Affine-Invariant Querying of Spatial Data using a Triangle-Based Logic

Sofie Haesevoets [a] Bart Kuijpers [b] Peter Z. Revesz [c]

[a] *Hexagon Geospatial, Arenberg Research-Park, Leuven, Belgium,*
sofie.haesevoets@hexagon.com

[b] *Hasselt University, Theoretical Computer Science, B-3590 Diepenbeek, Belgium,*
bart.kuijpers@uhasselt.be

[c] *University of Nebraska-Lincoln, Department of Computer Science and Engineering, Avery Hall 358, Lincoln, NE 68588-0115, USA,*
revesz@cse.unl.edu

Key words: Spatial queries, Affine invariant queries, Logic

1 Introduction

The need for *affine-invariance* arises naturally from many spatial applications where different images such as various aerial photographs of some land area have to be recognized to belong together. For example, after the recent floods in the U.S. Midwest an aerial survey had to be made to assess the damages. In bringing together the "before" and the "after" images, a computer program needs to solve two distinct problems. First, the change of the landscape due to the flood. Second, even where there is no flood, due to different viewing angles, the "before" and the "after" images are not perfect copies but only affine-invariant images of each other. Hence affine-invariant similarity measures are needed between pairs of images [14,15,18]. After matching the "before" and the "after" images, the finer details of the flood can be examined by a spatial database query. In this paper, we propose a practical approach to combine the two steps and use *affine-invariant spatial database queries*.

The main idea of our affine-invariant spatial database queries, which we call *affine triangle logic*, is to use sets of triangles as a basic data representation. This can be considered an abstract spatial data type [21]. Geographic information systems also rely on triangles as the fundamental basis of representation, particularly in *Triangulated Irregular Networks* [19,29]. In computer graphics and traffic network simulations data is approximated by triangular meshes (e.g., [4,6,7,31]), and in many spatial interpolation algorithms those triangular meshes serve as the basis for the interpolation [3,20]. If in all these areas the data is represented as a finite union of triangles, why should one reason about data as

a collection of points [12]? That is the main motivation of our paper for considering first-order languages in which variables are interpreted to range over triangles.

The rest of the paper is organized as follows. Section 2 reviews some prior work on affine-invariant similarity measures and affine-invariant queries. Section 3 introduces triangle databases and a simple first-order logic over the reals on triangle databases. Section 4 proposes a new, first-order query language that has triangles as basic elements. We show that this language has the same expressive power as the affine-invariant segment of the queries in Section 3. We give some examples illustrating the expressiveness of our language and address the notion of safety of triangle queries. We show that it is undecidable whether a specific triangle query returns a finite output on finite input. However, it is decidable whether the output of a query on a particular finite input database can be represented as a finite union of triangles. We also show that we can express this finite representation in our triangle language. Section 5 considers the problem of affine-invariant image recognition for patterned triangles, which occurs many applications, where we have to consider not only the shape but also the patterns and colors that may be present on the surface of the objects. The patterns and colors can add extra visual information about the nature of objects and help recognize them. Finally, Section 6 gives some conclusion and directions for future work.

2 Related Work and Basic Concepts

Affinities are one of the transformation groups proposed at the introduction of the concept of "genericity of query languages" applied to constraint databases [24]. In addition, various subgroups of the affinities [24] and supergroups of the affinities [17,23] have been studied. Affine-invariance of patterned triangles was studied by Revesz [28]. Affine-invariant norms and triangulations were studied and used in computer graphics [22], affine-invariant image retrieval was studied in [13,26,30], affine-invariant data transformations of spatial and spatio-temporal constraint databases in [8,25,27] and affine-invariant query languages in [11,12]. Affine-invariance is also useful in analyzing and generating mazes [9].

The idea that the result of a spatial database query should be invariant under some group of spatial transformations was introduced by Paredaens, Van den Bussche and Van Gucht [24]. In a follow-up article, Gyssens, Van den Bussche and Van Gucht [12] proposed several first-order query languages which are invariant under the group of the affinities of the ambient space or some subgroup thereof. In these languages, variables are assumed to range over points in some real space $\mathbb{R}^n$ ($\mathbb{R}$ denotes the set of real numbers), rather than over real numbers (that is, the coordinates of such points). For the transformation group consisting of the affinities, the point language with only one predicate that expresses *betweenness* of three points, is shown to have the same expressivity as the affine-invariant fragment of first-order logic over the reals, on point databases. We use this result to prove the expressiveness of the triangle-based logic which is introduced and discussed in this paper. Therefore, we recall some definitions from the article of Gyssens, Van den Bussche

and Van Gucht [12]. All definitions listed in this section can be found there.

We start with the well-known definition of a constraint database [16,25,27] (or semi-algebraic database), since this is the general setting which we work in.

