Towards a Theory of Movie Database Queries

Bart Kuijpers
University of Limburg (LUC)
Department WNI
B-3590 Diepenbeek, Belgium
bart.kuijpers@luc.ac.be

Jan Paredaens
University of Antwerp (UIA)
Dept. of Math. & Computer Science
Universiteitsplein 1
B-2610 Antwerpen, Belgium
pareda@uia.ua.ac.be

Dirk Van Gucht
Indiana University
Computer Science Dept.
Bloomington, IN 47405-4101, USA
vgucht@cs.indiana.edu

Abstract

We present a data model for movies and movie databases. A movie is considered to be a 2-dimensional semi-algebraic figure that can change in time. We give a number of computability results concerning movies: it can be decided whether a frame of a movie is only a topologically transformation of another frame; a movie has a finite number of scenes and cuts and these can be effectively computed.

Based on these computability results we define an SQL-like query language for movie databases. This query language supports most movie editing operations like cutting, pasting and selection of scenes.

1. Introduction

We present a data model for movies and movie databases. We consider a movie to be an infinite sequence of 2-dimensional figures that evolve in time. Each figure consists of a possibly infinite number of points in the 2-dimensional plane. A recent and much acclaimed method for effectively representing infinite geometrical figures is provided by the *constraint database model*, that was introduced by Kanellakis, Kuper and Revesz in their 1990 seminal paper [10] (an overview of the area of constraint databases can be found in [13]). In this model, a 2-dimensional geometrical figure is finitely represented by means of a Boolean combination of polynomial equalities and inequalities. These involve polynomials with two real variables that represent the spatial coordinates of a point in the plane. The set of points on the upper half of the unit circle, for instance, is in this context given by

$$\{(x, y) \in \mathbf{R}^2 \mid x^2 + y^2 = 1 \wedge y \geq 0\}.$$

In more mathematical terminology, these figures are called *semi-algebraic sets* and for an overview of their properties we refer to [3, 6].

This way of representing fixed figures can easily be adapted to describe figures that change. Indeed, we can add a time dimension and consider geometrical objects in 3-dimensional space-time that are described by polynomial equalities and inequalities that also have a time variable t. This gives us a data model for *movies*. Figure 1 gives an example of a movie, in particular a potential scene from Star Trek. In this short movie the starship Enterprise remains at a constant position in space and can therefore be described by formula $\varphi_{\text{Enterprise}}(x, y, t) = (x^2 + y^2 = 1 \vee x^2 + y^2 = (1/4)^2 \vee \cdots)$ in which t is lacking. A fired photon torpedo follows the dotted line (between the moments $t = 0$ and $t = 1$) an then explodes (depicted as increasing dotted circles, between $t = 1$ and $t = 2$). At the bottom of Figure 1 three frames of the movie are shown: at $t = 1/2, 1$ and 2. The complete movie can be described by the set

$$\begin{aligned} \{(x, y; t) \in \mathbf{R}^2 \times \mathbf{R} \mid (\varphi_{\text{Enterprise}}(x, y) \wedge 0 \leq t \leq 2) \vee \\ \big((y = 0 \wedge x = 4t \wedge 0 \leq t \leq 1) \vee \\ ((x - 4)^2 + y^2 \leq (t - 1) \wedge 1 < t \leq 2)\big)\}. \end{aligned}$$

The movie of Figure 1 can be used to illustrate a number of properties that all movies in this model have in common.

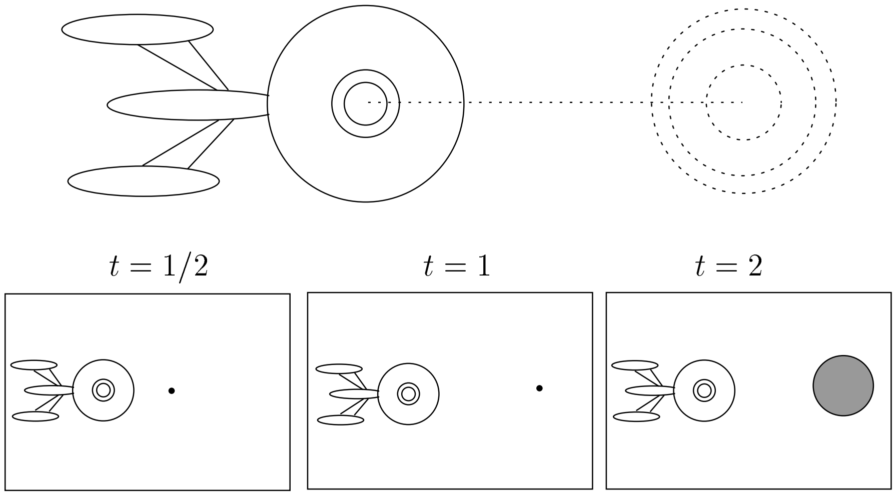

Figure 1. USS Enterprise firing a photon torpedo at a (cloaked) Klingon vessel.

For instance, between $t = 1$ and $t = 2$ the movie frames change continuously and all frames ($1 < t \leq 2$) are, topologically seen, the same. We will call such a sequence of frames a *scene* of the movie. Moments in which the movie changes discontinuously are referred to as *cuts* in the movie. In the movie of Figure 1 there is, for instance, a cut at $t = 1$: a point changes into an increasing circle. The movie of Figure 1 has five scenes and six cuts (start and end of the movie included). We remark that our notion of scene is finer that the cinematographic notion of scene.

