \mathsf{x}_0, \mathsf{y}_0), (\mathsf{t}_1, \mathsf{x}_1, \mathsf{y}_1), ..., (\mathsf{t}_N, \mathsf{x}_N, \mathsf{y}_N)\}$, i.e., $(\mathsf{t}_i, \alpha_x(\mathsf{t}_i), \alpha_y(\mathsf{t}_i)) = (\mathsf{t}_i, \mathsf{x}_i, \mathsf{y}_i)$ for $i = 0, ..., N$, is called a *geospatial lifeline* for this trajectory sample (Egenhofer 2003). A common example of a lifeline, is the reconstruction of a trajectory from a trajectory samples by linear interpolation (Güting and Schneider 2005).

### 2.2 *Modeling uncertainty with space-time prisms*

Often, in practical applications, more is known about trajectories than merely some sample points $(\mathsf{t}_i, \mathsf{x}_i, \mathsf{y}_i)$. For instance, background knowledge like a physically or law imposed speed limitation $\mathsf{v}_i$ at location $(\mathsf{x}_i, \mathsf{y}_i)$ might be available. Such a speed limit might even depend on $\mathsf{t}_i$. The speed limits that hold between two consecutive sample points can be used to model the uncertainty of a moving object's location between sample points.

More specifically, we know that at a time $t$, $\mathsf{t}_i \leq t \leq \mathsf{t}_{i+1}$, the object's distance to $(\mathsf{x}_i, \mathsf{y}_i)$ is at most $\mathsf{v}_i(t - \mathsf{t}_i)$ and its distance to $(\mathsf{x}_{i+1}, \mathsf{y}_{i+1})$ is at most $\mathsf{v}_i(\mathsf{t}_{i+1} - t)$. The spatial location of the object is therefore somewhere in the intersection of the disc with center $(\mathsf{x}_i, \mathsf{y}_i)$ and radius $\mathsf{v}_i(t - \mathsf{t}_i)$ and the disc with center $(\mathsf{x}_{i+1}, \mathsf{y}_{i+1})$ and radius $\mathsf{v}_i(\mathsf{t}_{i+1} - t)$. The geometric location of these points is referred to as a *space-time prism* (Pfoser and Jensen 1999, Egenhofer 2003) and defined, for general points $p = (t_p, x_p, y_p)$ and $q = (t_q, x_q, y_q)$ and speed limit $v_{\max}$ as follows.

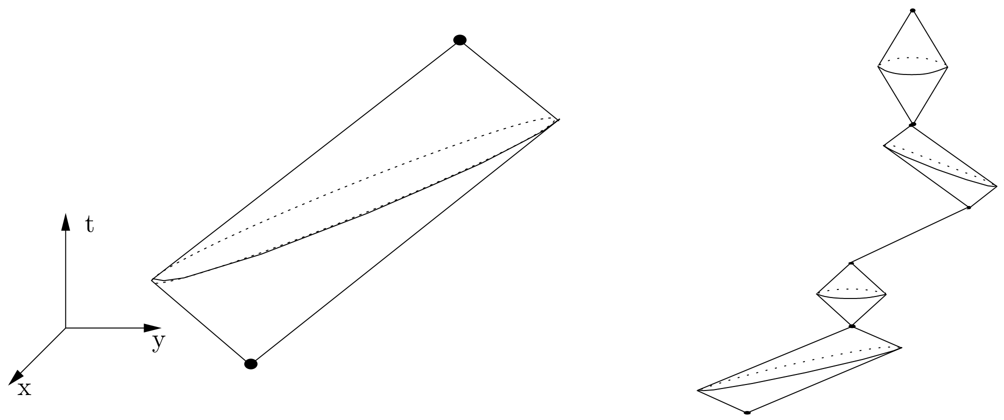


Figure 1. A space-time prism and a lifeline necklace.

**Definition 2.3** The *space-time prism* with origin $p = (t_p, x_p, y_p)$, destination $q = (t_q, x_q, y_q)$, with $t_p \leq t_q$, and maximal speed $v_{\max} \geq 0$ is the set of all points $(t, x, y) \in \mathbf{R} \times \mathbf{R}^2$ that satisfy the following constraint formula[1]

$$\Psi_{\mathcal{P}}(t, x, y, t_p, x_p, y_p, t_q, x_q, y_q, v_{\max}) := (x - x_p)^2 + (y - y_p)^2 \leq (t - t_p)^2 v_{\max}^2$$
$$\wedge\ (x - x_q)^2 + (y - y_q)^2 \leq (t_q - t)^2 v_{\max}^2\ \wedge\ t_p \leq t \leq t_q.$$

We denote this set by $\mathcal{P}(p, q, v_{\max})$ or $\mathcal{P}(t_p, x_p, y_p, t_q, x_q, y_q, v_{\max})$. □

In the formula $\Psi_{\mathcal{P}}(t, x, y, t_p, x_p, y_p, t_q, x_q, y_q, v_{\max})$, we consider $t_p, x_p, y_p, t_q, x_q, y_q, v_{\max}$ to be parameters, whereas $t, x, y$ are considered variables defining the subset of $\mathbf{R} \times \mathbf{R}^2$.

Figure 2 illustrates the notion of space-time prism in time-space. Whereas a continuous curve connecting the sample points of a trajectory sample was called a geospatial lifeline, a chain of space-time prisms connecting succeeding trajectory sample points is called a *lifeline necklace* (Egenhofer 2003).

## 2.3 *Trajectory databases*

We assume the existence of an infinite set $\mathsf{Labels} = \{\mathsf{a}, \mathsf{b}, ..., \mathsf{a}_1, \mathsf{b}_1, ..., \mathsf{a}_2, \mathsf{b}_2, ...\}$ of *trajectory labels*, that serve to identify individual trajectory samples. We now define the notion of trajectory database.

**Definition 2.4** A *trajectory (sample) database* is a finite set of tuples $(\mathsf{a}_i, \mathsf{t}_{i,j}, \mathsf{x}_{i,j}, \mathsf{y}_{i,j}, \mathsf{v}_{i,j})$, with $i = 1, ..., r$ and $j = 0, ..., N_i$, such that $\mathsf{a}_i \in \mathsf{Labels}$ cannot appear twice in combination with the same $t$-value, such that $\{(\mathsf{t}_{i,0}, \mathsf{x}_{i,0}, \mathsf{y}_{i,0}), (\mathsf{t}_{i,1}, \mathsf{x}_{i,1}, \mathsf{y}_{i,1}), ..., (\mathsf{t}_{i,N_i}, \mathsf{x}_{i,N_i}, \mathsf{y}_{i,N_i})\}$ is a trajectory sample for each $i = 1, ..., r$ and such that the $\mathsf{v}_{i,j} \geq 0$ for each $i = 1, ..., r$ and $j = 0, ..., N_i$. □

## 3 Trajectory queries and the alibi query

In this section, we define the notion of trajectory database query, we show how constraint database languages can be used to query trajectories and we define the alibi query and the parametric alibi query.

[1]Later on, this type of formula's will be refered to as $\mathsf{FO}(+, \times, <, 0, 1)$-formulas.

## 3.1 *Trajectory queries*

A *trajectory database query* has been defined as a partial computable function from trajectory databases to trajectory databases (Kuijpers and Othman 2007). Often, we are also interested in queries that express a property, i.e., in boolean queries. More formally, we can say that a *boolean trajectory database query* is a partial computable function from trajectory databases to $\{\mathsf{True}, \mathsf{False}\}$.

When we say that a function is computable, this is with respect to some fixed encoding of the trajectory databases (e.g., rational numbers are represented as pairs of natural numbers in bit representation).

## 3.2 *A constraint-based query language*

Several languages have been proposed to express queries on moving object data and trajectory databases (see (Güting and Schneider 2005) and references therein). One particular language for querying trajectory data, that was recently studied in detail by the present authors, is provided by the formalism of constraint databases. This query language is a first-order logic which extends first-order logic over the real numbers with a predicate $S$ to address the input trajectory database. We denote this logic by $\mathsf{FO}(+, \times, <, 0, 1, S)$ and define it as follows.

**Definition 3.1** The language $\mathsf{FO}(+, \times, <, 0, 1, S)$ is a two-sorted logic with *label variables* $a, b, c, ...$ (possibly with subscripts) that refer to trajectory labels and *real variables* $x, y, z, ..., v, ...$ (possibly with subscripts) that refer to real numbers. The atomic formulas of $\mathsf{FO}(+, \times, <, 0, 1, S)$ are

- $P(x_1, ..., x_n) > 0$, where $P$ is a polynomial with integer coefficients in the real variables $x_1, ..., x_n$;
- $a = b$; and
- $S(a, t, x, y, v)$ ($S$ is a 5-ary predicate).

The formulas of $\mathsf{FO}(+, \times, <, 0, 1, S)$ are built from the atomic formulas using the logical connectives $\wedge, \vee, \neg, ...$ and quantification over the two types of variables: $\exists x$, $\forall x$ and $\exists a$, $\forall a$. $\Box$

The label variables are assumed to range over the labels occurring in the input trajectory database and the real variables are assumed to range over $\mathbf{R}$. The formula $S(a, t, x, y, v)$ expresses that a tuple $(a, t, x, y, v)$ belongs to the input trajectory database. The interpretation of the other formulas is standard.

For example, the $\mathsf{FO}(+, \times, <, 0, 1, S)$-sentence

$$\exists a \exists b (\neg(a = b) \wedge \forall t \forall x \forall y \forall v S(a, t, x, y, v) \leftrightarrow S(b, t, x, y, v))$$

expresses the boolean trajectory query that says that there are two identical trajectories in the input database with different labels.

When we instantiate the free variables in a $\mathsf{FO}(+, \times, <, 0, 1, S)$-formula $\varphi(a, b, ..., t, x, y, ...)$ by concrete values $\mathsf{a}, \mathsf{b}, ..., \mathsf{t}, \mathsf{x}, \mathsf{y}, ...$ we write $\varphi[\mathsf{a}, \mathsf{b}, ..., \mathsf{t}, \mathsf{x}, \mathsf{y}, ...]$ for the formula we obtain.

## 3.3 *The alibi query*

The *alibi query* is the boolean query which asks whether two moving objects, say with labels $\mathsf{a}$ and $\mathsf{a}'$, that are available as samples in a trajectory database, can have physically met. Since the possible positions of these moving objects are, in between sample points, given by space-time prisms, the alibi query asks to decide if the two lifeline necklaces of $\mathsf{a}$ and $\mathsf{a}'$ intersect or not.

More concretely, if the trajectory $\mathsf{a}$ is given in the trajectory database by the tuples $(\mathsf{a}, \mathsf{t}_0, \mathsf{x}_0, \mathsf{y}_0, \mathsf{v}_0), ...., (\mathsf{a}, \mathsf{t}_N, \mathsf{x}_N, \mathsf{y}_N, \mathsf{v}_N)$ and the trajectory $\mathsf{a}'$ by the tuples $(\mathsf{a}', \mathsf{t}'_0, \mathsf{x}'_0, \mathsf{y}'_0, \mathsf{v}'_0), ...., (\mathsf{a}', \mathsf{t}'_M, \mathsf{x}'_M, \mathsf{y}'_M, \mathsf{v}'_M)$, then $\mathsf{a}$ has an alibi for not meeting $\mathsf{a}'$ if for all $i$, $0 \le i \le N-1$ and all $j$, $0 \le j \le M-1$,

$$\mathcal{P}(\mathsf{t}_i, \mathsf{x}_i, \mathsf{y}_i, \mathsf{t}_{i+1}, \mathsf{x}_{i+1}, \mathsf{y}_{i+1}, \mathsf{v}_i) \cap \mathcal{P}(\mathsf{t}'_j, \mathsf{x}'_j, \mathsf{y}'_j, \mathsf{t}'_{j+1}, \mathsf{x}'_{j+1}, \mathsf{y}'_{j+1}, \mathsf{v}'_j) = \emptyset. \qquad (\dagger)$$

We remark that the alibi query can be expressed by a formula in the logic $\mathsf{FO}(+, \times, <, 0, 1, S)$, which we now give. To start, we denote the subformula

$$S(a, \mathsf{t}_1, \mathsf{x}_1, \mathsf{y}_1, \mathsf{v}_1) \wedge S(a, t_2, x_2, y_2, v_2) \wedge \forall t_3 \forall x_3 \forall y_3 \forall v_3 (S(a, t_3, x_3, y_3, v_3) \rightarrow \neg(t_1 < t_3 \wedge t_3 < t_2)),$$

that expresses that $(t_1, x_1, y_1)$ and $(t_2, x_2, y_2)$ are consecutive sample points on the trajectory $a$ by $\sigma(a, t_1, x_1, y_1, v_1, t_2, x_2, y_2, v_2)$.

The alibi query on $\mathsf{a}$ and $\mathsf{a}'$ is then expressed as $\varphi_{\text{alibi}}[\mathsf{a}, \mathsf{a}'] =$

$$\neg \exists t_1 \exists x_1 \exists y_1 \exists v_1 \exists t_2 \exists x_2 \exists y_2 \exists v_2 \exists t'_1 \exists x'_1 \exists y'_1 \exists v'_1 \exists t'_2 \exists x'_2 \exists y'_2 \exists v'_2$$
$$(\sigma(\mathsf{a}, t_1, x_1, y_1, v_1, t_2, x_2, y_2, v_2) \wedge \sigma(\mathsf{a}', t'_1, x'_1, y'_1, v'_1, t'_2, x'_2, y'_2, v'_2) \wedge \ \exists t \exists x \exists y (t_1 \leq t \leq t_2 \wedge t'_1 \leq t \leq t'_2 \wedge$$
$$(x - x_1)^2 + (y - y_1)^2 \leq (t - t_1)^2 v_1^2 \wedge (x - x_2)^2 + (y - y_2)^2 \leq (t_2 - t)^2 v_1^2 \wedge$$
$$(x - x'_1)^2 + (y - y'_1)^2 \leq (t - t'_1)^2 v_1'^2 \wedge (x - x'_2)^2 + (y - y'_2)^2 \leq (t'_2 - t)^2 v_1'^2)).$$

It is well-known that $\mathsf{FO}(+, \times, <, 0, 1, S)$-expressible queries can be evaluated effectively on arbitrary trajectory database inputs (Paredaens et al. 2000, Kuijpers and Othman 2007). Briefly explained, this evaluation can be performed by (1) replacing the occurrences of $S(\mathsf{a}, t, x, y, v)$ by a disjunction describing all the sample points belonging to the trajectory sample $\mathsf{a}$; the same for $\mathsf{a}'$; and (2) eliminating all the quantifiers in the obtained formula. In concreto, using the notation from above, each occurrence of $S(\mathsf{a}, t, x, y, v)$ would be replaced in $\varphi_{\text{alibi}}[\mathsf{a}, \mathsf{a}']$ by $\bigvee_{i=0}^{N-1} (t = \mathsf{t}_i \wedge x = \mathsf{x}_i \wedge y = \mathsf{y}_i \wedge v = \mathsf{v}_i)$, and similar for $\mathsf{a}'$. This results in a (rather complicated) first-order formula over the reals $\tilde{\varphi}_{\text{alibi}}[\mathsf{a}, \mathsf{a}']$ in which the predicate $S$ does not occur any more. Since first-order logic over the reals admits the elimination of quantifiers (i.e., every formula can be equivalently expressed by a quantifier-free formula), we can decide the truth value of $\tilde{\varphi}_{\text{alibi}}[\mathsf{a}, \mathsf{a}']$ by eliminating all quantifiers from this expression. In this case, we have to eliminate one block of existential quantifiers.