Definition 2.1 A *semi-algebraic relation in* $\mathbb{R}^n$ is a subset of $\mathbb{R}^n$ that can be described as a Boolean combination of sets of the form

$$\{(x_1, x_2, \ldots, x_n) \in \mathbb{R}^n \mid p(x_1, x_2, \ldots, x_n) > 0\},$$

where p is a polynomial with integer coefficients in the real variables $x_1, x_2, \ldots, x_n$. □

In mathematical terms, semi-algebraic relations are known as *semi-algebraic sets* [5].

We also call a semi-algebraic relation in $\mathbb{R}^n$ *a semi-algebraic relation of arity* n. A semi-algebraic database is essentially a finite collection of semi-algebraic relations. We give the definition next.

Definition 2.2 A *(semi-algebraic) database schema* σ is a finite set of (semi-algebraic) relation names, where each relation name R has an arity associated to it, which is a natural number and which is denoted by $ar(R)$.

Let σ be a database schema. A *semi-algebraic database over* σ is a structure $\mathscr{D}$ over σ with domain $\mathbb{R}$ such that, for each relation name R of σ, the associated relation $R^{\mathscr{D}}$ in $\mathscr{D}$ is a semi-algebraic relation of arity $ar(R)$. □

Example 2.1 Let $\sigma = \{R, S\}$, with $ar(R) = 2$ and $ar(S) = 1$ be a semi-algebraic database schema. Then the structure $\mathscr{D}$ given by

$$(\mathbb{R}, R^{\mathscr{D}} = \{(x_1, x_2) \in \mathbb{R}^2 \mid x_1^2 + x_2^2 < 1\}, S^{\mathscr{D}} = \{x \in \mathbb{R} \mid 0 \leq x \leq 1\})$$

is an example of a semi-algebraic database over σ that contains the open unit disk and the closed unit interval. □

Definition 2.3 Let σ be a semi-algebraic database schema. The language FO(+, ×, <, 0, 1, σ) (or FO(+, ×, <, 0, 1), for short , if σ is clear from the context), which is first-order logic over the real numbers with polynomial constraints, is the first-order language where the variables are assumed to range over real numbers, where the atomic formulas are either of the form $p(x_1, x_2, \ldots, x_n) > 0$, with p a polynomial with integer coefficients in the real variables $x_1, x_2, \ldots, x_n$, or the relation names from σ applied to real terms. Atomic formulas are composed into formulas, using the operations $\wedge$, $\vee$ and $\neg$ and the quantifiers $\forall$ and $\exists$. □

Example 2.2 Consider the semi-algebraic database from Example 2.1. The expression

$$R(x, y) \wedge y > 0$$

is an FO(+, ×, <, 0, 1, $\{R, S\}$)-formula selecting the part of the open unit disk that lies strictly above the x-axis of the plane $\mathbb{R}^2$. □

We restrict all further definitions and results to the dimension $n = 2$, as this is the dimension we work in in the remainder of this paper.

Apart from semi-algebraic databases, we also consider geometric and triangle databases, in this paper. For clarity of notation, we use the notational convention that is summarized in Table 1, throughout the paper. This convention contains the notation for the three types of databases as well as their schemes and relations. In the context of semi-algebraic, geometric and triangle databases, the basic objects are real number, 2-dimensional points and triangles in the plane. Table 1 also gives the notation for variables and constants of these data types.

Table 1
Notational conventions

	variables	constants	schema	relation	database
semi-algebraic	x_i	x_i	σ	R	$\mathscr{D}$
geometric	p_i	p_i	$\dot{\sigma}$	$\dot{R}$	$\dot{\mathscr{D}}$
triangle	$\triangle_i$	T_i	$\hat{\sigma}$	$\hat{R}$	$\hat{\mathscr{D}}$

Now, we give the definition of a "geometric database", a special type of constraint database. In a geometric database, the relations contain (a possibly infinite number of) tuples of points in $\mathbb{R}^2$. To express the relationship between geometric and semi-algebraic databases, we use the *canonical bijection* $\mathsf{can}_p : (\mathbb{R}^2)^k \to \mathbb{R}^{2k}$, which associates with each k-tuple $(p_1, \ldots, p_k)$ of points in $\mathbb{R}^2$ the $2k$-tuple $\mathsf{can}_p((p_1, \ldots, p_k)) = (x_1^1, x_1^2, \ldots, x_k^1, x_k^2)$ of real numbers, where we have $p_i = (x_i^1, x_i^2)$, for $1 \leq i \leq k$.

Definition 2.4 A *(geometric) database schema* $\dot{\sigma}$ is a finite set of (geometric) relation names, where each relation name $\dot{R}$ has an arity associated to it, which is a natural number and which is denoted by $ar(\dot{R})$.