We will show that the number of cuts and scenes in a semi-algebraic movie is always finite and that a representation of them by means of polynomial constraints can be effectively computed. A key ingredient in this computation is a decision procedure for testing whether two movie frames can be topologically transformed into one another, i.e., whether they are homeomorphic. Although deciding whether two 2-dimensional semi-algebraic sets are homeomorphic is a result that belongs to the mathematical folklore, a written proof of it is not to be found in the mathematical literature [15, 18]. We give a decision procedure and we also generalize it to parameterized frames: there is an algorithm that, given two movie frames that depend on time parameters t_1 and t_2, produces a formula built with conjunction and disjunction from formulas of the form $a < t_i < b$ and $t_i = c$ (a, b, c constants and $i = 1, 2$) that expresses, in function of t_1 and t_2, whether the frames are homeomorphic.

Finally, we define an SQL-like language to query movie databases. This language is based on the above computability results and on a well-known language to query databases in the constraint model, namely the relational calculus augmented with polynomial inequalities [10, 16, 13]. It follows from a result by Tarski that the latter language is also effective [20] (although variables range over the real continuum). Our query language supports all basic movie editing operations like selecting scenes that satisfy some condition, composing several scenes into a movie, removing scenes, etc. It also allows for the manipulation of single scenes and even of single frames.

This paper is organized as follows. In Section 2, we formally define the notion of movie, frame, scene and cut. Procedures to decide homeomorphism of frames and to compute the scenes and cuts of a movie are given in Section 3. In Section 4, we present a query language and discuss expressibility issues.

2. Movies, Frames, Scenes and Cuts

We denote the set of the real numbers by $\mathbf{R}$. In the following we will consider planar figures that change in time. A moving figure is described by means of an (often infinite) set of tuples $(x, y; t)$ in $\mathbf{R}^2 \times \mathbf{R}$, where x and y represent the spatial coordinates of a point in the 2-dimensional real plane $\mathbf{R}^2$ and t represents the time coordinate in $\mathbf{R}$. We first define the notion of a movie.

Definition 2.1 A *movie* is a set

$$\mathcal{M} = \{(x, y; t) \in \mathbf{R}^2 \times \mathbf{R} \mid a \leq t \leq b \wedge \varphi(x, y, t)\},$$

where a and b are real algebraic numbers and where $\varphi(x, y, t)$ is a formula built with the logical connectives $\wedge, \vee, \neg$ from atomic formulas of the form $p(x, y, t) > 0$, with $p(x, y, t)$ a polynomial with real algebraic coefficients and real variables x, y, t. The numbers a and b are called the *beginning* and *end* of the movie respectively and are denoted by $\mathbf{Begin}(\mathcal{M})$ and $\mathbf{End}(\mathcal{M})$.

A *movie database* is a finite set of movies. $\square$

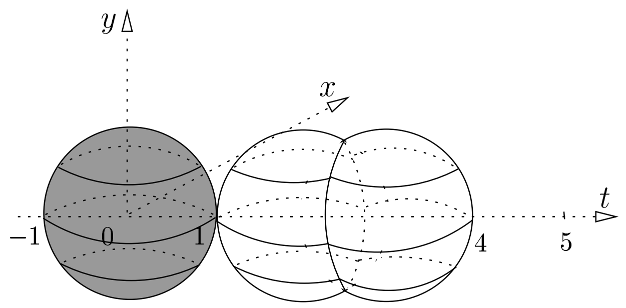

Figure 2. An example of a movie.

Figure 2 depicts the movie $\{(x, y; t) \in \mathbf{R}^2 \times \mathbf{R} \mid -1 \leq t \leq 5 \wedge \big(x^2 + y^2 + t^2 \leq 1 \vee (x^2 + y^2 + (t-2)^2 = 1 \wedge t \leq 5/2) \vee (x^2 + y^2 + (t-3)^2 = 1 \wedge t > 5/2)\big)\}$ in the space $\mathbf{R}^2 \times \mathbf{R}$. This movie shows at its beginning (i.e., at $t = -1$) a single point in the origin. Then it shows a disk whose radius increases and later decreases and ends in a point at moment $t = 1$, followed by a circle whose radius increases, decreases, increases and then shrinks to a point. Finally, for $4 < t \leq 5$, this movie shows nothing.

Definition 2.2 Let $\mathcal{M}$ be the movie $\{(x, y; t) \in \mathbf{R}^2 \times \mathbf{R} \mid a \leq t \leq b \wedge \varphi(x, y, t)\}$ and let $a \leq t_0 \leq b$. The set $\{(x, y) \in \mathbf{R}^2 \mid \varphi(x, y, t_0)\}$ is called the *frame of the movie* $\mathcal{M}$ *at the moment* t_0 and is denoted by $\mathcal{M}^{t_0}$. □

For the movie of Figure 2, for instance, the frame $\mathcal{M}^{-1}$ is the origin, $\mathcal{M}^0$ is the closed unit disk and $\mathcal{M}^5$ is the empty set.

We can use this same example to illustrate the notion of a scene. For $-1 < t < 1$, the movie of Figure 2 shows a disk on which is zoomed into and then zoomed out of. This continuous sequence of frames will be called a *scene*. Also for $1 < t < 4$, we have a scene in which a circle is continuously deformed. Scenes are separated by *cuts*.

In the following definition these concepts are formalized. The notion of continuity that we will give may seem rather involved. It corresponds, however to the intuitive notion of "continuously changing," as illustrated by the above example.