We can however simplify the quantifier-elimination problem. It is easy to see, looking at (†) above, that $\neg \tilde{\varphi}_{\text{alibi}}[\mathsf{a}, \mathsf{a}']$ is equivalent to

$$\bigvee_{i=0}^{N-1} \bigvee_{j=0}^{M-1} \psi_{alibi}[\mathsf{t}_i, \mathsf{x}_i, \mathsf{y}_i, \mathsf{t}_{i+1}, \mathsf{x}_{i+1}, \mathsf{y}_{i+1}, \mathsf{v}_i, \mathsf{t}'_j, \mathsf{x}'_j, \mathsf{y}'_j, \mathsf{t}'_{j+1}, \mathsf{x}'_{j+1}, \mathsf{y}'_{j+1}, \mathsf{v}'_j],$$

where the restricted alibi-query formula $\psi_{alibi}(t_i, x_i, y_i, t_{i+1}, x_{i+1}, y_{i+1}, v_i, t'_j, x'_j, y'_j, t'_{j+1}, x'_{j+1}, y'_{j+1}, v'_j)$ abbreviates the formula

$$\exists t \exists x \exists y (t_i \leq t \leq t_{i+1} \wedge t'_j \leq t \leq t'_{j+1} \wedge (x - x_i)^2 + (y - y_i)^2 \leq (t - t_i)^2 v_i^2 \wedge$$
$$(x - x_{i+1})^2 + (y - y_{i+1})^2 \leq (t_{i+1} - t)^2 v_i^2 \wedge$$
$$(x - x'_j)^2 + (y - y'_j)^2 \leq (t - t'_j)^2 v_j'^2 \wedge (x - x'_{j+1})^2 + (y - y'_{j+1})^2 \leq (t'_{j+1} - t)^2 v_j'^2)$$

that expresses that two space-time prisms intersect.

So, the instantiated formula

$$\psi_{alibi}[\mathsf{t}_i, \mathsf{x}_i, \mathsf{y}_i, \mathsf{t}_{i+1}, \mathsf{x}_{i+1}, \mathsf{y}_{i+1}, \mathsf{v}_i, \mathsf{t}'_j, \mathsf{x}'_j, \mathsf{y}'_j, \mathsf{t}'_{j+1}, \mathsf{x}'_{j+1}, \mathsf{y}'_{j+1}, \mathsf{v}'_j]$$

expresses (†). To eliminate the existential block of quantifiers $(\exists t \exists x \exists y)$ from this expression, existing software-packages for quantifier elimination, such as QEPCAD (Hong 1990), Redlog (Sturm 2000) and Mathematica (Wolfram 2007) can be used. We experimented QEPCAD, Redlog and Mathematica to decide if several space-time prisms intersected. The latter two programs have a similar performance and they outperform QEPCAD. To give an idea of their performance, we give some results with Mathematica: the computation of $\psi_{alibi}[0, 0, 0, 1, 2, 2, \sqrt{8}, 0, 3, 3, 1, 2, 2, 2]$ took 6 seconds; that of $\psi_{alibi}[0, 0, 0, 1, 2, 2, \sqrt{8}, 0, 3, 4, 1, 2, 2, 2]$ took 209 seconds and the computation of $\psi_{alibi}[0, 0, 0, 1, -1, -1, 1, 0, 1, 1, 2, -1, 1, 2]$ took 613 seconds. Roughly speaking, our experiments show that, using Mathematica , this quantifier elimination can be computed on average in about 2 minutes (running Windows XP Pro, SP2, with a Intel Pentium M, 1.73GHz, 1GB RAM). This means that evaluating the alibi query on the lifeline necklaces of two moving objects that each consist of 100 space-time prisms would take around $100 \times 100 \times 2$ minutes, which is almost two weeks, when applied naively and at most $(100 + 100) \times 2$ minutes or a quarter day, when first the intersection of time-intervals is tested. Clearly, in both cases, such an amount of time is unacceptable.

There is a better solution, however, which we discuss next, that can decide if two space-time prisms intersect or not in a couple of milliseconds.

## 3.4 *The parametric alibi query*

The uninstantiated formula

$$\psi_{alibi}(t_i, x_i, y_i, t_{i+1}, x_{i+1}, y_{i+1}, v_i, t'_j, x'_j, y'_j, t'_{j+1}, x'_{j+1}, y'_{j+1}, v'_j)$$

can be viewed as a parametric version of the restricted alibi query, where the free variables are considered parameters. This formula contains three existential quantifiers and the existing software-packages for quantifier elimination could be used to obtain a quantifier-free formula $\tilde{\psi}_{alibi}(t_i, x_i, y_i, t_{i+1}, x_{i+1}, y_{i+1}, v_i, t'_j, x'_j, y'_j, t'_{j+1}, x'_{j+1}, y'_{j+1}, v'_j)$ that is equivalent to $\psi_{alibi}$. The formula $\tilde{\psi}_{alibi}$ could then be used to straightforwardly answer the alibi query in time linear in its size, which is independent of the size of the input and therefore constant. We have tried to eliminate the existential block of quantifiers $\exists t \exists x \exists y$ from $\psi_{alibi}$ using Mathematica, Redlog and QEPCAD. After some minutes of running, Redlog invokes QEPCAD. After several days of running QEPCAD on the configuration described above, we have interrupted the computation without result. Also Mathematica ran into problems without giving an answer. It is clear that eliminating a block of three existential quantifiers from a formula in 17 variables is beyond the existing quantifier-elimination implementations. Also, the instantiation of several parameters to adequately chosen constant values does not help to produce a solution. For instance, without loss of generality we can locate $(t_i, x_i, y_i)$ in the origin $(0, 0, 0)$ and locate the other apex of the first space-time prism above the $y$-axis, i.e., we can take $x_{i+1} = 0$. Furthermore, we can take $v_i = 1$ and $t_{i+1} = 1$. But Mathematica, Redlog and QEPCAD cannot also not cope with this simplified situation.

The main contribution of this paper is a the description of a quantifier-free formula equivalent to $\psi_{alibi}(t_i, x_i, y_i, t_{i+1}, x_{i+1}, y_{i+1}, v_i, t'_j, x'_j, y'_j, t'_{j+1}, x'_{j+1}, y'_{j+1}, v'_j)$. The solution we give is not a quantifier-free first-order formula in a strict sense, since it contains root expressions, but it can be easily turned into a quantifier-free first-order formula of similar length. It answers the alibi query on the lifeline necklaces of two moving objects that each consist of 100 space-time prisms in less than a minute. This description of this quantifier-free formula is the subject of the next section.

## 4 Preliminaries on the geometry of space-time prisms

Before we can give an analytic solution to the alibi query and prove its correctness, we need to introduce some terminology concerning space-time prisms.

### 4.1 *Geometric components of space-time prisms*

Various geometric properties of space-time prisms have already been described (Egenhofer 2003, Kuijpers and Othman 2007, Miller 2005). Here, we need some more definitions and notations to describe various components of a space-time prism. These components are illustrated in Figure 2. In this section, let $p = (t_p, x_p, y_p)$ and $q = (t_q, x_q, y_q)$ be two time-space points, with $t_p \leq t_q$ and let $v_{\max}$ be a positive real number.

The space-time prism $\mathcal{P}(p, q, v_{\max})$ is the intersection of two filled cones, given by the equations $(x - x_p)^2 + (y - y_p)^2 \leq (t - t_p)^2 v_{\max}^2 \ \wedge\ t_p \leq t$ and $(x - x_q)^2 + (y - y_q)^2 \leq (t_q - t)^2 v_{\max}^2 \ \wedge\ t \leq t_q$ respectively.

The border of its *bottom cone* is the set of all points $(t, x, y)$ that satisfy

$$\Psi_{\mathsf{C}^-}(t, x, y, t_p, x_p, y_p, v_{\max}) := (x - x_p)^2 + (y - y_p)^2 = (t - t_p)^2 v_{\max}^2 \ \wedge \ t_p \leq t$$

and is denoted by $\mathsf{C}^-(p, v_{\max})$ or $\mathsf{C}^-(t_p, x_p, y_p, v_{\max})$; and the border of its *upper cone* is the set of all points $(t, x, y)$ that satisfy

$$\Psi_{\mathsf{C}^+}(t, x, y, t_q, x_q, y_q, v_{\max}) := (x - x_q)^2 + (y - y_q)^2 = (t_q - t)^2 v_{\max}^2 \ \wedge \ t \leq t_q$$

and is denoted by $\mathsf{C}^+(q, v_{\max})$ or $\mathsf{C}^+(t_q, x_q, y_q, v_{\max})$.

The set of the two apexes of $\mathcal{P}(p, q, v_{\max})$ is denoted by $\tau\mathcal{P}(p, q, v_{\max})$, i.e., $\tau\mathcal{P}(p, q, v_{\max}) = \{p, q\}$.

We call the topological border of the space-time prism $\mathcal{P}(p, q, v_{\max})$ its *mantel* and denote it by $\partial\mathcal{P}(p, q, v_{\max})$. It can be easily verified that the mantel consists of the set of points $(t, x, y)$ that satisfy

$$\begin{aligned}
\Psi_\partial(t, x, y, t_p, x_p, y_p, t_q, x_q, y_q, v_{\max}) := {} & t_p \leq t \leq t_q \wedge \\
& \big(2x(x_p - x_q) + x_q^2 - x_p^2 + 2y(y_p - y_q) + y_q^2 - y_p^2 \leq v_{\max}^2 \left(2t(t_p - t_q) + t_q^2 - t_p^2\right) \wedge \\
& (x - x_p)^2 + (y - y_p)^2 = (t - t_p)^2 v_{\max}^2 \vee (x - x_q)^2 + (y - y_q)^2 = (t_q - t)^2 v_{\max}^2 \\
& \wedge\ 2x(x_p - x_q) + x_q^2 - x_p^2 + 2y(y_p - y_q) + y_q^2 - y_p^2 \geq v_{\max}^2 \left(2t(t_p - t_q) + t_q^2 - t_p^2\right)\big) .
\end{aligned}$$

The first conjunction describes the lower half of the mantel and the second conjunction describes the upper half of the mantel. The upper and lower half of the mantel are separated by a plane. The intersection of this plane with the space-time prism is an ellipse, and the border of this ellipse is what we will refer to as the *rim* of the space-time prism. We denote the rim of the space-time prism $\mathcal{P}(p, q, v_{\max})$ by $\rho\mathcal{P}(p, q, v_{\max})$ and remark that it is described by the formula

$$\begin{aligned}
\Psi_\rho(t, x, y, t_p, x_p, y_p, t_q, x_q, y_q, v_{\max}) := {} & (x - x_p)^2 + (y - y_p)^2 = (t - t_p)^2 v_{\max}^2 \wedge t_p \leq t \leq t_q \wedge \\
& 2x(x_p - x_q) + x_q^2 - x_p^2 + 2y(y_p - y_q) + y_q^2 - y_p^2 = v_{\max}^2 \left(2t(t_p - t_q) + t_q^2 - t_p^2\right) .
\end{aligned}$$

The plane in which the rim lies splits the space-time prism into an *upper-half space-time prism* and a *bottom-half space-time prism*. The *bottom-half space-time prism* is the set of all points $(t, x, y)$ that satisfy

$$\begin{aligned}
\Psi_{\mathcal{P}^-}(t, x, y, t_p, x_p, y_p, t_q, x_q, y_q, v_{\max}) := {} & (x - x_p)^2 + (y - y_p)^2 \leq (t - t_p)^2 v_{\max}^2 \wedge t_p \leq t \leq t_q \wedge \\
& 2x(x_p - x_q) + x_q^2 - x_p^2 + 2y(y_p - y_q) + y_q^2 - y_p^2 \leq v_{\max}^2 \left(2t(t_p - t_q) + t_q^2 - t_p^2\right)
\end{aligned}$$

and is denoted by $\mathcal{P}^-(t_p, x_p, y_p, t_q, x_q, y_q, v_{\max})$.

The upper space-time prism is the set of all points $(t, x, y)$ that satisfy

$$\begin{aligned}
\Psi_{\mathcal{P}^+}(t, x, y, t_p, x_p, y_p, t_q, x_q, y_q, v_{\max}) := {} & (x - x_q)^2 + (y - y_q)^2 \leq (t_q - t)^2 v_{\max}^2 \wedge t_p \leq t \leq t_q \wedge \\
& 2x(x_p - x_q) + x_q^2 - x_p^2 + 2y(y_p - y_q) + y_q^2 - y_p^2 \geq v_{\max}^2 \left(2t(t_p - t_q) + t_q^2 - t_p^2\right)
\end{aligned}$$

and is denoted by $\mathcal{P}^+(t_p, x_p, y_p, t_q, x_q, y_q, v_{\max})$.

### 4.2 *The intersection of two cones*

Let $\mathsf{C}^-(t_1, x_1, y_1, v_1)$ and $\mathsf{C}^-(t_2, x_2, y_2, v_2)$ be two bottom cones. A bottom cone, e.g., $\mathsf{C}^-(t_1, x_1, y_1, v_1)$, can be seen as a circle in 2-dimensional space $(x, y)$-space with center $(x_1, y_1)$ and linearly growing radius $(t - t_1)v_1$ as $t_1 \leq t$.

Let us assume that the apex of neither of these cones is inside the other cone, i.e., $(x_1 - x_2)^2 + (y_1 - y_2)^2 > (t_1 - t_2)^2 v_1^2 \vee t_1 < t_2$ and $(x_1 - x_2)^2 + (y_1 - y_2)^2 > (t_1 - t_2)^2 v_2^2 \vee t_2 < t_1$. This assumption implies that at

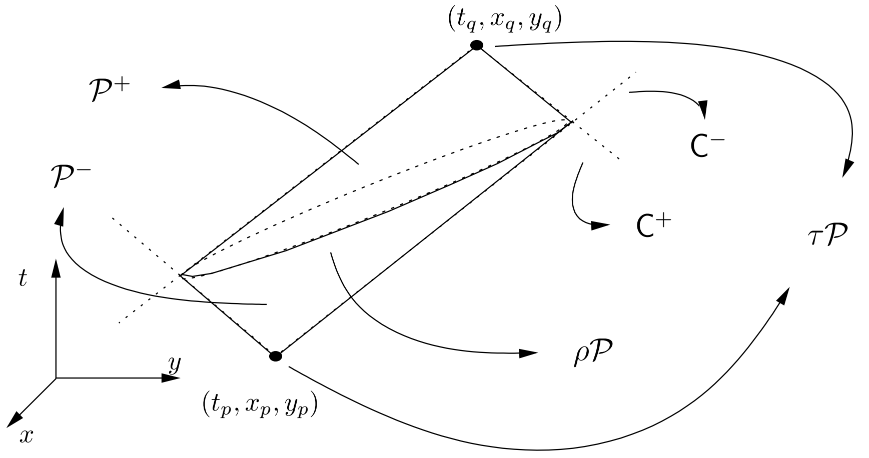


Figure 2. A dissection of the space-time prism $\mathcal{P}(t_p, x_p, y_p, t_q, x_q, y_q, v_{\max})$.