Let $\dot{\sigma}$ be a geometric database schema. A *geometric database* over $\dot{\sigma}$ in $\mathbb{R}^2$ is a structure $\dot{\mathscr{D}}$ over $\dot{\sigma}$ with domain $\mathbb{R}^2$ such that, for each relation name $\dot{R}$ of $\dot{\sigma}$ of arity $k = ar(\dot{R})$, the associated relation $\dot{R}^{\dot{\mathscr{D}}}$ in $\dot{\mathscr{D}}$ is mapped by the canonical bijection can_p to a semi-algebraic relation in $\mathbb{R}^{2k}$. □

A geometric database $\dot{\mathscr{D}}$ over $\dot{\sigma}$ in $\mathbb{R}^2$ can be viewed naturally as a semi-algebraic database $\mathscr{D} = \mathsf{can}_p(\dot{\mathscr{D}})$ over a schema σ, which has, for each relation name $\dot{R}$ of $\dot{\sigma}$, a relation name R with arity $2k$, where k is the arity of $\dot{R}$ in $\dot{\sigma}$. For each relation name $\dot{R}$, of arity k, $R^{\mathscr{D}}$ is obtained from $\dot{R}^{\dot{\mathscr{D}}}$ by applying the canonical bijection can_p between $(\mathbb{R}^2)^k$ and $\mathbb{R}^{2k}$.

Definition 2.5 Let $\dot{\sigma}$ be a geometric database schema. A *k-ary geometric query* $\dot{Q}$ over $\dot{\sigma}$ in $\mathbb{R}^2$ is a partial computable function on the set of geometric databases over $\dot{\sigma}$. Furthermore, for each geometric database $\dot{\mathscr{D}}$ over $\dot{\sigma}$ on which $\dot{Q}$ is defined, $\dot{Q}(\dot{\mathscr{D}})$ is a geometric

relation of arity k. □

Queries that are invariant under some transformation group G of $\mathbb{R}^2$, are also called *G-generic* [24]. We define this next.

Definition 2.6 Let $\dot{\sigma}$ be a geometric database schema and $\dot{Q}$ a geometric query over $\dot{\sigma}$ in $\mathbb{R}^2$. Let G be a group of transformations of $\mathbb{R}^2$. Then $\dot{Q}$ is called *G-generic* if, for any two geometric databases $\dot{\mathscr{D}}$ and $\dot{\mathscr{D}}'$ over $\dot{\sigma}$ in $\mathbb{R}^2$ for which $\dot{\mathscr{D}}' = g(\dot{\mathscr{D}})$, for some $g \in G$, we have that $\dot{Q}(\dot{\mathscr{D}}') = g(\dot{Q}(\dot{\mathscr{D}}))$. □

In the remainder of this text, we focus on the group G of *affinities*. The affinities of $\mathbb{R}^2$ form the group of linear transformations having a regular matrix, that is, their matrix has a determinant different from zero. In other words, affinities of the plane have the following form:

$$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} e \\ f \end{pmatrix},$$

where $ad - bc \neq 0$.

We now give the definition of the first-order point logic FO($\{$**Between**$\}$, $\dot{\sigma}$), a first-order language where the variables are not interpreted as real numbers, as in FO($+$, $\times$, $<$, 0, 1), but as 2-dimensional points.

We first introduce the point predicate **Between**.

Definition 2.7 Let $p = (p_x, p_y)$, $q = (q_x, q_y)$ and $r = (r_x, r_y)$ be points in the plane $\mathbb{R}^2$. The expression **Between**(p, q, r) is true if and only if either q lies on the closed line segment between p and r or p and/or q and/or r coincide. □

In the example of Figure 1, **Between**$(\mathsf{p}, \mathsf{t}, \mathsf{q})$, **Between**$(\mathsf{p}, \mathsf{p}, \mathsf{q})$ and **Between**$(\mathsf{t}, \mathsf{s}, \mathsf{r})$ are true. On the other hand, **Between**$(\mathsf{t}, \mathsf{q}, \mathsf{p})$ and **Between**$(\mathsf{p}, \mathsf{q}, \mathsf{r})$ are not true.

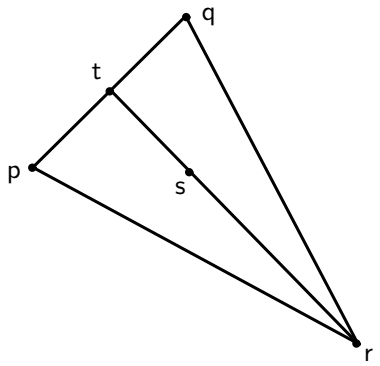

Fig. 1. The predicate **InTriangle** can be expressed using **Between**.

Definition 2.8 Let $\dot{\sigma}$ be a geometric database schema in $\mathbb{R}^2$. The first-order point language over $\dot{\sigma}$ and $\{$**Between**$\}$, denoted by FO($\{$**Between**$\}$, $\dot{\sigma}$) (or, if $\dot{\sigma}$ is clear from

the context, FO({**Between**})), is a first-order language with variables that range over points in $\mathbb{R}^2$, (denoted $p, q, \ldots$), where the atomic formulas are equality constraints on point variables, the predicate **Between** applied to point variables, and the relation names from $\dot{\sigma}$ applied to point variables. □

A FO({**Between**})-formula $\varphi(p_1, p_2, \ldots, p_l)$ over the relation names of $\dot{\sigma}$ and the predicate **Between** defines on each geometric database $\dot{\mathscr{D}}$ over $\dot{\sigma}$ a subset $\varphi(\dot{\mathscr{D}})$ of $(\mathbb{R}^2)^l$ in the standard manner.