Definition 2.3 Let $\mathcal{M} = \{(x, y; t) \in \mathbf{R}^2 \times \mathbf{R} \mid a \leq t \leq b \wedge \varphi(x, y, t)\}$ be a movie and let $a \leq t_0 \leq b$.

- $\mathcal{M}$ is *continuous on the right in* t_0 if there exists an $\varepsilon > 0$ and a continuous (in t) series $(h_t : \mathbf{R}^2 \to \mathbf{R}^2 : (x, y) \mapsto h_t(x, y) \mid t_0 \leq t \leq t_0 + \varepsilon)$ of homeomorphisms of $\mathbf{R}^2$ such that $h_t(\mathcal{M}^{t_0}) = \mathcal{M}^t$ for all $t \in [t_0, t_0 + \varepsilon]$. $\mathcal{M}$ is *continuous on the left in* t_0 is defined similarly. And $\mathcal{M}$ is *continuous in* t_0 if it is continuous both on the right and on the left in t_0.
- A set $\{(x, y; t) \in \mathbf{R}^2 \times \mathbf{R} \mid t \in I \wedge \varphi(x, y, t)\}$ is a *scene of* $\mathcal{M}$ if I is a maximal open interval in $\{t \mid a \leq t \leq b\}$ such that $\mathcal{M}$ is continuous in each $t_0 \in I$.
- A point t_0 in which $\mathcal{M}$ is not continuous is called a *cut in* $\mathcal{M}$. □

The movie of Figure 2 is continuous in every t in the open intervals $(-1, 1)$, $(1, 4)$ and $(4, 5)$. These intervals therefore determine the three scenes of the movie. There are four cuts in the movie of Figure 2: in $t = -1$, 1, 4 and 5. Remark that the beginning and the end of a movie are always cuts.

3. Computability Results

In this section, we present two computability results concerning movies. First, we show that it is decidable whether two (parameterized) frames are homeomorphic. Next, we will show that a movie has a finite number of scenes and cuts and that formulas describing them can be computed from the formula that defines the movie.

A key lemma in this context is the following.

Lemma 3.1 *It is decidable whether two movie frames are isotopic*[1]*, and also whether they are homeomorphic.*

Proof (sketch). Let A and B be two movie frames. A and B are homeomorphic if and only if A is isotopic to B or to a reflection of B (see, e.g., [7, 14, 19]). It therefore suffices to prove that it is decidable whether A is isotopic to B.

The algorithm to decide isotopy first computes for $X = A$, respectively B the labeled planar graph embedding G_X as follows. The nodes of G_X are the "singular points" of X, i.e., the points that do not belong to the topological interior of X or the complement of X, nor to a topologically smooth border of X (see [12] for a formal definition) together with the lowest left most points on each closed curve of the topological border of X on which there is no singular point. As an illustration, we take the frame X shown in (a) of Figure 3. The singular points of X are p_1, p_2 and p_3. The closed polygon in the right upper corner of X does not contain a singular point and has more than one most left point. Of these the lowest is picked: p_4. We remark that the singular points can be computed by means of a first-order formula in the theory of the real numbers. It was shown by Tarski that this theory is effective [20], and symbolic algorithms for the first-order theory of the reals [1, 5, 17] can effectively compute the nodes. The computation of the other nodes can be performed via a Cylindrical Algebraic Decomposition (CAD) [5] (see also [1]).

[1]Isotopic means homeomorphic by an orientation preserving homeomorphism.

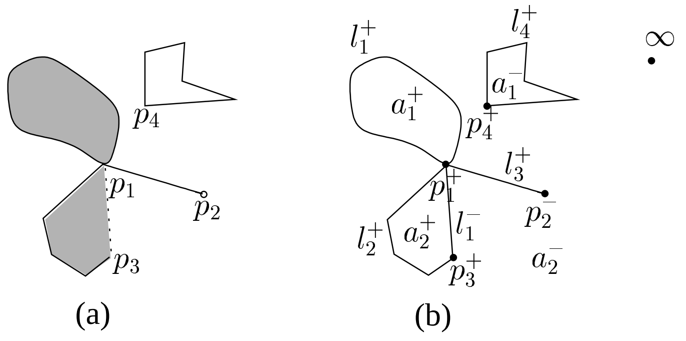

Figure 3. A frame of a movie X (a) and its graph G_X (b).

In the graph G_X (see (b) in Figure 3) these nodes are labeled with typed labels: p_i^+ for nodes that belong to X and p_i^- labels for nodes that do not belong to X.

Next, the connected components of the intersection of X with the topological border of X minus the labeled points are computed. These form edges of G_X and are labeled with labels of type l_i^+. Similarly, l_i^- labels are given to the connected components of the border that does not belong to X. The topological border of a frame can be computed in the first-order theory of the reals. The computation of connected components of a frame is described in [4, 8, 9]. Finally, the areas formed by the graph embedding are computed and labeled a_i^+, respectively a_i^- depending on their containment in X. Let the sets of labeled nodes, edges and areas be called $\mathcal{P}_X$, $\mathcal{L}_X$ and $\mathcal{A}_X$, respectively.