$t_1$ and $t_2$ neither radius is larger than or equal to the distance between the two cone centers. So, at first the two circles are disjoint and after growing for some time they intersect in one point. We call the first (in time) time-space point where the two circles touch in a single point, and thus for which the sum of the two radii is equal to the distance between the two centers the *initial contact* of the two cones $\mathsf{C}^-(t_1, x_1, y_1, v_1)$ and $\mathsf{C}^-(t_2, x_2, y_2, v_2)$. It is the unique point $(t, x, y)$ that satisfies the formula

$$\Psi_{IC^-}(t, x, y, t_1, x_1, y_1, v_1, t_2, x_2, y_2, v_2) := t_1 \leq t \wedge t_2 \leq t \wedge$$
$$(x - x_1)^2 + (y - y_1)^2 = (t - t_1)^2 v_1^2 \wedge (x - x_2)^2 + (y - y_2)^2 = (t - t_2)^2 v_2^2 \wedge$$
$$((t - t_1)v_1 + (t - t_2)v_2)^2 = (x_1 - x_2)^2 + (y_1 - y_2)^2.$$

The initial contact of two cones $\mathsf{C}^+(t_1, x_1, y_1, v_1)$ and $\mathsf{C}^+(t_2, x_2, y_2, v_2)$ is given by the formula $\Psi_{IC^+}(t, x, y, t_1, x_1, y_1, v_1, t_2, x_2, y_2, v_2)$ that we obtain from $\Psi_{IC^-}$ by replacing in $t_1 \leq t \wedge t_2 \leq t$ by $t \leq t_1 \wedge t \leq t_2$. We denote the singleton sets containing the initial contacts by $\mathsf{IC}(\mathsf{C}^-(t_1, x_1, y_1, v_1), \mathsf{C}^-(t_2, x_2, y_2, v_2))$ and $\mathsf{IC}(\mathsf{C}^+(t_1, x_1, y_1, v_1), \mathsf{C}^+(t_2, x_2, y_2, v_2))$.

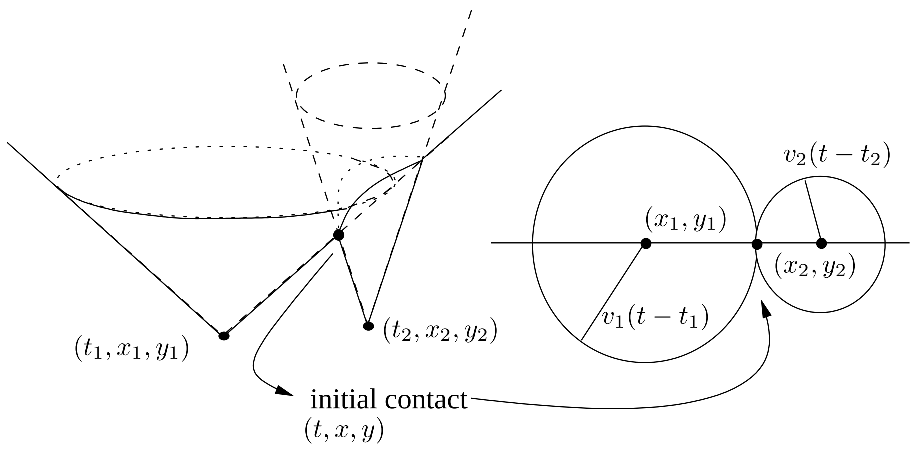


Figure 3. Intersecting cones and their initial contact (3-dimensional view on the left and 2-dimensional view on the right).

From the last equation in of the system in $\Psi_{IC^-}$ and $\Psi_{IC^+}$, we easily obtain $t = \frac{\sqrt{(x_1-x_2)^2+(y_1-y_2)^2}+t_1v_1+t_2v_2}{v_1+v_2}$. To compute the other two coordinates $(x, y)$ of the initial contact, we observe that for in the plane of this time value $t$, it is on the line segment bounded by $(x_1, y_1)$ and $(x_2, y_2)$ and that its distance from $(x_1, y_1)$ is $v_1(t - t_1)$ and its distance from $(x_1, y_1)$ is $v_2(t - t_2)$. We can conclude that the initial contact has $(t, x, y)$-coordinates given by the following system of equations

$$\begin{cases} t = \frac{\sqrt{(x_1-x_2)^2+(y_1-y_2)^2}+t_1v_1+t_2v_2}{v_1+v_2} \\ x = x_1 + v_1(t-t_1)\frac{x_2-x_1}{\sqrt{(x_2-x_1)^2+(y_2-y_1)^2}} \\ y = y_1 + v_1(t-t_1)\frac{y_2-y_1}{\sqrt{(x_2-x_1)^2+(y_2-y_1)^2}}. \end{cases}$$

This means that we can give more explicit descriptions to replace $\Psi_{IC^-}$ and $\Psi_{IC^+}$.

## 5 An analytic solution to the alibi query

In this section, we first describe our solution to the alibi query on a geometric level. Next, we prove its correctness and transform it into an analytic solution and finally we show how to construct a quantifier-free first-order formula out of the analytic solution.

### 5.1 *Preliminary geometric considerations*

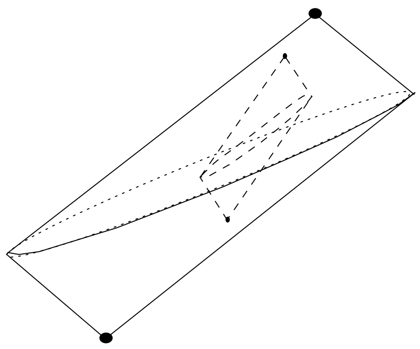


Figure 4. One space-time prism is contained in the other.

The solution we present is based on the observation that the two main cases of intersection (that do not exclude each other) are: (1) an apex of one space-time prism is in the other; and (2) the mantels of the space-time prisms intersect.

The inclusion of one space-time prism in the other, illustrated in Figure 4, is an example of the first case. It is clear that if no apex is contained in another space-time prism and we still assume that the space-time prisms intersect, than their mantels must intersect. We show this more formally in Lemma 5.1. In this second case, the idea is to find a special point (a witness point) that is easily computable and necessarily in the intersection.

Let us consider two space-time prisms with bottom cones $\mathsf{C}^-(t_1, x_1, y_1, v_1)$ and $\mathsf{C}^-(t_2, x_2, y_2, v_2)$ and let us assume that none of the apeces is inside the other cone. One special point is the point of initial contact $\mathsf{IC}(\mathsf{C}^-(t_1, x_1, y_1, v_1), \mathsf{C}^-(t_2, x_2, y_2, v_2))$. However, this point can not be guaranteed to be in the intersection if the mantels of the two space-time prisms intersect, as we will show in the following example. Consider two space-time prisms with bottom cones $\mathsf{C}^-(0, 0, 0, 1)$ and $\mathsf{C}^-(0, 2, 0, 1)$. The intersection is a hyperbola in the plane $x = 1$ with equation $t^2 - y^2 = 1$. The initial contact of the two bottom cones is the point $(1, 0, 1)$. To show that this point of initial contact does not need to be in the intersection of the two space-time prisms, the idea is to cut this point out of the intersection as follows. Suppose one space-time prism has apexes, $(0, 0, 0)$ and $(a, b, c)$ and speed 1. The plane in which its rim lies is given by $-2ax + a^2 - 2by + b^2 + 2ct - c^2 = 0$. This plane cuts the plane $\alpha$ given by the equality $x = 1$ in a line given by the equation $-2by + 2ct - 2a + a^2 - c^2 = 0$. Clearly, we can choose $(a, b, c)$ such that the line contains the points $\left(\frac{\sqrt{5}}{2}, 1, \frac{1}{2}\right)$ and $\left(\sqrt{2}, 1, 1\right)$. Everything below this line will be part of the first space-time

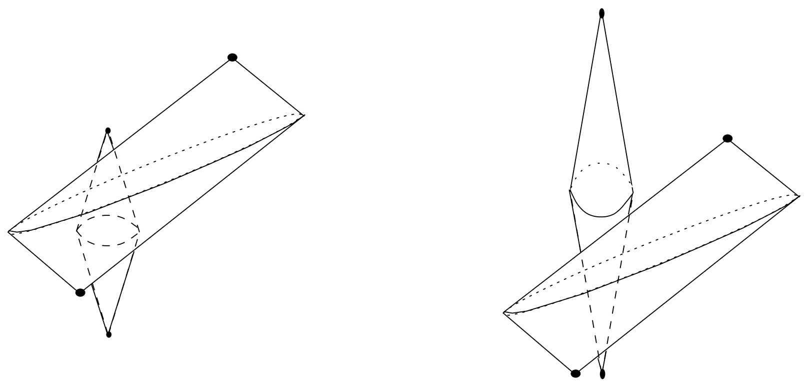
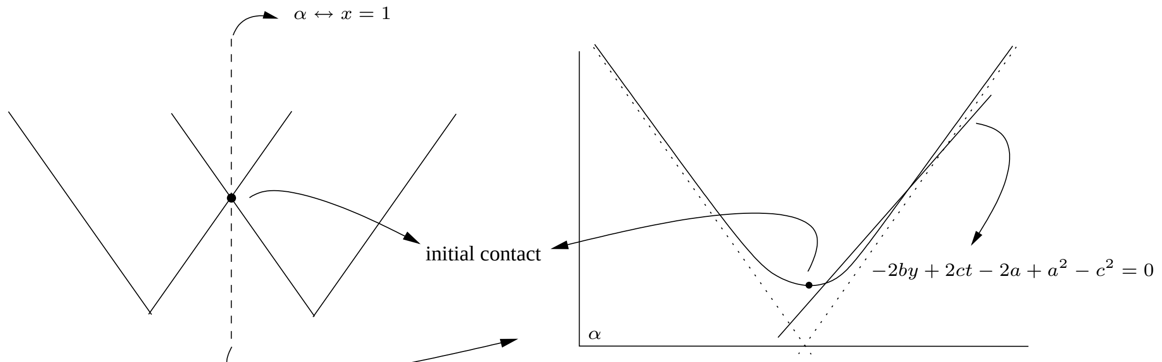


Figure 5. Clean cut between cones.

prism and the second cone, but the initial contact is situated above the line, effectively cutting it out of the intersection. All this is illustrated in Figure 6.



$\alpha \leftrightarrow x = 1$
initial contact
$-2by + 2ct - 2a + a^2 - c^2 = 0$
$\alpha$

Figure 6. The initial contact cut out.

We notice how the plane in which the rim lies and the rim itself is the evil do-er. If neither rim intersects the mantel of the other space-time prism, then the intersection of mantels is the same as an intersection of cones. In which case the initial contact will not be cut out and can be used to determine if there is intersection in this manner.

Using contraposition on the statement in the previous paragraph we get: if there is an intersection and no initial contact is in the intersection then a rim must intersect the other space-time prism's mantel.

To verify intersection with the apexes and initial contacts is straightforward. Verifying if a rim intersects a mantel results in solving a quartic polynomial equation in one variable and verifying the solution in a single inequality in which no variable appears with a degree higher than one.

### 5.2 *Outline of the solution*

Suppose, for the remainder of this section, we wish to verify if the space-time prisms $\mathcal{P}_1 = \mathcal{P}(t_1, x_1, y_1, t_2, x_2, y_2, v_1)$ and $\mathcal{P}_2 = \mathcal{P}(t_3, x_3, y_3, t_4, x_4, y_4, v_2)$ intersect. Moreover, we assume the space-time prisms are non-empty, i.e., $(x_2 - x_1)^2 + (y_2 - y_1)^2 \leq (t_2 - t_1)^2 v_1^2$ and $(x_4 - x_3)^2 + (y_4 - y_3)^2 \leq (t_4 - t_3)^2 v_2^2$.

We first observe that an intersection between space-time prisms can be classified into three, mutually exclusive, cases. The three cases then are:

**(I)** an apex of one space-time prism is contained in the other, i.e.,

$$\tau\mathcal{P}_1 \cap \mathcal{P}_2 \neq \emptyset \text{ or } \mathcal{P}_1 \cap \tau\mathcal{P}_2 \neq \emptyset;$$

**(II)** not **(I)**, but the rim of one space-time prism intersects the mantel of the other, i.e.,

$$\rho\mathcal{P}_1 \cap \partial\mathcal{P}_2 \neq \emptyset \text{ or } \rho\mathcal{P}_2 \cap \partial\mathcal{P}_1 \neq \emptyset;$$

**(III)** not **(I)** and not **(II)** and the initial contact of the upper or lower cones is in the intersection of the space-time prisms, i.e.,

$$\mathsf{IC}(\mathsf{C}_1^-, \mathsf{C}_2^-) \subset \mathcal{P}_1 \cap \mathcal{P}_2 \text{ or } \mathsf{IC}(\mathsf{C}_1^+, \mathsf{C}_2^+) \subset \mathcal{P}_1 \cap \mathcal{P}_2.$$

If none of these three cases occur then the space-time prisms do not intersect, as we show in the correctness proof below. First, we give the following geometric lemma.

Lemma 5.1 *If $\mathcal{P}_1 \cap \mathcal{P}_2 \neq \emptyset$, $\tau\mathcal{P}_1 \cap \mathcal{P}_2 = \emptyset$ and $\tau\mathcal{P}_2 \cap \mathcal{P}_1 = \emptyset$, then $\partial\mathcal{P}_1 \cap \partial\mathcal{P}_2 \neq \emptyset$.*

*Proof* From the assumptions, we know there is a point $p_1$ in $\mathcal{P}_2$, e.g., an apex of $\mathcal{P}_2$, that is not in $\mathcal{P}_1$. Also, there is a point $p_2$ that is in $\mathcal{P}_2$ and in $\mathcal{P}_1$. The line segment bounded by $p_1$ and $p_2$ lies in $\mathcal{P}_2$, since $\mathcal{P}_2$ is convex. The line segment cuts the mantel of $\mathcal{P}_1$ since $p_2$ is inside $\mathcal{P}_1$ and $p_1$ is not. Let $p$ be this point where the segment bounded by $p_1$ and $p_2$ intersects $\partial\mathcal{P}_1$. This point lies either on the upper-half space-time prism $\mathcal{P}_1^+$ or on the bottom-half space-time prism $\mathcal{P}_1^-$. Let $r$ be the apex of this half space-time prism. Since $p$ is inside $\mathcal{P}_2$ and $r$ is not, the line segment bounded by $p$ and $r$ must cut $\partial\mathcal{P}_2$ in a point $q$. This point lies of course on $\partial\mathcal{P}_2$ and on $\partial\mathcal{P}_1$ since the line segment bounded by $p$ and $r$ is a part of $\partial\mathcal{P}_1$. Hence their mantels must have a non-empty intersection if the space-time prisms have a non-empty intersection and neither space-time prism contains the apexes of the other. □

Now, we show that if $\mathcal{P}_1$ and $\mathcal{P}_2$ intersect and neither (**I**), nor (**II**) occur, then (**III**) occurs.