Gyssens, Van den Bussche and Van Gucht have shown that the language FO({**Between**}) expresses exactly the affine-generic geometric queries expressible in FO(+, ×, <, 0, 1).

3 Triangle Databases and Triangle Database Queries

In this section, we introduce triangle databases and triangle database queries. Triangles in the plane $\mathbb{R}^2$ are the basic objects for this type of databases. The notational conventions concerning triangle variables, constants, relations and databases (and their schemes) are given in Table 1. We remark that triangles can be modelled as triples of points in $\mathbb{R}^2$. Therefore, in addition to these conventions, we remark that we also use the notation $\mathsf{T}_{\mathsf{pqr}}$ when we want to stress that a (constant) triangle has corner points p, q and r. Occasionally, we need to refer to the area of a triangle. The area of a triangle T will be denoted $\mathsf{area}(\mathsf{T})$.

We start with the definition of a *triangle database*, that is, a database that contains a (possibly infinite) collection of triangles. We model triangles by triples of points of $\mathbb{R}^2$, that is, by elements of $(\mathbb{R}^2)^3$. Triangles can degenerate, that is, corner points are allowed to coincide. For the remainder of this text, the term triangle refers to a triple of points. We refer to the set of points that is represented by a triangle as the *drawing* of that triangle.

Definition 3.1 (Drawing of a triangle) Let $\mathsf{T}_{\mathsf{pqr}} = (\mathsf{p}, \mathsf{q}, \mathsf{r}) \in (\mathbb{R}^2)^3$ be a spatial triangle. The *drawing* of $\mathsf{T}_{\mathsf{pqr}}$ is the subset of $\mathbb{R}^2$ that is the convex closure of the points p, q and r. □

We use the following generalization of the canonical bijection can_p and use the same notation for it: $\mathsf{can}_p : (\mathbb{R}^n)^k \to \mathbb{R}^{nk}$ maps tuples $(\mathsf{p}_1, \ldots, \mathsf{p}_k)$ to $(p_{1,1}, \ldots, p_{1,n}, \ldots, p_{k,1}, \ldots, p_{k,n})$, where, for $1 \leq i \leq k$ and $1 \leq j \leq n$, $p_{i,j}$ denotes the jth real coordinate of the point p_i of $\mathbb{R}^n$.

We also introduce the new bijection $\mathsf{can}_{tr} : \left((\mathbb{R}^2)^3\right)^k \to (\mathbb{R}^2)^{3k}$ that maps k-tuples of triangles in $\mathbb{R}^2$ to $3k$-tuples of points in $\mathbb{R}^2$.

We noe give the definition of triangle relations and databases (and their schemas).

Definition 3.2 (Triangle relations and databases) A *(triangle) database schema* $\hat{\sigma}$ is a finite set of relation names, where each relation name $\hat{R}$ has a natural number $ar(\hat{R})$, called its arity, associated to it.

A subset $\mathscr{T}$ of $\left(\left(\mathbb{R}^2\right)^3\right)^k$ is a *triangle relation of arity* k if

(i) its image under the canonical bijection $\mathsf{can}_p \circ \mathsf{can}_{tr} : \left(\left(\mathbb{R}^2\right)^3\right)^k \to \mathbb{R}^{6k}$ is a semi-algebraic relation of arity $6k$, and

(ii) for each element $t = ((\mathsf{p}_{1,1}, \mathsf{p}_{1,2}, \mathsf{p}_{1,3}), (\mathsf{p}_{2,1}, \mathsf{p}_{2,2}, \mathsf{p}_{2,3}), \ldots, (\mathsf{p}_{k,1}, \mathsf{p}_{k,2}, \mathsf{p}_{k,3})) \in \mathscr{T}$, also the elements $((\mathsf{p}_{1,j_{1,1}}, \mathsf{p}_{1,j_{1,2}}, \mathsf{p}_{1,j_{1,3}}), (\mathsf{p}_{2,j_{2,1}}, \mathsf{p}_{2,j_{2,2}}, \mathsf{p}_{2,j_{2,3}}), \ldots, (\mathsf{p}_{k,j_{k,1}}, \mathsf{p}_{k,j_{k,2}}, \mathsf{p}_{k,j_{k,3}}))$ are in $\mathscr{T}$, where $\sigma_i(1,2,3) = (j_{i,1}, j_{i,2}, j_{i,3})$, for $1 \leq i \leq k$ and $\sigma_i \in \mathscr{S}_3$, where $\mathscr{S}_3$ is the set of all permutations of $\{1,2,3\}$.