From results in [11] it follows that A and B are isotopic if and only if there exist bijections $b_P : \mathcal{P}_A \to \mathcal{P}_B$, $b_L : \mathcal{L}_A \to \mathcal{L}_B$, and $b_A : \mathcal{A}_A \to \mathcal{A}_B$ that map $+$-labels to $+$-labels and $-$-labels to $-$-labels and that preserve the clockwise occurrence of edges and areas around each of the labeled nodes. These conditions can be verified. It is therefore decidable whether two movie frames are isotopic. $\square$

It should be remarked (details omitted) that it can be decided whether two frames A and B are isotopic in polynomial time (in the size of the polynomial constraint formulas that describe A and B).

Theorem 3.1 *Let $\mathcal{M}_1$ and $\mathcal{M}_2$ be two movies. There is an algorithm that on input these two movies produces a formula $\psi_{\mathcal{M}_1\mathcal{M}_2}(t_1, t_2)$ built with conjunction and disjunction from formulas of the form $a < t_i < b$ and $t_i = c$ (a, b, c constants and $i = 1, 2$) that expresses, in function of t_1 and t_2, whether the two frames $\mathcal{M}_1^{t_1}$ and $\mathcal{M}_2^{t_2}$ are isotopic (the same is true for homeomorphic).*

Proof (sketch). Let $\mathcal{M}$ be the movie $\{(x, y; t) \in \mathbf{R}^2 \times \mathbf{R} \mid a \le t \le b \wedge \varphi(x, y, t)\}$. Collins proves that from the formula $a \le t \le b \wedge \varphi(x, y, t)$ a Cylindrical Algebraic Decomposition (CAD) $\mathcal{C}$ of $\mathbf{R}^2 \times \mathbf{R}$ can be computed such that each cell in $\mathcal{C}$ entirely belongs to $\mathcal{M}$ or to the complement of $\mathcal{M}$ [5] (see also [1]). $\mathcal{C}$ induces a CAD $\mathcal{C}_{x,t}$ of the (x, t)-plane which in its turn induces a CAD $\mathcal{C}_t$ of the t-axis. For the movie of Figure 2 this is illustrated in Figure 4. The cells of $\mathcal{C}$ are points, lines, curves, 2-dimensional surfaces and 3-dimensional areas that are built as stacks on the different cells of $\mathcal{C}_{x,t}$. The cells of $\mathcal{C}_{x,t}$ are the (black and grey) dots, arcs and patches of white space in Figure 4 (compare with Figure 2). $\mathcal{C}_t$ consists of the grey dots on the t-axis and the open intervals determined by them as shown in Figure 4. The cells of $\mathcal{C}_{x,t}$ are stacks built on these points and intervals. It can be easily shown that the movie $\mathcal{M}$ is continuous in each t that belongs to one of the open intervals of $\mathcal{C}_t$. So, $\mathcal{M}$ remains isotopic in these intervals. The only possible cuts of the movie are therefore the points of $\mathcal{C}_t$ (grey dots).

The algorithm we seek could therefore work as follows. Compute, using Collins's CAD algorithm, both for $\mathcal{M}_1$ and $\mathcal{M}_2$ the representatives of all the cells in the induced CAD $\mathcal{C}_t$. Let these be $r_1 < r_2 < \cdots < r_n$ and $s_1 < s_2 < \cdots < s_m$ respectively. Decide, using Lemma 3.1, for each pair (r, s) in the set $\{r_1, \ldots, r_n\} \times \{s_1, \ldots, s_m\}$, whether $\mathcal{M}_1^r$ is isotopic to $\mathcal{M}_2^s$. If they are, and if r represents, let's say an interval $a < t_1 < b$ of the induced CAD of the t-axis in $\mathcal{M}_1$ and if s represents, let's say a point c in the induced CAD of the t-axis in $\mathcal{M}_2$, then we add the conjunction $a < t_1 < b \wedge t_2 = c$ as a disjunct to the formula $\psi_{\mathcal{M}_1\mathcal{M}_2}(t_1, t_2)$. $\square$

To illustrate the previous theorem, we give $\psi_{\mathcal{M}\mathcal{M}}(t_1, t_2)$ for the movie $\mathcal{M}$ of Figure 2:

$$\big((t_1 = -1 \vee t_1 = 1 \vee t_1 = 4) \wedge (t_2 = -1 \vee t_2 = 1 \vee t_2 = 4)\big)$$
$$\vee\ (-1 < t_1 < 1 \wedge -1 < t_2 < 1) \vee (1 < t_1 < 4 \wedge 1 < t_2 < 4)$$
$$\vee\ \big((t_1 > 4 \vee t_1 < -1) \wedge (t_2 > 4 \vee t_2 < -1)\big).$$

Theorem 3.2 *A movie has a finite number of scenes and cuts. Polynomial constraint formulas describing them can be effectively computed.*

Proof (sketch). Let $\mathcal{M}$ be the movie $\{(x, y; t) \in \mathbf{R}^2 \times \mathbf{R} \mid a \le t \le b \wedge \varphi(x, y, t)\}$. Consider again a CAD of $\mathcal{M}$,

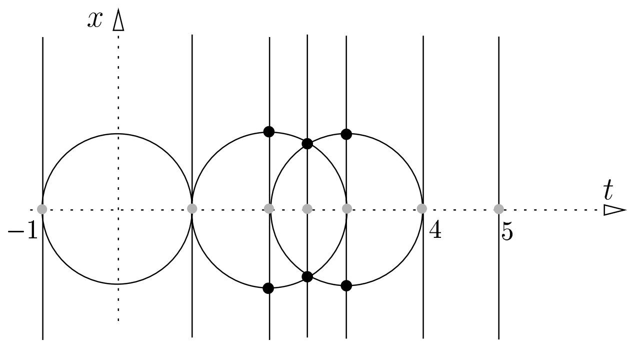

Figure 4. The induced CADs $\mathcal{C}_{x,t}$ (black and grey) and $\mathcal{C}_t$ (grey only) for the movie of Figure 2.

as in the proof of the previous theorem. From the proof of the previous theorem it is clear that there are only a finite number of cuts and scenes. The cuts are among the points of the induced CAD $\mathcal{C}_t$. Not all these points must be cuts, however. For the example of Figures 2 and 4, the grey dots at $t = -1, 1, 4$ and 5 are cuts. The one at $t = 5/2$, however, is not.