Theorem 5.2 *If $\mathcal{P}_1 \cap \mathcal{P}_2 \neq \emptyset$, $\tau\mathcal{P}_1 \cap \mathcal{P}_2 = \emptyset$, $\mathcal{P}_1 \cap \tau\mathcal{P}_2 = \emptyset$, $\rho\mathcal{P}_1 \cap \partial\mathcal{P}_2 = \emptyset$ and $\rho\mathcal{P}_2 \cap \partial\mathcal{P}_1 = \emptyset$, then $\mathsf{IC}(\mathsf{C}_1^-, \mathsf{C}_2^-) \subset \mathcal{P}_1 \cap \mathcal{P}_2$ or $\mathsf{IC}(\mathsf{C}_1^+, \mathsf{C}_2^+) \subset \mathcal{P}_1 \cap \mathcal{P}_2$.*

*Proof* Let us assume that the hypotheses of the statement of the theorem is true. It is sufficient to prove that either $\mathsf{C}_1^- \cap \mathsf{C}_2^- \subset \mathcal{P}_1^- \cap \mathcal{P}_2^-$ or $\mathsf{C}_1^+ \cap \mathsf{C}_2^+ \subset \mathcal{P}_1^+ \cap \mathcal{P}_2^+$. We will split the proof in two cases. From the fourth and fifth hypotheses it follows that either (1) $\rho\mathcal{P}_1 \subset \mathcal{P}_2$ or $\rho\mathcal{P}_2 \subset \mathcal{P}_1$; or (2) $\rho\mathcal{P}_1 \cap \mathcal{P}_2 = \emptyset$ and $\rho\mathcal{P}_2 \cap \mathcal{P}_1 = \emptyset$.

**Case (1)**: We assume $\rho\mathcal{P}_2 \subset \mathcal{P}_1$ (the case $\rho\mathcal{P}_1 \subset \mathcal{P}_2$ is completely analogous). We prove $\mathsf{C}_1^- \cap \mathsf{C}_2^- \subset \mathcal{P}_1^- \cap \mathcal{P}_2^-$ (the case for upper cones is completely analogous). The following argument is illustrated in Figure 7.

Since $\rho\mathcal{P}_2 \subset \mathcal{P}_1$, we know that $\rho\mathcal{P}_2$ is inside $\mathsf{C}_1^-$, and $(t_3, x_3, y_3)$ is outside. We can show that $v_2 < v_1$. Consider the plane spanned by the two axis of symmetry of both $\mathsf{C}_1^-$ and $\mathsf{C}_2^-$. Both $\mathsf{C}_1^-$ and $\mathsf{C}_2^-$ intersect this plane in two half lines each. Moreover, we know that $\mathsf{C}_1^-$ intersects the axis of symmetry of $\mathsf{C}_2^-$. Let $t_0$ be the moment at which this happens. Obviously $t_0 > t_1$, but we also know $t_0 > t_3$ since $(t_3, x_3, y_3)$ is outside $\mathsf{C}_1^-$. We have that $v_1(t_0 - t_1) = \sqrt{(x_1 - x_3)^2 + (y_1 - y_3)^2}$. Since $\rho\mathcal{P}_2$ is inside $\mathsf{C}_1^-$ and $(t_3, x_3, y_3)$ is outside, this means both half lines from $\mathsf{C}_2^-$ intersect the half lines from $\mathsf{C}_1^-$. Let $t_0'$ and $t_0''$ be the moments in time at which this happens and let $t_0' > t_0''$. We have again that $t_0' > t_1$ and $t_0' > t_3$. Then $v_1(t_0' - t_1) = \sqrt{(x_1 - x_3)^2 + (y_1 - y_3)^2} + v_2(t_0' - t_3)$ if and only if $v_1(t_0' - t_0) = v_2(t_0' - t_3)$. Since $t_0 > t_3$, we get $v_2 < v_1$. This is depicted in Figure 7.

It follows that every straight half line starting in $(t_3, x_3, y_3)$ on $\mathsf{C}_2^-$ intersects $\mathsf{C}_1^-$ between $(t_3, x_3, y_3)$ and $\rho\mathcal{P}_2$, since $\rho\mathcal{P}_2$ is inside $\mathsf{C}_1^-$, and $(t_3, x_3, y_3)$ is outside. We also know that this line does not intersect $\mathsf{C}_1^-$ beyond $\rho\mathcal{P}_2$ since the cone $\mathsf{C}_2^-$ is entirely inside $\mathsf{C}_1^-$ beyond the rim $\rho\mathcal{P}_2$. Therefore, $\mathsf{C}_1^- \cap \mathsf{C}_2^- \subset \mathcal{P}_2^-$.

Clearly, $\mathcal{P}_2^-$ intersects $\mathcal{P}_1^-$ since it can not intersect $\mathcal{P}_1^+$. We know $\mathsf{C}_1^- \cap \partial\mathcal{P}_2^-$ is a closed continuous curve that lies entirely in $\mathsf{C}_1^-$. This curve is also contained in $\mathcal{P}_1^-$. Indeed, if we assume this is not the case, then it intersects the plane in which $\rho\mathcal{P}_1$ lies, and hence it intersects $\rho\mathcal{P}_1$ itself, contradicting the assumption $\rho\mathcal{P}_1 \cap \partial\mathcal{P}_2 = \emptyset$.

**Case (2)**: Now assume $\rho\mathcal{P}_1 \cap \mathcal{P}_2 = \emptyset$ and $\rho\mathcal{P}_2 \cap \mathcal{P}_1 = \emptyset$. Clearly, $v_1$ can not be equal to $v_2$, otherwise the depicted intersection can not occur. So suppose without loss of generality that $v_2 < v_1$. Now either $\mathcal{P}_2^-$ intersects both $\mathcal{P}_1^-$ and $\mathcal{P}_1^+$ or $\mathcal{P}_2^+$ intersects both $\mathcal{P}_1^-$ and $\mathcal{P}_1^+$. These cases are mutually exclusive because

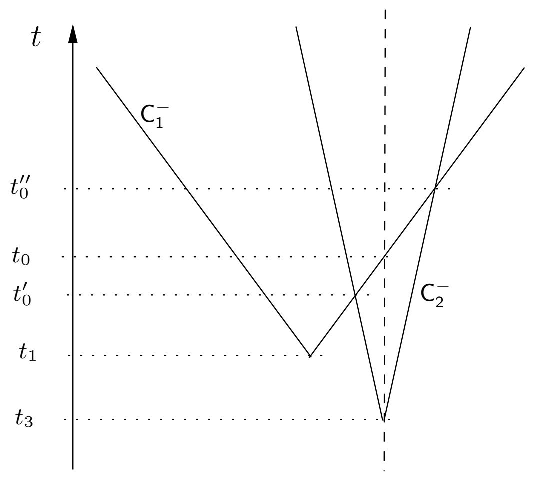


Figure 7. Illustration to the proof.

of the following. If $\mathcal{P}_2^+$ intersects $\mathcal{P}_1^+$ then $\rho\mathcal{P}_2$ is inside $\mathsf{C}_1^+$, likewise if $\mathcal{P}_2^-$ intersects $\mathcal{P}_1^-$ then $\rho\mathcal{P}_2$ is inside $\mathsf{C}_1^-$. Hence $\rho\mathcal{P}_2 \subset \mathcal{P}_1$ which contradicts our hypothesis. If $\mathcal{P}_2^+$ intersects $\mathcal{P}_1^-$ then $\rho\mathcal{P}_2$ must be outside $\mathsf{C}_1^-$ and thus $\mathcal{P}_2^-$ must be as well, hence $\mathcal{P}_2^-$ intersects neither $\mathcal{P}_1^-$ nor $\mathcal{P}_1^+$. Likewise, if $\mathcal{P}_2^-$ intersects $\mathcal{P}_1^+$ then $\mathcal{P}_2^+$ can not intersect $\mathcal{P}_1$.

To prove that if $\mathcal{P}_2^-$ intersects $\mathcal{P}_1^-$ then it also intersects $\mathcal{P}_1^+$ and if $\mathcal{P}_2^-$ intersects $\mathcal{P}_1^+$ then it also intersects $\mathcal{P}_1^-$ we proceed as follows (the case for $\mathcal{P}_2^+$ is analogous). Suppose $\mathcal{P}_2^-$ intersects $\mathcal{P}_1^-$, then $\mathcal{P}_2^- \cap \mathcal{P}_1^- \subset \mathcal{P}_1$, but $\rho\mathcal{P}_2$ is outside $\mathcal{P}_1$, that means $\mathcal{P}_2^-$ must intersect $\mathcal{P}_1^+$ since it can not intersect $\mathcal{P}_1^-$ anymore. This is the "what goes in must come out"-principle. Likewise, suppose $\mathcal{P}_2^-$ intersects $\mathcal{P}_1^+$, then $\mathcal{P}_2^- \cap \mathcal{P}_1^+ \subset \mathcal{P}_1$, but $(t_3, x_3, y_3)$ is outside $\mathcal{P}_1$, that means $\mathcal{P}_2^-$ must intersect $\mathcal{P}_1^-$ since it can not intersect $\mathcal{P}_1^-$ anymore.

So suppose now that $\mathcal{P}_2^-$ intersects both $\mathcal{P}_1^-$ and $\mathcal{P}_1^+$ (the case for $\mathcal{P}_2^+$ is completely analogous). If $\mathcal{P}_2^-$ intersects $\mathcal{P}_1^-$ that means $\rho\mathcal{P}_2$ is completely inside $\mathsf{C}_1^-$ and therefore that $\mathsf{C}_1^- \cap \mathsf{C}_2^- \subset \mathcal{P}_2^-$. We proceed like in the first case, we know that $\mathsf{C}_1^- \cap \mathcal{P}_2^-$ is a closed continuous curve. This curve lies entirely in $\mathsf{C}_1^-$. If this curve is not entirely in $\mathcal{P}_1^-$ that means it intersects the plane in which $\rho\mathcal{P}_1$ lies, and hence intersects $\rho\mathcal{P}_1$ itself. But this is contradictory to the assumption that $\rho\mathcal{P}_1 \cap \partial\mathcal{P}_2 = \emptyset$. $\square$

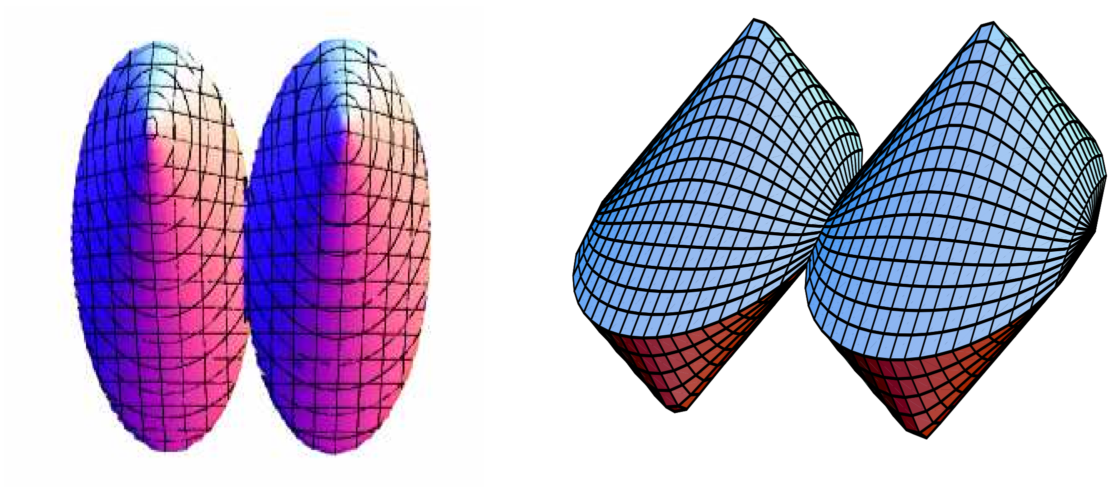


Figure 8. Case **(II)** is not redundant: $\mathcal{P}(0, 0, 0, 2, 0, 2, 1.9)$ and $\mathcal{P}(0, 3, 0, 2, 3, 2, 1.9)$ seen from the top and the side.

In Theorem 5.2, we proved that if there is an intersection and neither rim cuts the other space-time prism's mantel and neither apex of a space-time prism is contained in the other then there must be an initial contact in the intersection. Visualizing how space-time prisms intersect might tempt one to think there is always an initial contact in the intersection. There exist counterexamples in which there is an intersection and no initial contact is in that intersection. That means case **(II)** is not redundant. This situation is depicted in Figure 8. The space-time prisms are $\mathcal{P}(0,0,0,2,0,2,1.9)$ and $\mathcal{P}(0,3,0,2,3,2,1.9)$.

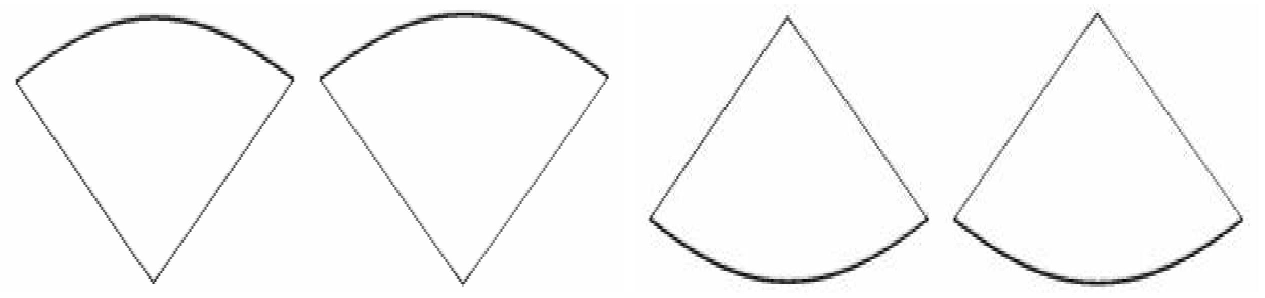


Figure 9. Intersection of Figure 8 with the plane $y = 0$ (left) and with the plane $y = 3$ (right).

It is clear that the initial contact of the bottom cones lies in the plane spanned by the axis of symmetry of those bottom cones, in this case this is the plane $y = 0$. The intersection of Figure 8 can be seen in Figure 9 on the left, where the two space-time prisms clearly have no intersection and thus no initial contact in the intersection.

In the case of the upper cones the initial contact must lie in the plane $y = 3$. The intersection of Figure 8 can be seen in Figure 9 on the right, where the two space-time prisms clearly have no intersection and there is again no initial contact in the intersection.