Let $\hat{\sigma}$ be a triangle database schema. A *triangle database* over $\hat{\sigma}$ in $\left(\mathbb{R}^2\right)^3$ is a structure $\hat{\mathscr{D}}$ over $\hat{\sigma}$ with domain $\left(\mathbb{R}^2\right)^3$ such that, for each relation name $\hat{R}$ of $\hat{\sigma}$, the associated triangle relation $\hat{R}^{\hat{\mathscr{D}}}$ in $\hat{\mathscr{D}}$ is a spatial triangle relation of arity $ar(\hat{R})$. □

We make the following remarks about items (i) and (ii) in Definition 3.2. They are discussed in Remark 3.1 below and Remark 3.4, which is postponed until after the definition of triangle database queries.

Remark 3.1 A triangle database $\hat{\mathscr{D}}$ over $\hat{\sigma}$ in $\left(\mathbb{R}^2\right)^3$ can be viewed naturally as a geometric database $\dot{\mathscr{D}}$ over the schema $\dot{\sigma}$, which has, for each relation name $\hat{R}$ of $\hat{\sigma}$, a relation name $\dot{R}$ with arity $3 \cdot ar(\hat{R})$. For each relation name $\hat{R}$, of arity k, $\dot{R}^{\dot{\mathscr{D}}}$ is obtained from $\hat{R}^{\hat{\mathscr{D}}}$ by applying the canonical bijection $\mathsf{can}_{tr} : \left(\left(\mathbb{R}^2\right)^3\right)^k \to \left(\mathbb{R}^2\right)^{3k}$. □

Example 3.1 It follows from the definition of triangle relations that they can be finitely represented by polynomial constraints on the coordinates of the corner points of the triangles they contain. For example, the unary triangle relation containing all triangles with one corner point on the x-axis, one on the y-axis and a third corner point on the diagonal $y = x$, can be finitely represented as follows:

$$\begin{aligned}\{(p_1,p_2,p_3) = ((x_1,y_1),(x_2,y_2),(x_3,y_3)) \in \left(\mathbb{R}^2\right)^3 \mid \\ (x_1 = 0 \wedge y_2 = 0 \wedge x_3 = y_3) \vee (x_1 = 0 \wedge y_3 = 0 \wedge x_2 = y_2) \\ \vee (x_2 = 0 \wedge y_1 = 0 \wedge x_3 = y_3) \vee (x_2 = 0 \wedge y_3 = 0 \wedge x_1 = y_1) \\ \vee (x_3 = 0 \wedge y_2 = 0 \wedge x_1 = y_1) \vee (x_3 = 0 \wedge y_1 = 0 \wedge x_2 = y_2)\}.\end{aligned}$$

Figure 2 gives some elements of this relation. Each triangle that is drawn is stored three times in the relation. □

Remark 3.2 For the remainder of this paper, we assume that databases are finitely encoded by systems of polynomial equations and that a specific data structure is fixed

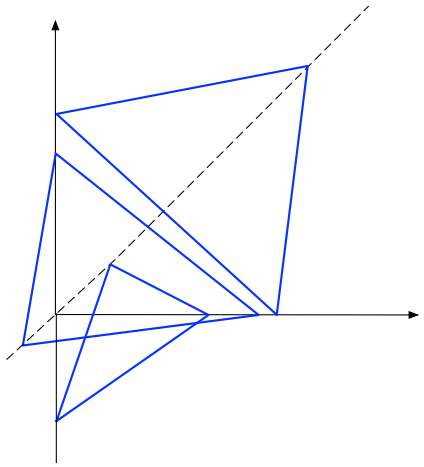

Fig. 2. Some elements of the relation represented in Example 3.1.

(possible data structures are dense or sparse representations of polynomials). The specific choice of data structure is not relevant to the topic of this paper, but we assume that one is fixed. When we talk about computable queries later on, we mean Turing computable with respect to the chosen encoding and data structures. □

We also remark the following.

Remark 3.3 The data model and the query languages presented in this paper can be extended straightforwardly to the situation where spatial relations are accompanied by classical thematic information. However, because the problem that is discussed here is captured by this simplified model, we stick to it for reasons of simplicity of exposition. □

We now define triangle database queries.

Definition 3.3 (Triangle database queries) Let $\hat{\sigma}$ be a triangle database schema and let us consider input triangle databases over $\hat{\sigma}$. A *k-ary triangle database query* $\hat{Q}$ over $\hat{\sigma}$ is a computable partial mapping (in the sense of Remark 3.2) from the set of triangle databases over $\hat{\sigma}$ to the set of k-ary triangle relations. □

Remark 3.4 In item (ii) of Definition 3.2, we require that, if a triangle T is involved in a relation, that also all other triangles with the same drawing are stored in that relation. The reason for this is that we do not want the triangle queries to be dependent of the actual order and orientation used when enumerating the corner points of a triangle. When emphasizing property (ii) of a triangle relation, we will call it *consistency* and talk about *consistent triangle relations*. Also, a database is said to be consistent, if all its relations are consistent. □

We illustrate the consistency property with some examples:

Example 3.2 Let $\hat{\sigma} = \{\hat{R}\}$ be a triangle database schema. First, we list some queries over $\hat{\sigma}$ that are not consistent:

• $\hat{Q}_1$: *Give all triangles in $\hat{R}$ for which their first and second corner points coincide.*

• $\hat{Q}_2$: *Give all triangles for which the segment defined by their first and second corner point is a boundary segment of one of the triangles in $\hat{R}$.*

Now some consistent queries follow:

• $\hat{Q}_3$: *Give all triangles in $\hat{R}$ that are degenerated into a line segment.*

• $\hat{Q}_4$: *Give all triangles that share a boundary segment with some triangle in $\hat{R}$.*

It is clear that the inconsistent queries are rather artificial. When a user specifies the triangles that should be in the result of a query, she intuitively thinks of the drawings of those triangles. The order of the corner points used in the construction of those triangles should not be important. □

Definition 3.4 (Equivalence of point queries and triangle queries) Let $\hat{\sigma}$ be a triangle database schema and let us consider input triangle databases over $\hat{\sigma}$. Let $\dot{\sigma}$ be the corresponding point database schema (see Remark 3.1). Let $\hat{Q}$ be a k-ary triangle database query over $\hat{\sigma}$ and let $\dot{Q}$ be a $3k$-ary point database query over $\dot{\sigma}$. We say that $\hat{Q}$ *and* $\dot{Q}$ *are equivalent*, denoted $\hat{Q} \equiv_{\triangle} \dot{Q}$, if for every database $\hat{\mathscr{D}}$ over $\hat{\sigma}$ we have

$$\mathsf{can}_{tr}(\hat{Q}(\hat{\mathscr{D}})) = \dot{Q}(\mathsf{can}_{tr}(\hat{\mathscr{D}})).$$

□

With this definition of equivalence between triangle database queries and point database queries we can now discuss how the point language FO({**Between**}) can be used to query triangle databases. We have to keep in mind that only point databases can be considered that are the image under the bijections can_{tr} of spatial triangle databases.

Definition 3.5 (FO({Between}) as a triangle query language) Let $\hat{\sigma} = \{\hat{R}_1, \hat{R}_2, \ldots, \hat{R}_m\}$ be a triangle database schema. Let $\dot{R}_i$ be the corresponding point relation names of arity $3{\cdot}ar(\hat{R}_i)$, for $i = 1 \ldots m$, and let $\dot{\sigma}$ be the point database schema $\{\dot{R}_1, \dot{R}_2, \ldots, \dot{R}_m\}$.

Let $\varphi(p_{1,1}, p_{1,2}, p_{1,3}, p_{2,1}, p_{2,2}, p_{2,3}, \ldots, p_{k,1}, p_{k,2}, p_{k,3})$ be a FO({**Between**})-formula expressing a $(3k)$-ary point query $\dot{Q}$ which is equivalent to a k-ary triangle query $\hat{Q}$. For each input triangle database $\hat{\mathscr{D}}$ over $\hat{\sigma}$, $\hat{Q}(\hat{\mathscr{D}})$ is defined as the set of points $(\mathsf{p}_{1,1}, \mathsf{p}_{1,2}, \mathsf{p}_{1,3}, \mathsf{p}_{2,1}, \mathsf{p}_{2,2}, \mathsf{p}_{2,3}, \ldots, \mathsf{p}_{k,1}, \mathsf{p}_{k,2}, \mathsf{p}_{k,3})$ in $(\mathbb{R}^6)^k$ such that

$$(\mathbb{R}^2, =, \mathbf{Between}, \dot{R}_1^{\dot{\mathscr{D}}}, \dot{R}_2^{\dot{\mathscr{D}}}, \ldots, \dot{R}_m^{\dot{\mathscr{D}}}) \models \varphi[\mathsf{p}_{1,1}, \mathsf{p}_{1,2}, \mathsf{p}_{1,3}, \mathsf{p}_{2,1}, \mathsf{p}_{2,2}, \mathsf{p}_{2,3}, \ldots, \mathsf{p}_{k,1}, \mathsf{p}_{k,2}, \mathsf{p}_{k,3}].$$

Here, $\dot{\mathscr{D}}$ is the image of $\hat{\mathscr{D}}$ under the canonical bijection can_{tr}. □

The language FO($\{$**Between**$\}$, $\dot{\sigma}$) is designed to formulate queries on point databases over some input schema $\dot{\sigma}$. Using this language to query triangle databases, involves expressing relations between the point sets that compose the triangles. This is a rather indirect way of expressing triangle relations. In the spirit of Geerts, Haesevoets and Kuijpers [11], we now construct affine-generic query languages based on triangle variables. As this language directly expresses relations between the triangles, this results in a more intuitive way of querying triangle databases. We define triangle-based logics next. Afterwards, we propose a specific triangle logic in Section 4.