For what concerns the computation of the scenes and cuts, we first compute the candidate points for cuts (namely, the points in $\mathcal{C}_t$). It remains to be tested whether $\mathcal{M}$ is continuous in these points. Let t_0 be a point in $\mathcal{C}_t$ different from the beginning or end of $\mathcal{M}$. The test for continuity of $\mathcal{M}$ in t_0 is two-fold:

1. test whether the frame $\mathcal{M}^{t_0}$ is isotopic to two frames $\mathcal{M}^{t_0-\tau}$ and $\mathcal{M}^{t_0+\tau}$ in the neighboring intervals;

2. test for each $(x, y, t_0) \in \mathcal{M}$ (resp. $\notin \mathcal{M}$) whether for each small enough $\tau > 0$ and each $\delta > 0$ there exist a points (x', y') and (x'', y'') at distance at most δ from (x, y) such that $(x', y', t_0 + \tau) \in \mathcal{M}$ (resp. $\notin \mathcal{M}$) and $(x'', y'', t_0 - \tau) \in \mathcal{M}$ (resp. $\notin \mathcal{M}$).

The first test can be performed using the techniques outlined in Lemma 3.1. The second test can be expressed as a sentence in the first-order theory of the reals and is therefore effective [20]. Condition 1 is clearly necessary for continuity. It is, however, not sufficient. At a cut, a frame can, for instance, just jump to a different location. This would not violate Condition 1. Condition 2 guarantees that there is not a jump. When it is decided which of the points of $\mathcal{C}_t$ are cuts, the computation of the scenes is straightforward. Collins's algorithm produces polynomial constraint formulas for all the cells in a CAD. The scenes and cuts can therefore also be given by means of their defining polynomial constraint formula. $\square$

4. Querying Movie Databases

In this section, we present an effective SQL-like language to query movie databases: $\mathcal{M}$SQL. This language is based on the computability results of the previous section and on a well-known query language for databases in the constraint databases model, namely the relational calculus augmented with polynomial constraints.

First, we define

Definition 4.1 A *movie database query* is a mapping that maps every n-tuple of movies to a movie. $\square$

In the following, we will consider queries that have parameters $\mathcal{M}_1, \ldots, \mathcal{M}_n$.

The calculus: We will use the relational calculus augmented with polynomial constraints, the *calculus* for short, as an essential part of $\mathcal{M}$SQL. The calculus was introduced and studied in, e.g., [10, 16] (see also [13]).

A calculus formula

$$\varphi(x_1, \ldots, x_m, \mathcal{M}_1, \ldots, \mathcal{M}_n)$$

is built from the atomic formulas $\mathcal{M}_i(x, y, t)$ $(i = 1, \ldots, n)$ and $p(z_1, \ldots z_k) > 0$ (p a polynomial), the logical connectives $\neg, \wedge, \vee$ and the quantifiers $\exists, \forall$.

The calculus formula $(\exists x_0)(\exists y_0)(\exists r > 0)(\forall x)(\forall y)$ $\big((x - x_0)^2 + (y - y_0)^2 = r^2 \rightarrow \mathcal{M}_1(x, y, t)\big)$, for instance, defines the moments when there appears a circle (as a subset) in movie $\mathcal{M}_1$.

The language $\mathcal{M}$SQL: An *elementary* query in $\mathcal{M}$SQL is of the form

```
Select   $(x, y; t)$
From     $\mathcal{M}_i, \mathcal{M}_j, \ldots$
Where    $a \leq t \leq b \wedge C,$
```

where a and b are real algebraic numbers and C is a condition that can be expressed by means of the usual logical connectives and quantifiers, calculus expressions, other elementary $\mathcal{M}$SQL queries and the following primitives:

- $\mathbf{Scene}(\{(x,y;t) \mid a \leq t \leq b \wedge \varphi(x,y,t,\mathcal{M}_1,\ldots,\mathcal{M}_n)\},(f,g))$, with φ a calculus formula and f and g computable[2] functions that work on inputs $(\mathcal{M}_1,\ldots,\mathcal{M}_n)$ and $(i,\mathcal{M}_1,\ldots,\mathcal{M}_n)$ (i a natural number), respectively, and return natural numbers. This primitive returns the movie consisting of the scenes and cuts in the movie defined by $a \leq t \leq b \wedge \varphi(x,y,t,\mathcal{M}_1,\ldots,\mathcal{M}_n)$ whose sequence numbers are $g(1,\mathcal{M}_1,\ldots,\mathcal{M}_n), g(2,\mathcal{M}_1,\ldots,\mathcal{M}_n),\ldots, g(f(\mathcal{M}_1,\ldots,\mathcal{M}_n),\mathcal{M}_1,\ldots,\mathcal{M}_n)$ (in that order and consecutive);