This concludes the outline.

### 5.3 *A formula for Case (I)*

In Case **(I)**, we verify whether $\tau\mathcal{P}_1 \cap \mathcal{P}_2 \neq \emptyset$ or $\mathcal{P}_1 \cap \tau\mathcal{P}_2 \neq \emptyset$. To check if that is the case we merely need to verify if one of the apexes satisfies the set of equations of the other space-time prism. In this way we obtain

$$\begin{aligned}
&\Phi_{\mathbf{I}}\left(t_1,x_1,y_1,t_2,x_2,y_2,v_1,t_3,x_3,y_3,t_4,x_4,y_4,v_2\right) := \\
&\qquad (\Psi_{\mathcal{P}}\left(t_3,x_3,y_3,t_1,x_1,y_1,t_2,x_2,y_2,v_1\right) \vee \Psi_{\mathcal{P}}\left(t_4,x_4,y_4,t_1,x_1,y_1,t_2,x_2,y_2,v_1\right) \vee \\
&\qquad \Psi_{\mathcal{P}}\left(t_1,x_1,y_1,t_3,x_3,y_3,t_4,x_4,y_4,v_2\right) \vee \Psi_{\mathcal{P}}\left(t_2,x_2,y_2,t_3,x_3,y_3,t_4,x_4,y_4,v_2\right)).
\end{aligned}$$

For the following sections we assume that the apex sets of the space-time prisms are not singletons, i.e., $t_1 < t_2$ and $t_3 < t_4$.

### 5.4 *A formula for Case (II)*

Now, let us assume that $\Phi_{\mathbf{I}}$ failed in the previous section. Note that we can always apply a speed-preserving (Kuijpers and Othman 2007) transformation to $\mathbf{R} \times \mathbf{R}^2$ to obtain easier coordinates. We can always find a transformation such that $(t'_1, x'_1, y'_1) = (0,0,0)$ and that the line-segment connecting $(t'_1, x'_1, y'_1)$ and $(t'_2, x'_2, y'_2)$ is perpendicular to the $y$-axis, i.e., $y'_2 = 0$. This transformation is a composition of a translation in $\mathbf{R} \times \mathbf{R}^2$, a spatial rotation in $\mathbf{R}^2$ and a scaling in $\mathbf{R} \times \mathbf{R}^2$ (Kuijpers and Othman 2007). Let the coordinates without a prime be the original set, and let coordinates with a prime be the image of the same coordinates without a prime under this transformation. Note that we do not need to transform back because the query is invariant under such transformations (Kuijpers and Othman 2007). The following formula returns the transformed coordinates $(t', x', y')$ of $(t, x, y)$ given the points $(t_1, x_1, y_1)$ and $(t_2, x_2, y_2)$:

$$\varphi_A(t_1,x_1,y_1,t_2,x_2,y_2,t,x,y,t',x',y') := \Big(y_2 \neq y_1 \ \wedge \ t' = (t-t_1)\sqrt{(x_2-x_1)^2+(y_2-y_1)^2}$$

$$\wedge\ x' = (x - x_1)(x_2 - x_1) + (y - y_1)(y_2 - y_1)\ \wedge\ y' = (x - x_1)(y_1 - y_2) + (y - y_1)(x_2 - x_1)\big)$$
$$\vee\ \big(y_2 = y_1\ \wedge\ t' = (t - t_1)\ \wedge\ x' = (x - x_1)\ \wedge\ y' = (y - y_1)\big)\,.$$

The translation is over the vector $(-t_1, -x_1, -y_1)$, the rotation over minus the angle that $(t_2 - t_1, x_2 - x_1, y_2 - y_1)$ makes with the $x$-axis, and a scaling by a factor $\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$. Notice that the rotation and scaling only need to occur if $y_2$ is not already in place, i.e., if $y_2 \neq y_1$.

The formula $\psi_{crd}(t'_1, x'_1, y'_1, t_1, x_1, y_1, t'_2, x'_2, y'_2, t_2, x_2, y_2, t'_3, x'_3, y'_3, t_3, x_3, y_3, t'_4, x'_4, y'_4, t_4, x_4, y_4)$ is short for $\varphi_A(t_1, x_1, y_1, t_2, x_2, y_2, t_1, x_1, y_1, t'_1, x'_1, y'_1)\ \wedge\ \varphi_A(t_1, x_1, y_1, t_2, x_2, y_2, t_2, x_2, y_2, t'_2, x'_2, y'_2)\ \wedge\ \varphi_A(t_1, x_1, y_1, t_2, x_2, y_2, t_3, x_3, y_3, t'_3, x'_3, y'_3)\ \wedge\ \varphi_A(t_1, x_1, y_1, t_2, x_2, y_2, t_4, x_4, y_4, t'_4, x'_4, y'_4)$.

This transformation yields some simple equations for the rim $\rho\mathcal{P}_1$:

$$\rho\mathcal{P}_1 \leftrightarrow \begin{cases} x^2 + y^2 = t^2 v_1^2 \\ 2x(-x'_2) + x'^2_2 = v_1^2(2t(-t'_2) + t'^2_2) \\ 0 \leq t \leq t'_2\ . \end{cases}$$

Not only that, but with these equations we can deduce a simple parametrization in the $x$-coordinate for the rim,

$$\rho\mathcal{P}_1 \leftrightarrow \begin{cases} t = \frac{2xx'_2 - x'^2_2 + v_1^2 t'^2_2}{2v_1^2 t'_2} \\ y = \pm\sqrt{v_1^2\left(\frac{2xx'_2 - x'^2_2 + v_1^2 t'^2_2}{2v_1^2 t'_2}\right)^2 - x^2} \\ 0 \leq t \leq t'_2\ . \end{cases}$$

We remark that this implies $t'_2 \neq 0$ and $v_1 \neq 0$. If $t'_2 = 0$, then $\mathcal{P}_1$ is a point, hence degenerate. If $v_1 = 0$, then $\mathcal{P}_1$ is a line segment, and again degenerate. Next we will inject these parameterizations in the constraints for $\partial\mathcal{P}_2^+$ and $\partial\mathcal{P}_2^-$ separately. The constraints for $\partial\mathcal{P}_2^-$ are

$$\begin{cases} (x - x'_3)^2 + (y - y'_3)^2 = (t - t'_3)^2 v_2^2 \\ 2x(x'_3 - x'_4) + x'^2_4 - x'^2_3 + 2y(y'_3 - y'_4) + y'^2_4 - y'^2_3 \leq v_2^2\left(2t(t'_3 - t'_4) + t'^2_4 - t'^2_3\right) \\ t'_3 \leq t \leq t'_4\ . \end{cases}$$

We will explain how to proceed to compute the intersection with $\partial\mathcal{P}_2^-$ and simply reuse formulas for intersection with $\partial\mathcal{P}_2^+$. First, we insert our expressions for $x$ and $y$ in the first equation. This is equivalent to computing intersections of $\rho\mathcal{P}_1$ with $\mathsf{C}_2^-$ and gives

$$\left(x - x'_3\right)^2 + \left(\pm\sqrt{v_1^2\left(\frac{2xx'_2 - x'^2_2 + v_1^2 t'^2_2}{2v_1^2 t'_2}\right)^2 - x^2} - y'_3\right)^2 = \left(\frac{2xx'_2 - x'^2_2 + v_1^2 t'^2_2}{2v_1^2 t'_2} - t'_3\right)^2 v_2^2,$$

or equivalently

$$\pm 2y'_3\sqrt{v_1^2\left(2xx'_2 - x'^2_2 + v_1^2 t'^2_2\right)^2 - \left(2v_1^2 t'_2\right)^2 x^2} = \left(2xx'_2 - x'^2_2 + v_1^2 t'^2_2 - \left(2v_1^2 t'_2\right) t'_3\right)^2 v_2^2 - \left(2v_1^2 t'_2\right)^2 (x - x'_3)^2$$
$$- \left(2v_1^2 t'_2\right)^2 y'^2_3 - \left(v_1^2\left(2xx'_2 - x'^2_2 + v_1^2 t'^2_2\right)^2 - \left(2v_1^2 t'_2\right)^2 x^2\right)$$

or equivalently

$$\pm v_1 2y'_3\sqrt{x^2 4\left(x'^2_2 - v_1^2 t'^2_2\right) + x4x'^2_2\left(v_1^2 t'^2_2 - x'^2_2\right) + \left(v_1^2 t'^2_2 - x'^2_2\right)^2} = x^2 4x'^2_2\left(v_2^2 - v_1^2\right) +$$

$$
\begin{aligned}
&x4\left(-x_2'^2v_2^2\left(-x_2'^2+v_1^2t_2'^2-4v_1^4t_2'^2t_3'\right)+2v_1^4t_2'^2x_3'+v_1^2x_2'\left(v_1^2t_2'^2-x_2'^2\right)\right)\\
&+\left(v_2^2\left(-x_2'^2+v_1^2t_2'^2-4v_1^4t_2'^2t_3'\right)^2-4v_1^4t_2'^2\left(x_3'^2+y_3'^2\right)-v_1^4\left(-x_2'^2+v_1^2t_2'^2\right)\right)\ .
\end{aligned}
$$

By squaring left and right hand in this last expression, we rid ourselves of the square root and obtain the following polynomial equation of degree four. Squaring may create new solutions, so to ensure we only get useful solutions, we have to add the condition that the square root exists. This is the case if and only if

$$\phi_{\sqrt{}}(x,t_2',x_2',v_1):=x^24\left(x_2'^2-v_1^2t_2'^2\right)+x4x_2'^2\left(v_1^2t_2'^2-x_2'^2\right)+\left(v_1^2t_2'^2-x_2'^2\right)^2\geq 0$$

is satisfied.

We notice that if $\mathcal{P}_1$ is degenerate, i.e., $x_2'^2=v_1^2t_2'^2$, then the square root vanishes and the polynomial in $\phi_4$ is the square of a polynomial of degree two, yielding to at most two roots and intersection points as we expect. The case were $v_1=0$ is captured by the formula in the next section, that is why we leave that case out here and demand that $v_1\neq 0$. So the following still works if one or both space-time prisms is degenerate:

$$
\begin{aligned}
&\phi_4(x,t_2',x_2',v_1,t_3',x_3',y_3',v_2):=\exists a\exists b\exists c\exists d\exists e\left(ax^4+bx^3+cx^2+dx+e=0\right.\\
&\wedge\ a=\left(4x_2'^2\left(v_2^2-v_1^2\right)\right)^2\ \wedge\ b=-32x_2'^4v_2^2\left(v_2^2-v_1^2\right)\left(-x_2'^2+v_1^2t_2'^2-4v_1^4t_2'^2t_3'+2v_1^4t_2'^2x_3'+v_1^2x_2'\left(v_1^2t_2'^2-x_2'^2\right)\right)\\
&\wedge\ c=8\left(x_2'^2-v_1^2t_2'^2\right)\left(-4v_1^4t_2'^2\left(x_3'^2+y_3'^2\right)+v_2^2\left(-x_2'^2+v_1^2t_2'^2-4v_1^4t_2'^2t_3'\right)^2-v_1^4\left(-x_2'^2+v_1^2t_2'^2\right)\right)\\
&+\left(2v_1y_3'\right)^2\left(x_2'^2-v_1^2t_2'^2\right)+\left(4\left(-x_2'^2v_2^2\left(-x_2'^2+v_1^2t_2'^2-4v_1^4t_2'^2t_3'\right)+2v_1^4t_2'^2x_3'+v_1^2x_2'\left(v_1^2t_2'^2-x_2'^2\right)\right)\right)^2\\
&\wedge\ d=8\left(-x_2'^2v_2^2\left(-x_2'^2+v_1^2t_2'^2-4v_1^4t_2'^2t_3'\right)+2v_1^4t_2'^2x_3'+v_1^2x_2'\left(v_1^2t_2'^2-x_2'^2\right)\right)\\
&\left(v_2^2\left(-x_2'^2+v_1^2t_2'^2-4v_1^4t_2'^2t_3'\right)^2-4v_1^4t_2'^2\left(x_3'^2+y_3'^2\right)-v_1^4\left(-x_2'^2+v_1^2t_2'^2\right)\right)+\left(2v_1y_3'\right)^2\left(4x_2'^2\left(v_1^2t_2'^2-x_2'^2\right)\right)\\
&\wedge\ e=\left(2v_1y_3'\right)^2\left(v_1^2t_2'^2-x_2'^2\right)^2+\left.\left(v_2^2\left(-x_2'^2+v_1^2t_2'^2-4v_1^4t_2'^2t_3'\right)^2-4v_1^4t_2'^2\left(x_3'^2+y_3'^2\right)-v_1^4\left(-x_2'^2+v_1^2t_2'^2\right)\right)^2\right).
\end{aligned}
$$

The quantifiers we introduced here are only in place for esthetical considerations and can be eliminated by direct substitution.

We note that if $v_1=v_2$, we get polynomials of degree merely two. This can be solved in an exact manner using nested square roots (or Maple if you will). This gives us at most four values for $x$. Let

$$\phi_{roots}(x_a,x_b,x_c,x_d,t_2',x_2',v_1,t_3',x_3',y_3',v_2)$$

be a formula that returns all four real roots, if they exist, that satisfy both $\phi_4(x,t_2',x_2',v_1,t_3',x_3',y_3',v_2)$ and $\phi_{\sqrt{}}(x,t_2',x_2',v_1)$. We substitute these values in the parameter equations of $\rho\mathcal{P}_1$. By substituting these in the last equation above, we can determine the sign of the square root we need to take for $y$. A point $(t,x,y)$ satisfies the following formula is a point on $\rho\mathcal{P}_1$, but instead of using the square root for $y$, we use an expression from above to get the correct sign for the square root if $y_3'\neq 0$. If $y_3'=0$ we have to use the square root expression and then it does not matter which sign the square root has; we need both:

$$
\begin{aligned}
&\psi_\rho(t,x,y,t_2',x_2',v_1,t_3',x_3',y_3',v_2):=\left(y_3'\neq 0\ \wedge\ t\left(2v_1^2t_2'\right)=2xx_2'-x_2'^2+v_1^2t_2'^2\ \wedge\right.\\
&2y_3'\left(2v_1^2t_2'\right)y=\left(2xx_2'-x_2'^2+v_1^2t_2'^2-\left(2v_1^2t_2'\right)t_3'\right)^2v_2^2-\left(2v_1^2t_2'\right)^2(x-x_3')^2-\left(2v_1^2t_2'\right)^2y_3'^2\\
&\left.-\left(v_1^2\left(2xx_2'-x_2'^2+v_1^2t_2'^2\right)^2-\left(2v_1^2t_2'\right)^2x^2\right)\ \wedge\ \ 0\leq t\leq t_2'\ \right)\vee\left(y_3'=0\ \wedge\ t\left(2v_1^2t_2'\right)=2xx_2'-x_2'^2+v_1^2t_2'^2\right.\\
&\left.\wedge\ 0\leq t\leq t_2'\ \wedge\ \left(2v_1^2t_2'\right)^2y^2=\left(2xx_2'-x_2'^2+v_1^2t_2'^2\right)^2-\left(2v_1^2t_2'\right)^2x^2\right).
\end{aligned}
$$

The four roots give us four spatio-temporal points on $\rho\mathcal{P}_1 \cap \mathsf{C}_2^-$. In order for these points $(t, x, y)$ to be in $\rho\mathcal{P}_1 \cap \partial\mathcal{P}_2^-$, they need to satisfy

$$\psi_-(t,x,y,t'_3,x'_3,y'_3,t'_4,x'_4,y'_4,v_2) :=$$
$$2x(x'_3 - x'_4) + x'^2_4 - x'^2_3 + 2y(y'_3 - y'_4) + y'^2_4 - y'^2_3 \leq v_2^2\left(2t(t'_3 - t'_4) + t'^2_4 - t'^2_3\right) .$$

This formula returns TRUE if $(t, x, y)$ lies in the same half space as the bottom-half space-time prism.