Definition 3.6 (Triangle logics) Let $\hat{\sigma} = \{\hat{R}_1, \hat{R}_2, \ldots, \hat{R}_m\}$ be a triangle database schema and let $\mathbf{\Delta}$ be a set of (primitive) predicates of a certain arity over triangles in $\mathbb{R}^2$. The first-order logic over $\hat{\sigma}$ and $\mathbf{\Delta}$, denoted by FO($\mathbf{\Delta}$, $\hat{\sigma}$), can be used as a triangle query language when variables are interpreted to range over triangles in $\mathbb{R}^2$. The atomic formulas in FO($\mathbf{\Delta}$, $\hat{\sigma}$) are equality constraints on triangle variables, the predicates of $\mathbf{\Delta}$ applied to triangle variables and the relation names $\hat{R}_1, \hat{R}_2, \ldots, \hat{R}_m$ from $\hat{\sigma}$, applied to triangle variables.

A FO($\mathbf{\Delta}$, $\hat{\sigma}$)-formula $\varphi(\triangle_1, \triangle_2, \ldots, \triangle_k)$ defines for each spatial database $\hat{\mathscr{D}}$ over $\hat{\sigma}$ a subset $\varphi(\hat{\mathscr{D}})$ of $\left((\mathbb{R}^2)^3\right)^k$ defined as

$$\{(\mathsf{T}_1, \mathsf{T}_2, \ldots, \mathsf{T}_k) \in (\mathbb{R}^2)^{3k} \mid (\mathbb{R}^2, \mathbf{\Delta}^{\mathbb{R}^2}, \hat{R}_1^{\hat{\mathscr{D}}}, \hat{R}_2^{\hat{\mathscr{D}}}, \ldots, \hat{R}_m^{\hat{\mathscr{D}}}) \models \varphi[\mathsf{T}_1, \mathsf{T}_2, \ldots, \mathsf{T}_k] \}.$$

□

Remark 3.5 We use the symbol $=_\triangle$ to indicate equality of triangle variables, as opposed to equality of point variables. If it is clear from the context of a formula which type of variables is used, we will omit the index. □

In Section 4, we develop languages that have the same expressive power as FO($\{$**Between**$\}$) on triangle databases. We prove this by showing both soundness and completeness of this triangle language with respect to FO($\{$**Between**$\}$).

We introduce the concepts of soundness and completeness of logics at the level of coordinate-based languages and later lift them to geometric- and triangle-based langauges.

Definition 3.7 (Soundness and completeness) Let G be a group of transformations of $\mathbb{R}^2$ and let σ be a semi-algebraic database schema.

A query language $\mathscr{L}$ is said to be *sound* for the G-generic FO($+$, $\times$, $<$, 0, 1, σ)-queries on semi-algebraic databases, if formulas in $\mathscr{L}$ only express G-generic FO($+$, $\times$, $<$, 0, 1, σ)-queries on semi-algebraic databases.

A query language $\mathscr{L}$ is said to be *complete* for the G-generic FO($+$, $\times$, $<$, 0, 1, σ)-queries on semi-algebraic databases, if all G-generic FO($+$, $\times$, $<$, 0, 1, σ)-queries on semi-algebraic databases can be expressed in $\mathscr{L}$. □

4 Affine-Invariant Triangle Queries

In this section, we propose a triangle logic that captures exactly the class of first-order affine-generic queries on triangle databases. First, we remark the following.

Remark 4.1 We defined a triangle database as a special type of geometric database. Accordingly, we take the affine image of a triangle for affinities of $\mathbb{R}^2$, and not of $\mathbb{R}^6$. This corresponds to our intuition. One triangle is an affine image of another triangle, if the drawing of the first one is the affine image of the drawing of the second one. Hence, the affine image of a triangle with corner points p_1, p_2 and p_3 under some affinity α of the plane $\mathbb{R}^2$, is the triangle with corner points $\alpha(\mathsf{p}_1)$, $\alpha(\mathsf{p}_2)$ and $\alpha(\mathsf{p}_3)$. □

We introduce one binary triangle predicate, that is, **PartOf**. Intuitively, when applied to two triangles, this predicate expresses that the drawing of the first triangle is a subset ($\subseteq$) of the drawing of the second triangle. We consider $(\mathbb{R}^2)^3$ as the underlying domain. We show that the triangle predicate **PartOf** allows a natural extension to higher dimensions and other types of objects (not only triangles).

First, we define the predicate **PartOf** and equality on triangles more precisely.

Definition 4.1 (The triangle predicate PartOf) Let $\mathsf{T}_1 = (\mathsf{p}_{1,1}, \mathsf{p}_{1,2}, \mathsf{p}_{1,3})$ and $\mathsf{T}_2 = (\mathsf{p}_{2,1}, \mathsf{p}_{2,2}, \mathsf{p}_{2,3})$ be two triangles. The binary predicate **PartOf**, applied to T_1 and T_2, denoted **PartOf**$(\mathsf{T}_1, \mathsf{T}_2)$, expresses that the convex closure of the three points $\mathsf{p}_{1,1}$, $\mathsf{p}_{1,2}$ and $\mathsf{p}_{1,3}$ is a subset of the convex closure of the three points $\mathsf{p}_{2,1}$, $\mathsf{p}_{2,2}$ and $\mathsf{p}_{2,3}$. □

Figure 3 illustrates the predicate **PartOf**.