- $\mathbf{Cut}(\{(x,y;t) \mid a \leq t \leq b \wedge \varphi(x,y,t,\mathcal{M}_1,\ldots,\mathcal{M}_n)\},i)$, with φ a calculus formula and i a natural number. It returns the i-th cut in the movie defined by $a \leq t \leq b \wedge \varphi(x,y,t,\mathcal{M}_1,\ldots,\mathcal{M}_n)$;

- $\mathbf{Isotopic}(\{(x,y;t) \mid a \leq t \leq b \wedge \varphi_1(x,y,t,\mathcal{M}_1,\ldots,\mathcal{M}_n)\}^{t_1}, \{(x,y;t) \mid a \leq t \leq b \wedge \varphi_2(x,y,t,\mathcal{M}_1,\ldots,\mathcal{M}_n)\}^{t_2})$, with φ_1 and φ_2 calculus formulas. It expresses the condition on t_1 and t_2 that tells us when the two given frames are isotopic (as discussed in Theorem 3.1);

- $\mathbf{Begin}(\{(x,y;t) \mid a \leq t \leq b \wedge \varphi(x,y,t,\mathcal{M}_1,\ldots,\mathcal{M}_n)\})$ with φ a calculus formula. Also $\mathbf{End}(\{(x,y;t) \mid a \leq t \leq b \wedge \varphi(x,y,t,\mathcal{M}_1,\ldots,\mathcal{M}_n)\})$, with φ a calculus formula. They express the beginning and end of the movie defined by $a \leq t \leq b \wedge \varphi(x,y,t,\mathcal{M}_1,\ldots,\mathcal{M}_n)$.

Furthermore, from elementary queries more complicated queries can be constructed by composition, which we denote by $\circ$. For two movies $\mathcal{M}_1$ and $\mathcal{M}_2$, $\mathcal{M}_1 \circ \mathcal{M}_2$ is defined to be the movie consisting of $\mathcal{M}_1$ without its last frame, immediately followed by $\mathcal{M}_2$.

The result of an $\mathcal{M}$SQL query is a movie. The meaning of these queries is the obvious one and will be illustrated by the consequent examples. The function of the From-part of an elementary query is to indicate which of the input movies are under consideration. It should be noted that some of these primitives are redundant and are only added for ease of use. From the proof of Theorem 3.2 it follows that $\mathbf{Cut}()$ can be expressed in terms of $\mathbf{Isotopic}()$.

Finally, we remark that

Theorem 4.1 *$\mathcal{M}$SQL queries are computable.*

[2]This means computable on the polynomial constraint formulas that define the movies.

Proof (sketch). Let Q be a $\mathcal{M}$SQL query. Given the polynomial constraint formulas of the input movies $\mathcal{M}_1,\ldots,\mathcal{M}_n$, we can, by Theorems 3.1 and 3.2 replace all the occurrences of $\mathbf{Scene}()$, $\mathbf{Cut}()$ and $\mathbf{Isotopic}()$ in Q by concrete constraint formulas. Also the beginning and end of movies can be computed (from a CAD, for instance). Therefore all occurrences of $\mathbf{Begin}()$ and $\mathbf{End}()$ can be replaced by concrete constraint formulas. This way we obtain a first-order formula over the reals, that can possibly contain quantifiers. From Tarski's quantifier elimination theorem it follows that these can be eliminated [20]. This yields a polynomial constraint formula for the output of the query Q. The output is guaranteed to be a movie by the syntactic condition $a \leq t \leq b$ that appears in the Where part of elementary $\mathcal{M}$SQL queries. □

We give some examples of $\mathcal{M}$SQL queries, that illustrate that all the basic movie editing operations can be performed in $\mathcal{M}$SQL:

Example 1: "Give all the frames of the movie $\mathcal{M}_1$ that are homeomorphic to a circle" is expressible by the elementary query

```
Select   $(x,y;t)$
From     $\mathcal{M}_1$
Where    $\mathbf{Begin}(\mathcal{M}_1) \leq t \wedge$
         $t \leq \mathbf{End}(\mathcal{M}_1) \wedge \mathcal{M}_1(x,y,t) \wedge$
         $\mathbf{Isotopic}(\{(u,v) \mid u^2+v^2=1\}, \mathcal{M}_1^t)$.
```

Example 2: "Give all the scenes of the movie $\mathcal{M}_1$ of which all frames are homeomorphic to a circle" is a variation on the query of Example 1 and it is expressible by the elementary query

```
Select   $(x,y;t)$
From     $\mathcal{M}_1$
Where    $\mathbf{Begin}(\mathcal{M}_1) \leq t \wedge t \leq \mathbf{End}(\mathcal{M}_1) \wedge$
         $\mathbf{Scene}(\mathcal{M}_1, (\#\mathrm{scenes}, \mathrm{id}))(x,y,t) \wedge$
         $\mathbf{Isotopic}(\{(u,v) \mid u^2+v^2=1\},$
                  $\mathbf{Scene}(\mathcal{M}_1, (\#\mathrm{scenes}, \mathrm{id}))^t)$,
```

where $\#\mathrm{scenes}$ is the function that returns the number of scenes of the input movie and id is the identity function of the natural numbers.