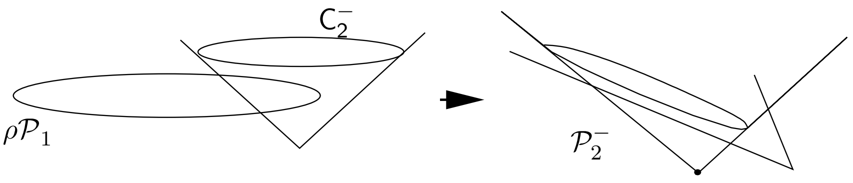


Figure 10. The rim intersects the cone and solutions are verified in a half-space.

The formula $\psi_+$ returns TRUE if $(t, x, y)$ lies in the same half space as the upper-half space-time prism, i.e., $\psi_+(t,x,y,t'_3,x'_3,y'_3,t'_4,x'_4,y'_4,v_2) := \psi_-(t,x,y,t'_4,x'_4,y'_4,t'_3,x'_3,y'_3,v_2)$. By combining $\psi_\rho(t,x,y,t'_2,x'_2,v_1,t'_3,x'_3,y'_3,v_2)$ and $\psi_-(t,x,y,\hat{t},\hat{x},\hat{y},\tilde{t},\tilde{x},\tilde{y},v)$ we get a formula that decides the emptiness of the intersection $\rho\mathcal{P}_1 \cap \partial\mathcal{P}_2^-$ in terms of a parameter $x$:

$$\psi_{\rho\cap\partial^\pm}(x,t'_2,x'_2,v_1,t'_3,x'_3,y'_3,v_2,\hat{t},\hat{x},\hat{y},\tilde{t},\tilde{x},\tilde{y},v) :=$$
$$\exists y\ \left(y'_3 = 0\ \wedge\ y^2\left(2v_1^2t'_2\right)^2 = \left(2xx'_2 - x'^2_2 + v_1^2t'^2_2\right)^2 v_1^2 - \left(2v_1^2t'_2\right)^2x^2\right)\ \vee\ \left(y'_3 \neq 0\ \wedge\right.$$
$$2y'_3\left(2v_1^2t'_2\right)y = \left(2xx'_2 - x'^2_2 + v_1^2t'^2_2 - \left(2v_1^2t'_2\right)t'_3\right)^2v_2^2 - \left(2v_1^2t'_2\right)^2(x - x'_3)^2 - \left(2v_1^2t'_2\right)^2y'^2_3 -$$
$$\left.\left(v_1^2\left(2xx'_2 - x'^2_2 + v_1^2t'^2_2\right)^2 - \left(2v_1^2t'_2\right)^2x^2\right)\right)\wedge\ \left(2x(\hat{x} - \tilde{x}) + \tilde{x}^2 - \hat{x}^2 + 2y(\hat{y} - \tilde{y}) + \tilde{y}^2 - \hat{y}^2\right)\left(2v_1^2t'_2\right) \leq$$
$$v^2\left(2\left(2v_1^2t'_2\right)\left(2xx'_2 - x'^2_2 + v_1^2t'^2_2\right)(\hat{t} - \tilde{t}) + \left(2v_1^2t'_2\right)\left(\tilde{t}^2 - \hat{t}^2\right)\right)\ \wedge\ 0 \leq t'_2\left(2xx'_2 - x'^2_2 + v_1^2t'^2_2\right) \leq 2v_1^2t'^3_2$$
$$\wedge\ \left(\hat{t}\left(2v_1^2t'^2_2\right) \leq t'_2\left(2xx'_2 - x'^2_2 + v_1^2t'^2_2\right) \leq \tilde{t}\left(2v_1^2t'^2_2\right)\ \vee\ \tilde{t}\left(2v_1^2t'^2_2\right) \leq t'_2\left(2xx'_2 - x'^2_2 + v_1^2t'^2_2\right) \leq \hat{t}\left(2v_1^2t'^2_2\right)\right).$$

We are ready now to construct the formula that decides if $\rho\mathcal{P}_1$ and $\mathcal{P}_2^-$ have a non-empty intersection:

$$\varphi_{\rho_1\cap\partial_2^-}(t'_2,x'_2,v_1,t'_3,x'_3,y'_3,t'_4,x'_4,y'_4,v_2) := \exists x\exists x_a\exists x_b\exists x_c\exists x_d\ \left(\ \phi_{roots}(x_a,x_b,x_c,x_d,t'_2,x'_2,v_1,t'_3,x'_3,y'_3,v_2)\right.$$
$$\left.\wedge\ (x = x_a \vee x = x_b \vee x = x_c \vee x = x_d)\ \wedge\ \psi_{\rho\cap\partial^\pm}(x,t'_2,x'_2,v_1,t'_3,x'_3,y'_3,v_2,t'_3,x'_3,y'_3,t'_4,x'_4,y'_4,v'_2)\ \right).$$

The formula that decides if $\rho\mathcal{P}_1$ intersects $\partial\mathcal{P}_2^+$ looks strikingly similar:

$$\varphi_{\rho_1\cap\partial_2^+}(t'_2,x'_2,v_1,t'_3,x'_3,y'_3,t'_4,x'_4,y'_4,v_2) := \exists x\exists x_a\exists x_b\exists x_c\exists x_d\ \left(\ \phi_{roots}(x_a,x_b,x_c,x_d,t'_2,x'_2,v_1,t'_4,x'_4,y'_4,v_2)\right.$$
$$\left.\wedge\ (x = x_a \vee x = x_b \vee x = x_c \vee x = x_d)\ \wedge\ \psi_{\rho\cap\partial^\pm}(x,t'_2,x'_2,v_1,t'_4,x'_4,y'_4,v_2,t'_4,x'_4,y'_4,t'_3,x'_3,y'_3,v'_2)\ \right).$$

The quantifiers introduced here can also be eliminated in a straightforward manner. Notice that $\phi_{roots}$ acts as a function rather than a formula that inputs $(t'_2, x'_2, v_1, t'_4, x'_4, y'_4, v_2)$ to construct a polynomial of degree four and returns the four roots $(x_a, x_b, x_c, x_d)$, if they exist, of that polynomial. The existential quantifier for the variable $x$ is used to cycle through those roots to see if any of them does the trick. Finally we are ready to present the formula for Case **(II)**:

$$\Phi_{\mathbf{II}}(t_1,x_1,y_1,t_2,x_2,y_2,v_1,t_3,x_3,y_3,t_4,x_4,y_4,v_2) := (v_1 \neq 0\ \wedge\ v_2 \neq 0)\ \wedge$$
$$\neg\ \Phi_{\mathbf{I}}(t_1,x_1,y_1,t_2,x_2,y_2,v_1,t_3,x_3,y_3,t_4,x_4,y_4,v_2) \wedge\ \exists t'_1\exists x'_1\exists y'_1\exists t'_2\exists x'_2\exists y'_2\exists t'_3\exists x'_3\exists y'_3\exists t'_4\exists x'_4\exists y'_4$$
$$(\ \psi_{crd}(t'_1,x'_1,y'_1,t_1,x_1,y_1,t'_2,x'_2,y'_2,t_2,x_2,y_2,t'_3,x'_3,y'_3,t_3,x_3,y_3,t'_4,x'_4,y'_4,t_4,x_4,y_4) \wedge$$

$$\Big(\varphi_{\rho_1\cap\partial_2^-}(t_2', x_2', v_1, t_3', x_3', y_3', t_4', x_4', y_4', v_2') \ \vee\ \varphi_{\rho_1\cap\partial_2^+}(t_2', x_2', v_1, t_3', x_3', y_3', t_4', x_4', y_4', v_2)\Big) \ \vee$$
$$\psi_{crd}(t_3', x_3', y_3', t_3, x_3, y_3, t_4', x_4', y_4', t_4, x_4, y_4, t_1', x_1', y_1', t_1, x_1, y_1, t_2', x_2', y_2', t_2, x_2, y_2) \ \wedge$$
$$\Big(\varphi_{\rho_1\cap\partial_2^-}(t_3', x_3', v_2, t_1', x_1', y_1', t_2', x_2', y_2', v_1') \ \vee\ \varphi_{\rho_1\cap\partial_2^+}(t_3', x_3', v_2, t_1', x_1', y_1', t_2', x_2', y_2', v_1')\Big)\Big) .$$

The reader may notice that a lot of quantifiers have been introduced in the formula above. These quantifiers are merely there to introduce easier coordinates and can be straightforwardly computed (and eliminated) by the formula $\psi_{crd}$ and hence the formula $\varphi_A(t_1, x_1, y_1, t_2, x_2, y_2, t, x, y, t', x', y')$. The latter actually acts like a function, parameterized by $(t_1, x_1, y_1, t_2, x_2, y_2)$, that inputs $(t, x, y)$ and outputs $(t', x', y')$.

## 5.5 *A formula for Case (III)*

Here, we assume that both $\varphi_{\mathbf{I}}$ and $\varphi_{\mathbf{II}}$ fail. So, there is no apex contained in the other space-time prism and neither rim cuts the mantel of the other space-time prism.

As we proved in Theorem 5.2, the intersection between two half space-time prisms will reduce to the intersection between two cones and that means there is an initial contact that is part of the intersection. To verify if this is the case we compute the two initial contacts and verify if they are effectively part of the intersection.

Using the expression for the initial contact $\mathsf{IC}(\mathsf{C}_1^-, \mathsf{C}_2^-)$, we computed in Section 4.2 we can construct a formula that decides if it is part of $\mathcal{P}_1^- \cap \mathcal{P}_2^-$. We will recycle the formulas $\psi_-$ from the previous section to construct an expression without the need for extra variables. The following formula that returns True if $\mathsf{IC}(\mathsf{C}_1^-, \mathsf{C}_2^-) = (t_0, x_0, y_0)$ satisfies $\psi_-(t_0, x_0, y_0, t', x', y', \hat{t}, \hat{x}, \hat{y}, v)$:

$$\phi_-(t_1, x_1, y_1, v_1, t_3, x_3, y_3, v_2, t', x', y', \hat{t}, \hat{x}, \hat{y}, v) :=$$
$$2(x' - \hat{x})\Big((x_1v_2 + x_3v_1)\sqrt{(x_1 - x_3)^2 + (y_1 - y_3)^2} + v_1\left((t_3 - t_1)v_2\right)(x_3 - x_1)\Big)$$
$$+\, 2(y' - \hat{y})\Big((y_1v_2 + y_3v_1)\sqrt{(x_1 - x_3)^2 + (y_1 - y_3)^2} + v_1\left((t_3 - t_1)v_2\right)(y_3 - y_1)\Big)$$
$$+ \sqrt{(x_1 - x_3)^2 + (y_1 - y_3)^2}(v_1 + v_2)\left(\hat{x}^2 - x'^2 + \hat{y}^2 - y'^2\right) \le v^2\left(\left(\hat{t}^2 - t'^2\right)(v_1 + v_2)\right.$$
$$\left.+\, 2\left(\sqrt{(x_1 - x_3)^2 + (y_1 - y_3)^2} + t_1v_1 + t_3v_2\right)\left(t' - \hat{t}\right)\right)\sqrt{(x_1 - x_3)^2 + (y_1 - y_3)^2} .$$

The following formula expresses that the time coordinate $t_0$ of $\mathsf{IC}(\mathsf{C}_1^-, \mathsf{C}_2^-)$ satisfies the constraints $t' \le t \le t''$ and $\hat{t} \le t \le$:

$$\psi_t\left(t_1, x_1, y_1, v_1, t_3, x_3, y_3, v_2, t', t'', \hat{t}, \right) :=$$
$$t'(v_1 + v_2) \le \sqrt{(x_1 - x_3)^2 + (y_1 - y_3)^2} + t_1v_1 + t_3v_2 \le t''(v_1 + v_2)$$
$$\wedge\ \hat{t}(v_1 + v_2) \le \sqrt{(x_1 - x_3)^2 + (y_1 - y_3)^2} + t_1v_1 + t_3v_2 \le (v_1 + v_2) .$$

Now, $\mathsf{IC}(\mathsf{C}_1^-, \mathsf{C}_2^-) \subset \mathcal{P}_1^- \cap \mathcal{P}_2^-$ if and only if $\psi_{\mathsf{IC}^-}(t_1, x_1, y_1, t_2, x_2, y_2, v_1, t_3, x_3, y_3, t_4, x_4, y_4, v_2)$ where $\psi_{\mathsf{IC}^-}(t_1, x_1, y_1, t_2, x_2, y_2, v_1, t_3, x_3, y_3, t_4, x_4, y_4, v_2) := \psi_t(t_1, x_1, y_1, v_1, t_3, x_3, y_3, v_2, t_1, t_2, t_3, t_4) \ \wedge\ \phi_-(t_1, x_1, y_1, v_1, t_3, x_3, y_3, v_2, t_1, x_1, y_1, t_2, x_2, y_2, v_1) \ \wedge\ \phi_-(t_1, x_1, y_1, v_1, t_3, x_3, y_3, v_2, t_3, x_3, y_3, t_4, x_4, y_4, v_2)$ and $\mathsf{IC}(\mathsf{C}_1^+, \mathsf{C}_2^+) \subset \mathcal{P}_1^+ \cap \mathcal{P}_2^+$ if and only if $\psi_{\mathsf{IC}^+}(t_1, x_1, y_1, t_2, x_2, y_2, v_1, t_3, x_3, y_3, t_4, x_4, y_4, v_2) := \psi_t(t_2, x_2, y_2, v_1, t_4, x_4, y_4, v_2, t_1, t_2, t_3, t_4) \ \wedge\ \phi_-(t_2, x_2, y_2, v_1, t_4, x_4, y_4, v_2, t_2, x_2, y_2, t_1, x_1, y_1, v_1) \ \wedge\ \phi_-(t_2, x_2, y_2, v_1, t_4, x_4, y_4, v_2, t_4, x_4, y_4, t_3, x_3, y_3, v_2)$.