Example 3: "Remove scene 2 from movie $\mathcal{M}_1$" is expressed by

```
Select   $(x,y;t)$
From     $\mathcal{M}_1$
Where    $\mathbf{Begin}(\mathcal{M}_1) \leq t \wedge t \leq \mathbf{End}(\mathcal{M}_1) \wedge$
         $\mathbf{Scene}(\mathcal{M}_1, (\#\mathrm{scenes}-1, g))(x,y,t)$,
```

where g is the function that maps 1 to itself and all $n > 1$ to $n+1$.

Example 4: "Give me the first scene of movie $\mathcal{M}_1$ followed by the second scene of movie $\mathcal{M}_2$ followed by movie $\mathcal{M}_3$" is a query that manipulates complete scenes. It can be expressed as the composition of the elementary queries given by the following three expressions:

$$\begin{array}{ll}\text{Select} & (x,y;t)\\ \text{From} & \mathcal{M}_1\\ \text{Where} & \mathbf{Begin}(\mathbf{Scene}(\mathcal{M}_1,(1,1))) \leq t \wedge\\ & t \leq \mathbf{End}(\mathbf{Scene}(\mathcal{M}_1,(1,1))) \wedge\\ & \mathbf{Scene}(\mathcal{M}_1,(1,1))(x,y,t)\end{array}$$

$$\begin{array}{ll}\text{Select} & (x,y;t)\\ \text{From} & \mathcal{M}_2\\ \text{Where} & \mathbf{Begin}(\mathbf{Scene}(\mathcal{M}_2,(1,2))) \leq t \wedge\\ & t \leq \mathbf{End}(\mathbf{Scene}(\mathcal{M}_2,(1,2))) \wedge\\ & \mathbf{Scene}(\mathcal{M}_2,(1,2))(x,y,t)\end{array}$$

$$\begin{array}{ll}\text{Select} & (x,y;t)\\ \text{From} & \mathcal{M}_3\\ \text{Where} & \mathbf{Begin}(\mathcal{M}_3) \leq t \wedge\\ & t \leq \mathbf{End}(\mathcal{M}_3) \wedge \mathcal{M}_3(x,y,t),\end{array}$$

where the numbers 1 and 2 stand for the constant functions to 1 and 2.

Example 5: A query of particular interest for the Star Trek movie of the Introduction: "Give all frames of $\mathcal{M}_1$ that contain a photon torpedo (i.e., an isolated point)" can be expressed as

$$\begin{array}{ll}\text{Select} & (x,y;t)\\ \text{From} & \mathcal{M}_1\\ \text{Where} & \mathbf{Begin}(\mathcal{M}_1) \leq t \wedge t \leq \mathbf{End}(\mathcal{M}_1) \wedge\\ & \mathcal{M}_1(x,y,t) \wedge (\exists x_0)(\exists y_0)\big(\mathcal{M}_1(x_0,y_0,t) \wedge\\ & (\exists \varepsilon > 0)(\forall u)(\forall v)(\mathcal{M}_1(u,v,t) \wedge\\ & \qquad (u-x_0)^2 + (v-y_0)^2 < \varepsilon\\ & \qquad\quad \rightarrow (u = x_0 \wedge v = y_0)\big).\end{array}$$

Example 6: The following query manipulates the frames of a scene. It also reverses a complete scene. "Play movie $\mathcal{M}_1$ at double speed followed by the reversed play of movie $\mathcal{M}_2$ turned upside down" is expressed as the composition of the movies expressed by the following elementary queries

$$\begin{array}{ll}\text{Select} & (x,y;t)\\ \text{From} & \mathcal{M}_1\\ \text{Where} & \mathbf{Begin}(\mathcal{M}_1) \leq 2t \wedge\\ & 2t \leq \mathbf{End}(\mathcal{M}_1)\\ & \wedge\ \mathcal{M}_1(x,y,2t)\end{array}$$

$$\begin{array}{ll}\text{Select} & (x,y;t)\\ \text{From} & \mathcal{M}_2\\ \text{Where} & -\mathbf{End}(\mathcal{M}_2) \leq t \wedge\\ & t \leq -\mathbf{Begin}(\mathcal{M}_2) \wedge \mathcal{M}_2(x,-y,-t).\end{array}$$

Example 7: The next example manipulates one scene. "Return the first half of the first scene of movie $\mathcal{M}_1$" is expressed by

$$\begin{array}{ll}\text{Select} & (x,y;t)\\ \text{From} & \mathcal{M}_1\\ \text{Where} & \mathbf{Scene}(\mathcal{M}_1,(1,1))(x,y,t) \wedge\\ & \mathbf{Begin}(\mathbf{Scene}(\mathcal{M}_1,(1,1))) < t \wedge\\ & t < (\mathbf{Begin}(\mathbf{Scene}(\mathcal{M}_1,(1,1)))+\\ & \qquad\qquad \mathbf{End}(\mathbf{Scene}(\mathcal{M}_1,(1,1))))/2.\end{array}$$

Example 8: The next example manipulates complete scenes without touching their individual frames. "Return the movie consisting of the scenes of movie $\mathcal{M}_1$ but played in reversed order" is expressed by

$$\begin{array}{ll}\text{Select} & (x,y;t)\\ \text{From} & \mathcal{M}_1\\ \text{Where} & \mathbf{Begin}(\mathcal{M}_1) \leq t \wedge\\ & t \leq \mathbf{End}(\mathcal{M}_1) \wedge\\ & \mathbf{Scene}(\mathcal{M}_1,(\#\text{scenes},g))(x,y,t),\end{array}$$

where g is the function that maps i to $\#\text{scenes} - i + 1$.