The formula that expresses the criterium for Case **(III)** then looks as follows:

$$\Phi_{\mathbf{III}}\left(t_1, x_1, y_1, t_2, x_2, y_2, v_1, t_3, x_3, y_3, t_4, x_4, y_4, v_2\right) :=$$

$$(v_1 + v_2 \neq 0) \ \wedge \neg \ \Phi_{\mathbf{I}}(t_1, x_1, y_1, t_2, x_2, y_2, v_1, t_3, x_3, y_3, t_4, x_4, y_4, v_2) \ \wedge$$
$$(\psi_{\mathsf{IC}^-}(t_1, x_1, y_1, t_2, x_2, y_2, v_1, t_3, x_3, y_3, t_4, x_4, y_4, v_2) \ \vee \ \psi_{\mathsf{IC}^+}(t_1, x_1, y_1, t_2, x_2, y_2, v_1, t_3, x_3, y_3, t_4, x_4, y_4, v_2)).$$

## 5.6 *The formula for the parametric alibi query*

The final formula that decides if two space-time prisms, $\mathcal{P}_1 = \mathcal{P}(t_1, x_1, y_1, t_2, x_2, y_2, v_1)$ and $\mathcal{P}_2 = \mathcal{P}(t_3, x_3, y_3, t_4, x_4, y_4, v_2)$, do not intersect looks as follows

$$\psi_{alibi}(t_1, x_1, y_1, t_2, x_2, y_2, v_1, t_3, x_3, y_3, t_4, x_4, y_4, v_2) := \neg((t_1 < t_2 \ \wedge \ t_3 < t_4) \ \wedge$$
$$(\Phi_{\mathbf{III}}(t_1, x_1, y_1, t_2, x_2, y_2, v_1, t_3, x_3, y_3, t_4, x_4, y_4, v_2)$$
$$\vee \ \Phi_{\mathbf{II}}(t_1, x_1, y_1, t_2, x_2, y_2, v_1, t_3, x_3, y_3, t_4, x_4, y_4, v_2))$$
$$\vee \ \Phi_{\mathbf{I}}(t_1, x_1, y_1, t_2, x_2, y_2, v_1, t_3, x_3, y_3, t_4, x_4, y_4, v_2)).$$

## 6 Experiments

In this section, we compare our solution to the alibi query (using the formula given in Section 5.6) with the method of eliminating quantifiers of Mathematica.

In the following table it is clear that traditional quantifier elimination performs badly on the example space-time prisms. Its running times highly deviates from their average and range in the minutes. Whereas the method described in this paper performs in running times that consistently only needs milliseconds or less. This shows our method is efficient and our claim, that it runs in milliseconds or less, holds.

For this first set of space-time prisms we chose to verify intersection of two oblique space-time prisms (1-2) and the intersection of one oblique and one straight space-time prism (3-4). The space-time prisms that actually intersected had a remarkable low running time with the QE-method.

| | The space-time prisms | | The running times | |
|---|---|---|---|---|
| | $\mathcal{P}_1$ | $\mathcal{P}_2$ | QE | Our Method |
| 1 | $(0, 0, 0, 2, 0, 2, 1.9)$ | $(0, 3, 0, 2, 3, 2, 2)$ | 0.656 Seconds | 0.016 Seconds |
| 2 | $(0, 0, 0, 2, 0, 2, 1.9)$ | $(0, 4, 0, 2, 4, 2, 2)$ | 324.453 Seconds | 0.063 Seconds |
| 3 | $(0, 0, 0, 2, 0, 2, 1.9)$ | $(0, 3, 0, 2, 3, 0, 2)$ | 0.438 Seconds | 0.015 Seconds |
| 4 | $(0, 0, 0, 2, 0, 2, 1.9)$ | $(0, 4, 0, 2, 4, 0, 2)$ | 475.719 Seconds | 0.031 Seconds |

The type of space-time prisms in this second set are as in the first. However, these space-time prisms all have overlapping time intervals unlike the first set, where the time intervals coincided.

| | The space-time prisms | | The running times | |
|---|---|---|---|---|
| | $\mathcal{P}_1$ | $\mathcal{P}_2$ | QE | Our Method |
| 1 | $(0, 0, 0, 2, 0, 2, 1.9)$ | $(1, 3, 0, 3, 3, 2, 2)$ | 63.375 Seconds | 0.078 Seconds |
| 2 | $(0, 0, 0, 2, 0, 2, 1.9)$ | $(1, 4, 0, 3, 4, 2, 2)$ | 59.485 Seconds | 0.078 Seconds |
| 3 | $(0, 0, 0, 2, 0, 2, 1.9)$ | $(1, 3, 0, 3, 3, 0, 2)$ | 29.734 Seconds | 0.031 Seconds |
| 4 | $(0, 0, 0, 2, 0, 2, 1.9)$ | $(1, 4, 0, 3, 4, 0, 2)$ | 27.281 Seconds | 0.032 Seconds |

The type of space-time prisms in this third set are as in the first. But this time the time intervals are completely disjoint. Note that the running times for the QE-method are more consistent in this set and the previous one.

| | The space-time prisms | | The running times | |
|---|---|---|---|---|
| | $\mathcal{P}_1$ | $\mathcal{P}_2$ | QE | Our Method |
| 1 | $(0, 0, 0, 2, 0, 2, 1.9)$ | $(3, 3, 0, 4, 3, 2, 2)$ | 63.641 Seconds | 0.046 Seconds |
| 2 | $(0, 0, 0, 2, 0, 2, 1.9)$ | $(3, 4, 0, 4, 4, 2, 2)$ | 61.781 Seconds | 0.016 Seconds |
| 3 | $(0, 0, 0, 2, 0, 2, 1.9)$ | $(3, 3, 0, 4, 3, 0, 2)$ | 52.735 Seconds | 0.031 Seconds |
| 4 | $(0, 0, 0, 2, 0, 2, 1.9)$ | $(3, 4, 0, 4, 4, 0, 2)$ | 56.875 Seconds | 0.046 Seconds |

After conducting numerous experiments, we have to conclude that there is no consistent trend in the running times of the traditional method. The only trend that showed is that the time the traditional quantifier elimination method takes is several orders of magnitude higher than our own method and that our method runs consistently under a tenth of a second.

## 7 The alibi query at a fixed moment in time

### 7.1 *Introduction*

In this section, we present another example where common sense prevails over the general quantifier-elimination methods. The problem is the following. As in the previous setting, we have lists of time stamped-locations of two moving objects and upper bounds on the object's speed between time stamps. We wish to know if two objects could have met at a given moment in time.

For the remainder of this section, we reuse the assumptions from the previous section. We wish to verify if the space-time prisms $\mathcal{P}_1 = \mathcal{P}(t_1, x_1, y_1, t_2, x_2, y_2, v_1)$ and $\mathcal{P}_2 = \mathcal{P}(t_3, x_3, y_3, t_4, x_4, y_4, v_2)$ intersect at a moment in time $t_0$. Moreover, we assume the space-time prisms are non-empty, i.e., $(x_2-x_1)^2+(y_2-y_1)^2 \leq (t_2-t_1)^2v_1^2$ and $(x_4-x_3)^2+(y_4-y_3)^2 \leq (t_4-t_3)^2v_2^2$ and that $t_1 \leq t_0 \leq t_2$ and $t_3 \leq t_0 \leq t_4$ are satisfied. This means we need to eliminate the quantifiers in

$$\exists x \exists y \left((x-x_1)^2+(y-y_1)^2 \leq v_1^2(t_0-t_1)^2 \ \wedge\ (x-x_2)^2+(y-y_2)^2 \leq v_1^2(t_0-t_2)^2 \right.$$
$$\left. \wedge\ (x-x_3)^2+(y-y_3)^2 \leq v_2^2(t_0-t_3)^2 \ \wedge\ (x-x_4)^2+(y-y_4)^2 \leq v_2^2(t_0-t_4)^2\right).$$

Eliminating quantifiers gives us a formula that decide whether or not four discs have a non-empty intersection. For ease of notation we will use the following abbreviations: $(x, y) \in D_i$ if and only if $(x-x_i)^2 + (y-y_i) \leq r_i^2$ and $(x, y) \in C_i$ if and only if $(x-x_i)^2 + (y-y_i) = r_i^2$.

### 7.2 *Main theorem*

Using Helly's theorem we can simplify the problem even more. Helly's theorem states that if you have a set $S$ of $m$ convex sets in $n$ dimensional space and if any subset of $S$ of $n+1$ convex sets has a non-empty intersection, then all $m$ convex sets have a non-empty intersection. For the plane, this means we only need to find a quantifier free-formula that decides if three discs have a non-empty intersection. For the remainder of this section assume that we want to verify whether $D_1 \cap D_2 \cap D_3$ is non-empty.

Theorem 7.1 *Three discs, $D_1$, $D_2$ and $D_3$, have a non-empty intersection if and only if one of the following cases occur:*

*(i) there is a disc whose center is in the other two discs; or*
*(ii) the previous case does not occur and there exists a pair of discs for which one of both intersection points of their bordering circles lies in the remaining disc.*

*Proof* The *if*-direction is trivial. The *only if*-direction is less trivial. We will use the following abbreviations, $D = D_1 \cap D_2 \cap D_3$ and $C = \partial D$.

Assume $D$ is non-empty and that neither (1) nor (2) holds. The intersection $D$ is convex as it is the intersection of convex sets. We distinguish between the case where $D$ is a point or and the case where $D$ is not a point.

- Suppose $D$ is a single point $p$. This point $p$ can not lie in the interior of the three discs, because $D$ would not be a point then.

  Nor can $p$ lie in the interior of two discs. If that would be the case then there exists a neighborhood of $p$ that is part of the intersection of those two discs, say $D_1$ and $D_2$. Moreover $p$ would be part of $C_3$ and this neighborhood would intersect the interior of $D_3$. This means $D$ is not a point.

  So $p$ must lie on the border of two discs, say $D_1$ and $D_2$, and $p$ must also be part of $D_3$ because $D = \{p\}$. This contradicts our assumption that (2) does not hold.

- Assume $D$ is not a point. All points on $C$ belong to at least one $C_i$. If there is a point that does not belong to any $C_i$, then it is in the interior of all $D_i$ and there exists a neighborhood of that point that is in the interior of all $D_i$ and hence in $D$. That contradicts to the fact that this point is in $C$.

  Furthermore, not all points of $C$ belong to a single $C_i$. If that was the case then $D_i$ would be part of (and equal to) $D$ and its center would be inside the other two discs which contradicts the assumption that (1) does not hold.

  So, $C$ is made up of parts of the $C_i$, of which some may coincide but not all of them. When traveling along $C$ you will encounter a point $p$ that connects part of a $C_i$ and part of a $C_j$, where $i \neq j$, that do not coincide, otherwise (1) must occur again which is a contradiction. However, this $p$ also yields to a contradiction since it belongs to two different $C_i$, say $C_1$ and $C_2$, and is part of $C$ hence $D$ and $D_3$. This contradicts the assumption that (2) does not occur.

$\square$

### 7.3 *Translating the theorem in a formula*

We can simplify the equations even further using coordinate transformations. By applying a translation, rotation and scaling we may assume that $(x_1, y_1) = (0,0)$, $x_2 \geq 0$, $r_1 = 1$ and $y_2 = 0$. Using these simplifications and translating Theorem 7.1, we get the following formula.

$$\begin{aligned}\Psi_1(x_2, r_2, x_3, y_3, r_3) := \big((-x_2)^2 \leq r_2^2 \ \wedge\ (-x_3)^2 + (-y_3)^2 \leq r_3^2\big) \ \vee \\ \exists x \exists y \left(x^2 + y^2 = 1 \ \wedge\ (x - x_2)^2 + y^2 = r_2^2 \ \wedge\ (x - x_3)^2 + (y - y_3)^2 \leq r_3^2\right) \ .\end{aligned}$$

This is a formula that decides if either the center of the first disc is part of the two other discs, see the first line, or if either there exists a point in the intersection of the first two circles that is part of the third disc. All that remains now is making the expression

$$\exists x \exists y \left(x^2 + y^2 = 1 \ \wedge\ (x - x_2)^2 + y^2 = r_2^2 \ \wedge\ (x - x_3)^2 + (y - y_3)^2 \leq r_3^2\right)$$

quantifier free.

To do this we assume that $C_1$ and $C_2$ do not coincide but have a non-empty intersection. This is equivalent to $x_2 \neq 0 \ \wedge\ x_2 \leq r_2 + 1$. Next, we need to compute the point(s) where $C_1$ and $C_2$ intersect.

$$\begin{cases} x^2 + y^2 = 1 \\ (x - x_2)^2 + y^2 = r_2^2 \end{cases} \text{or} \begin{cases} x = \frac{x_2^2+1-r_2^2}{2x_2} \\ y = \pm\sqrt{1 - \left(\frac{x_2^2+1-r_2^2}{2x_2}\right)^2} \end{cases}$$

$$\text{or} \begin{cases} x = \frac{x_2^2+1-r_2^2}{2x_2} \\ y = \pm\sqrt{\left(1 - \frac{x_2^2+1-r_2^2}{2x_2}\right)\left(1 + \frac{x_2^2+1-r_2^2}{2x_2}\right)} \end{cases}$$

$$\text{or} \begin{cases} x = \frac{x_2^2+1-r_2^2}{2x_2} \\ y = \pm\frac{1}{2x_2}\sqrt{\left(r_2^2 - (x_2-1)^2\right)\left((1+x_2)^2 - r_2^2\right)} \ . \end{cases}$$

If $y = 0$, then verifying if that single point of intersection is part of $D_3$ is easy, one only needs to verify if

$$\left(\frac{x_2^2 + 1 - r_2^2}{2x_2} - x_2\right)^2 + y_3^2 \leq r_3^2$$

$$\text{or equivalently } \left(1 - r_2^2 - x_2^2\right)^2 + 4x_2^2y_3^2 \le 4x_2^2r_3^2 \ .$$

If $y \neq 0$, then verifying if one both points of intersection is part of $D_3$ is less trivial, since this involves square roots

$$\left(\frac{x_2^2+1-r_2^2}{2x_2} - x_3\right)^2 + \left(\pm\frac{1}{2x_2}\sqrt{\left(r_2^2-(x_2-1)^2\right)\left((1+x_2)^2-r_2^2\right)} - y_3\right)^2 \le r_3^2$$

$$\text{or } \left(x_2^2+1-r_2^2-2x_2x_3\right)^2 + \left(\pm\sqrt{\left(r_2^2-(x_2-1)^2\right)\left((1+x_2)^2-r_2^2\right)} - 2x_2y_3\right)^2 \le 4x_2^2r_3^2$$

$$\begin{aligned}\text{or } &\left(x_2^2+1-r_2^2-2x_2x_3\right)^2 + \left(r_2^2-(x_2-1)^2\right)\left((1+x_2)^2-r_2^2\right)\\ &+(2x_2y_3)^2 \pm 4x_2y_3\sqrt{\left(r_2^2-(x_2-1)^2\right)\left((1+x_2)^2-r_2^2\right)} \le 4x_2^2r_3^2\end{aligned}$$

$$\begin{aligned}\text{or } &\left(x_2^2+1-r_2^2-2x_2x_3\right)^2 + \left(r_2^2-(x_2-1)^2\right)\left((1+x_2)^2-r_2^2\right)\\ &+(2x_2y_3)^2 - 4x_2^2r_3^2 \le \pm 4x_2y_3\sqrt{\left(r_2^2-(x_2-1)^2\right)\left((1+x_2)^2-r_2^2\right)}.\end{aligned}$$

This is almost a $\mathsf{FO}(+,\times,<,0,1)$-formula except for the square root. However, the square root can be eliminated as we will show next. The previous expression is of the form $L \le \pm a\sqrt{W}$. The presence of the $\pm$ simplifies this a lot, this means either sign of the square root will do, and also that we may assume the right hand-side is positive. Of course the square root must exist as well, this means $W \ge 0$.

This expression can then be simplified to

$$W \ge 0 \ \wedge \ \left(L \le 0 \ \vee \ L^2 \le a^2W\right)$$

and gives us the expression

$$\begin{aligned}\Phi_2(x_2,r_2,x_3,y_3,r_3) := &\left(r_2^2-(x_2-1)^2\right)\left((1+x_2)^2-r_2^2\right) \ge 0\\ \wedge\ &\Big(\left(x_2^2+1-r_2^2-2x_2x_3\right)^2 + \left(r_2^2-(x_2-1)^2\right)\left((1+x_2)^2-r_2^2\right) + (2x_2y_3)^2 - 4x_2^2r_3^2 \le 0\\ \vee\ &\Big(\left(x_2^2+1-r_2^2-2x_2x_3\right)^2 + \left(r_2^2-(x_2-1)^2\right)\left((1+x_2)^2-r_2^2\right) + (2x_2y_3)^2 - 4x_2^2r_3^2\Big)^2\\ &\le (4x_2y_3)^2\left(r_2^2-(x_2-1)^2\right)\left((1+x_2)^2-r_2^2\right)\Big).\end{aligned}$$

## 7.4 *The safety formula*

Now, all that remains to be constructed is a formula that returns the convenient coordinates and a formula that guarantees that $C_1$ and $C_2$ actually intersect for safety, i.e., to exclude the case of empty intersection. The latter is constructed as follows. The formula $\varphi(x_1,y_1,r_1,x_2,y_2,r_2)$ returns TRUE if and only if the two circles, with centers $(x_1,y_1)$ and $(x_2,y_2)$ and radii $r_1$ and $r_2$ respectively, have a distance between their centers that is not larger that the sum of their radii and not equal to zero to ensure they do not coincide. We have

$$\varphi(x_1, y_1, r_1, x_2, y_2, r_2) := 0 < (x_2 - x_1)^2 + (y_2 - y_1)^2 \leq (r_1 + r_2)^2 \ .$$

The formula $\phi(x_1, y_1, r_1, x_2, y_2, r_2)$ returns TRUE if and only if the second circle is not fully enclosed by the first, i.e., the sum of the distance between the centers plus the second radius is bigger than the first radius and vice versa. We can write

$$\phi(x_1, y_1, r_1, x_2, y_2, r_2) := (x_2 - x_1)^2 + (y_2 - y_1)^2 \geq (r_1 - r_2)^2 \ .$$

These two safety conditions give us our safety formula

$$\Phi_{\text{safety}}(x_1, y_1, r_1, x_2, y_2, r_2) := \varphi(x_1, y_1, r_1, x_2, y_2, r_2) \ \wedge \ \phi(x_1, y_1, r_1, x_2, y_2, r_2) \ .$$

## 7.5 *The change of coordinates*

The transformation consists of a translation, rotation and scaling. The translation to move the first circle's center to the origin. The rotation to align the second center with the $x$-axis. Finally the scaling to ensure that the first circle's radius is equal to one. First, the translation $T(x, y) := (x - x_1, y - y_1)$. The rotation is

$$R(x, y) := \frac{1}{\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}} \begin{pmatrix} x_2 - x_1 & y_2 - y_1 \\ y_1 - y_2 & x_2 - x_1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

and finally, the scaling is $S(x, y) := \frac{1}{r_1}(x, y)$. The transformation is then a composition of those three transformations $A(x, y) = (S \circ R \circ T)(x, y) :=$

$$\left( \frac{(x_2 - x_1)(x - x_1) + (y_2 - y_1)(y - y_1)}{r_1\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}}, \frac{(y_1 - y_2)(x - x_1) + (x_2 - x_1)(y - y_1)}{r_1\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}} \right) \ .$$

The following formula takes three circles with centers $(x_i, y_i)$ and radii $r_i$ respectively, and transforms them in three new circles where the first circle has center $(0, 0)$ and radius 1, the second circle has center $(x'_2, 0)$ and radius $r'_2$ and the third circle has center $(x'_3, y'_3)$ and radius $r'_3$.

$$\Phi_{\text{transformation}}(x_1, y_1, r_1, x_2, y_2, r_2, x_3, y_3, r_3, x'_2, r'_2, x'_3, y'_3, r'_3) := \ x'_2 = \frac{\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}}{r_1} \wedge r'_2 = \frac{r_2}{r_1}$$
$$\wedge\, r'_3 = \frac{r_3}{r_1} \wedge x'_3 = \frac{(x_2 - x_1)(x_3 - x_1) + (y_2 - y_1)(y_3 - y_1)}{r_1\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}} \wedge y'_3 = \frac{(y_1 - y_2)(x_3 - x_1) + (x_2 - x_1)(y_3 - y_1)}{r_1\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}} \ .$$

Note that this is not a FO$(+, \times, <, 0, 1)$-formula anymore due to the square roots and fractions. This "formula" is meant to act like a function, which substitutes coordinates. The substituted coordinates have fractions and square roots but these can easily be disposed of when having the entire inequality on a common denominator, isolating the square root and squaring the inequality, as we showed in Section 7.3.

### 7.6 *The formula for the alibi query at a fixed moment in time*

First, we construct a formula that checks for any of two circles out of three if any of the conditions in Theorem 7.1 are satisfied.

$$\begin{aligned}
&\Psi_{2/3}(x_1,y_1,r_1,x_2,y_2,r_2,x_3,y_3,r_3) \ := \\
&\qquad \exists x_2'\exists r_2'\exists x_3'\exists y_3'\exists r_3' \ \big(\Phi_{\text{transformation}}(x_1,y_1,r_1,x_2,y_2,r_2,x_3,y_3,r_3,x_2',r_2',x_3',y_3',r_3') \wedge \\
&\qquad \Big(\Psi_1(x_2',r_2',x_3',y_3',r_3') \ \vee \ \Phi_{\text{safety}}(x_1,y_1,r_1,x_2,y_2,r_2) \ \wedge \ \Phi_2(x_2',r_2',x_3',y_3',r_3')\Big) \\
&\vee \qquad \Phi_{\text{transformation}}(x_1,y_1,r_1,x_3,y_3,r_3,x_2,y_2,r_2,x_2',r_2',x_3',y_3',r_3') \wedge \\
&\qquad \Big(\Psi_1(x_2',r_2',x_3',y_3',r_3') \ \vee \ \Phi_{\text{safety}}(x_1,y_1,r_1,x_3,y_3,r_3) \ \wedge \ \Phi_2(x_2',r_2',x_3',y_3',r_3')\Big) \\
&\vee \qquad \Phi_{\text{transformation}}(x_2,y_2,r_2,x_3,y_3,r_3,x_1,y_1,r_1,x_2',r_2',x_3',y_3',r_3') \wedge \\
&\qquad \Big(\Psi_1(x_2',r_2',x_3',y_3',r_3') \ \vee \ \Phi_{\text{safety}}(x_2,y_2,r_2,x_3,y_3,r_3) \ \wedge \ \Phi_2(x_2',r_2',x_3',y_3',r_3')\Big)\Big).
\end{aligned}$$

This formula is all we need to incorporate Helly's theorem in our final formula. Four discs have a non-empty intersection if and only if the following formula is satisfied

$$\begin{aligned}
&\Psi(x_1,y_1,r_1,x_2,y_2,r_2,x_3,y_3,r_3,x_4,y_4,r_4) := \\
&\qquad \Psi_{2/3}(x_1,y_1,r_1,x_2,y_2,r_2,x_3,y_3,r_3) \ \wedge \ \Psi_{2/3}(x_1,y_1,r_1,x_2,y_2,r_2,x_4,y_4,r_4) \\
&\qquad \wedge \ \Psi_{2/3}(x_1,y_1,r_1,x_3,y_3,r_3,x_4,y_4,r_4) \ \wedge \ \Psi_{2/3}(x_2,y_2,r_2,x_3,y_3,r_3,x_4,y_4,r_4).
\end{aligned}$$

This is almost a quantifier free-formula except for the fractions and square roots. However, as we showed before these can easily be disposed of. We omitted these tedious conversions for the sake of clarity.

### 7.7 *Notes on experiments*

First we remark that the formula above is a lot shorter than the formula $\psi_{alibi}$, so it is reasonable to assume that this formula evaluates even faster than $\psi_{alibi}$. Before we started looking for a $\mathsf{FO}(+,\times,<,0,1)$-formula to answer the alibi query, we tried several software packages, that support quantifier elimination, to produce a formula for us. In both cases, the general alibi query and the alibi query for a fixed moment in time, the quantifier elimination methods failed to produce an answer at all after prolonged sessions. This was expected and, at the same time, a motivation to find such a formula ourselves.

For the non-parametric case however, Mathematica returns an answer in a matter of hundreds of seconds. In this case, the non-parametric alibi query for a fixed moment in time, the general methods do produce an answer efficiently. The relative differences were inconsistent, the same parameters would take, for example, .014 and .031 seconds, depending on the other processes the system was running and which we could not control. For this reason we decided not to include experimental results in this case.

## 8 Conclusion

In this paper, we proposed a method that decides if two space-time prisms have a non-empty intersection or not. Existing quantifier-elimination methods could achieve this already through means of quantifier elimination though not in a reasonable amount of time. Deciding intersection of concrete space-time prisms took of the order of minutes, while the parametric case could be measured at least in days if a solution would ever be obtained. The parametric solution we laid out in this paper only takes a few milliseconds or less.

The solution we present is a first-order formula containing square root-expressions. These can easily be disposed of using repeated squarings and adding extra conditions, thus obtaining a true quantifier free-expression for the alibi query.

We also give a solution to the alibi query at a fixed moment in time.

The solutions we propose are based on geometric argumentation and they illustrate the fact that some practical problems require creative solutions, where at least in theory, existing systems could provide a solution.

## Acknowledgements

This research has been partially funded by the European Union under the FP6-IST-FET programme, Project n. FP6-14915, GeoPKDD: Geographic Privacy-Aware Knowledge Discovery and Delivery, and by the Research Foundation Flanders (FWO-Vlaanderen), Research Project G.0344.05.

## References

Basu, S., R., P., and Roy, M.-F. 1996. On the combinatorial and algebraic complexity of quantifier elimination. *Journal of the ACM 43*, 1002–1045.

Collins, G. 1975. Quantifier elimination for real closed fields by cylindrical algebraic decomposition. In *Automata Theory and Formal Languages*. Lecture Notes in Computer Science, vol. 33. Springer-Verlag, 134–183.

Egenhofer, M. 2003. Approximation of geopatial lifelines. In *SpadaGIS, Workshop on Spatial Data and Geographic Information Systsems*. University of Genova. Electr. proceedings, 4p.

Geerts, F. 2004. Moving objects and their equations of motion. In *Constraint Databases, Proceedings of the 1st International Symposium on Applications of Constraint Databases, (CDB'04)*. Lecture Notes in Computer Science, vol. 3074. Springer, 41–52.

Grigor'ev, D. and Vorobjov, N. N. j. 1988. Solving systems of polynomial inequalities in subexponential time. *Journal of Symbolic Computation 5*, 37–64.

Güting, R. and Schneider, M. 2005. *Moving Object Databases*. Morgan Kaufmann.

Hägerstrand, T. 1970. What about people in regional science? *Papers of the Regional Science Association 24*, 7–21.

Heintz, J. and Kuijpers, B. 2004. Constraint databases, data structures and efficient query evaluation. In *Constraint Databases, Proceedings of the 1st International Symposium "Applications of Constraint Databases" (CDB'04)*. Lecture Notes in Computer Science, vol. 3074. Springer-Verlag, 1–24.

Heintz, J., Roy, M.-F., and Solernó, P. 1990. Sur la complexité du principe de Tarski-Seidenberg. *Bulletin de la Société Mathématique de France 118*, 101–126.

Hong, H. 1990. QEPCAD — quantifier elimination by partial cylindrical algebraic decomposition. http://www.cs.usna.edu/~qepcad/B/QEPCAD.html.

Hornsby, K. and Egenhofer, M. 2002. Modeling moving objects over multiple granularities. *Annals of Mathematics and Artificial Intelligence 36,* 1–2, 177–194.

Kanellakis, P., Kuper, G., and Revesz, P. 1995. Constraint query languages. *Journal of Computer and System Science 51,* 1, 26–52. A preliminary report appeared in the *Proceedings 9th ACM Symposium on Principles of Database Systems (PODS'90)*.

Kuijpers, B. and Othman, W. 2007. Trajectory databases: Data models, uncertainty and complete query languages. In *Proceedings of the 11th International Conference on Database Theory (ICDT'07)*. Lecture Notes in Computer Science, vol. 4353. Springer-Verlag, 224–238.

Miller, H. 2005. A measurement theory for time geography. *Geographical Analysis 37,* 1, 17–.

Paredaens, J., Kuper, G., and Libkin, L. 2000. *Constraint databases*. Springer-Verlag.

Paredaens, J., Van den Bussche, J., and Van Gucht, D. 1994. Towards a theory of spatial database queries. In *Proceedings of the 13th ACM Symposium on Principles of Database Systems (PODS'94)*. ACM Press, New York, 279–288.

Pfoser, D. and Jensen, C. S. 1999. Capturing the uncertainty of moving-object representations. In *Advances in Spatial Databases (SSD'99)*. Lecture Notes in Computer Science, vol. 1651. 111–132.

Renegar, J. 1992. On the computational complexity and geometry of the first-order theory of the reals I, II, III. *Jornal of Symbolic Computation 13*, 255–352.

Revesz, P. 2002. *Introduction to Constraint Databases*. Springer-Verlag.

Sturm, T. 2000. Redlog. http://www.algebra.fim.uni-passau.de/˜redlog/.

Su, J., Xu, H., and Ibarra, O. 2001. Moving objects: Logical relationships and queries. In *Advances in Spatial and Temporal Databases (SSTD'01)*. Lecture Notes in Computer Science, vol. 2121. Springer, 3–19.

Tarski, A. 1951. *A Decision Method for Elementary Algebra and Geometry*. University of California Press.

Wolfram. 2007. Mathematica 6. http://www.wolfram.com.

Wolfson, O. 2002. Moving objects information management: The database challenge. In *Proceedings of the 5th Intl. Workshop NGITS*. Springer, 75–89.