Example 9: The last example returns a certain computable scene. "Return the middle scene of movie $\mathcal{M}_1$" is expressed by

$$\begin{array}{ll}\text{Select} & (x,y;t)\\ \text{From} & \mathcal{M}_1\\ \text{Where} & \mathbf{Begin}(\mathcal{M}_1) \leq t \wedge\\ & t \leq \mathbf{End}(\mathcal{M}_1) \wedge\\ & \mathbf{Scene}(\mathcal{M}_1,(1,g))(x,y,t),\end{array}$$

where g is the function that maps every natural number to $(\#\text{scenes} + 1)/2$ if the number of scenes is odd and to $\#\text{scenes}/2$ if it is even.

5. Discussion and conclusion

We have presented a data model for movies and movie databases, in which a movie is considered to be a 2-dimensional semi-algebraic figure that can change in time. We have given a number of computability results concerning movies: homeomorphism and isotopy of movie frames are decidable; a movie has a finite number of scenes and cuts and these can be effectively computed.

Based on these computability results we have defined an SQL-like query language for movie databases. This query language supports most movie editing operations like cutting, pasting and selection of scenes.

We remark that the presented model is very elementary and that it can be extended and made more suitable for practical applications in many ways. For instance, the movies that we consider here have frames that are purely black and white (even without variations of grey and certainly without colors). Movies have one single "image", whereas many movies (e.g., cartoon movies) are multi-layered. The presented model can be extended to cope with this.

Finally, we remark that our model cannot straightforwardly be applied to existing movies. The problem of converting cinematographic movies, for instance, into the proposed model is beyond the scope of this paper. We remark however that a number of 3D animation tools and virtual reality environments work with data that can be readily converted into the constraint model. 3D Studio Max [21] and Virtual Reality Modeling Language (VRML) [2] are examples of such environments.

Acknowledgements. The authors are grateful to Sofie Haesevoets for helpful comments and suggestions to improve the paper.

References

[1] D.S. Arnon. Geometric reasoning with logic and algebra. *Artificial Intelligence*, 37, pages 37–60, 1988.

[2] G. Bell, A. Parisi, and M. Pesce. *The Virtual Reality Modeling Language*. `www.vrml.org/VRLM1.0/vrml10c.html`

[3] J. Bochnak, M. Coste, and M.-F. Roy. *Géométrie Algébrique Réelle*. Springer-Verlag, 1987.

[4] J. Canny, D. Grigor'ev, and N.N. Vorobjov jr. Finding Connected Components of a Semialgebraic Set in Subexponential Time. *Applicable Algebra in Engineering, Communication and Computing*, 2, pages 217–238, 1992.

[5] G.E. Collins. Quantifier elimination for real closed fields by cylindrical algebraic decomposition. In volume 33 of *Lecture Notes in Computer Science*, pages 134–183. Springer-Verlag, 1975.

[6] M. Coste. Ensembles semi-algébriques. In *Géometrie Algébrique Réelle et Formes Quadratiques*, volume 959 of *Lecture Notes in Mathematics*, pages 109–138. Springer-Verlag, 1982.

[7] R.H. Cromwell and R.H. Fox. *Introduction to Knot Theory*, volume 57 of *Graduate Texts in Mathematics*. Springer-Verlag, 1977.

[8] J. Heintz, T. Recio, and M.-F. Roy. Algorithms in Real Algebraic Geometry and Applications to Computational Geometry. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, Volume 6, pages 137–163, 1991.

[9] J. Heintz, M.-F. Roy, and P. Solernó. Description of the Connected Components of a Semialgebraic Set in Single Exponential Time. *Discrete and Computational Geometry*, 6, pages 1–20, 1993..

[10] P.C. Kanellakis, G.M. Kuper, and P.Z. Revesz. Constraint query languages. *Journal of Computer and System Sciences*, 51(1), pages 26–52, August 1995.

[11] B. Kuijpers, J. Paredaens, and J. Van den Bussche. Lossless Representation of Topological Spatial Data. In M. Egenhofer and J. Herring, editors, *Advances in Spatial Databases, 4th International Symposium, SSD'95*, volume 951 of *Lecture Notes in Computer Science*, pages 1–13, Springer-Verlag, 1995.

[12] B. Kuijpers, J. Paredaens, and J. Van den Bussche. On topological elementary equivalence of spatial databases. In F. Afrati and Ph. Kolaitis, editors, *6th International Conference on Database Theory (ICDT '97)*, volume 1186 of *Lecture Notes in Computer Science*, pages 432–446, Springer-Verlag, 1997.

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

[14] E.E. Moise. *Geometric Topology in Dimensions 2 and 3*, volume 47 of *Graduate Texts in Mathematics*. Springer-Verlag, 1977.

[15] A. Nabutovsky. Personal communication. June 1997.

[16] J. Paredaens, J. Van den Bussche, and D. Van Gucht. Towards a theory of spatial database queries. In *Proceedings 13th ACM Symposium on Principles of Database Systems*, pages 279–288. ACM Press, 1994.

[17] J. Renegar. On the computational complexity and geometry of the first-order theory of the reals. *Journal of Symbolic Computation*, 13, pages 255–352, 1989.

[18] M.-F. Roy. Personal communication. May 1997.

[19] J. Stillwell. *Classical Topology and Combinatorial Group Theory*, volume 72 of *Graduate Texts in Mathematics*. Springer-Verlag, 1980.

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

[21] 3D Studio MAX. `http://www.max3d.com/`.