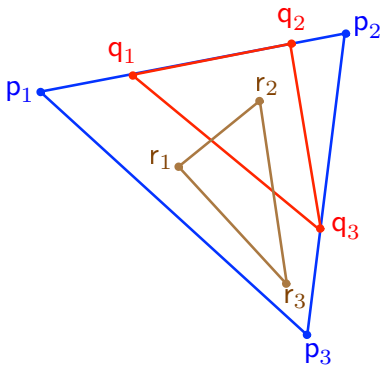

Fig. 3. An illustration of the **PartOf** predicate. Let $\mathsf{T}_1 = (\mathsf{p}_1, \mathsf{p}_2, \mathsf{p}_3)$, $\mathsf{T}_2 = (\mathsf{q}_1, \mathsf{q}_2, \mathsf{q}_3)$ and $\mathsf{T}_3 = (\mathsf{r}_1, \mathsf{r}_2, \mathsf{r}_3)$. Then the expressions **PartOf**$(\mathsf{T}_2, \mathsf{T}_1)$ and **PartOf**$(\mathsf{T}_3, \mathsf{T}_1)$ are true, but the expression **PartOf**$(\mathsf{T}_3, \mathsf{T}_2)$ is false.

We also define triangle-equality, which differs from the standard equality operation.

Definition 4.2 (Equality of triangles) Let T_1 and T_2 be two triangles. The expression $\mathsf{T}_1 =_\triangle \mathsf{T}_2$ is true if and only if both **PartOf**$(\mathsf{T}_1, \mathsf{T}_2)$ and **PartOf**$(\mathsf{T}_2, \mathsf{T}_1)$ are tue. □

Before analyzing the expressiveness of the language FO({**PartOf**}), we prove that the FO({**PartOf**})-queries are well-defined on consistent triangle databases. More concretely, given a triangle database schema $\hat{\sigma}$, we prove that the result of a k-ary FO({**PartOf**}, $\hat{\sigma}$)-query on a consistent input database over $\hat{\sigma}$ is a consistent triangle relation of arity k.

Lemma 4.1 (FO({PartOf}) is well-defined) Let $\hat{\sigma} = \{\hat{R}_1, \hat{R}_2, \ldots, \hat{R}_m\}$ be a spatial triangle database schema. Let $\hat{\mathscr{D}}$ be a consistent triangle database over $\hat{\sigma}$. For each FO({**PartOf**}, $\hat{\sigma}$)-query $\hat{Q}$, $\hat{Q}(\hat{\mathscr{D}})$ is a consistent triangle relation.

Proof. Let $\hat{\sigma} = \{\hat{R}_1, \hat{R}_2, \ldots, \hat{R}_m\}$ be a spatial triangle database schema. Let $\hat{\mathscr{D}}$ be a consistent spatial triangle database over $\hat{\sigma}$.

We prove this lemma by induction on the structure of FO({**PartOf**}, $\hat{\sigma}$)-queries. The atomic formulas of FO({**PartOf**}, $\hat{\sigma}$) are equality expressions on triangle variables, expressions of the form **PartOf**$(\triangle_1, \triangle_2)$, and expressions of the form $\hat{R}_i(\triangle_1, \triangle_2, \ldots, \triangle_{k_i})$, where $k_i = ar(\hat{R}_i)$. More complex formulas can be constructed using the Boolean operators $\wedge$, $\vee$ and $\neg$ and existential quantification.

For the atomic formulas, it is easy to see that, if two triangles T_1 and T_2 satisfy the conditions $\mathsf{T}_1 =_\triangle \mathsf{T}_2$ or **PartOf**$(\mathsf{T}_1, \mathsf{T}_2)$, that also $\mathsf{T}'_1 =_\triangle \mathsf{T}'_2$ respectively **PartOf**$(\mathsf{T}'_1, \mathsf{T}'_2)$ are true if and only if $\mathsf{T}_1 =_\triangle \mathsf{T}'_1$ and $\mathsf{T}_2 =_\triangle \mathsf{T}'_2$ are true. As we assume the input database $\hat{\mathscr{D}}$ to be consistent, the atomic formulas of the type $\hat{R}_i(\triangle_1, \triangle_2, \ldots, \triangle_{k_i})$, where $1 \leq i \leq m$, trivially return consistent triangle relations.

Next, we have to prove that the composed formulas always return consistent triangle relations. Let $\hat{\varphi}$ and $\hat{\psi}$ be two formulas in FO({**PartOf**}, $\hat{\sigma}$), of arity $k_{\hat{\varphi}}$ and $k_{\hat{\psi}}$ respectively, already defining consistent triangle relations. Then, the formula $(\hat{\varphi} \wedge \hat{\psi})$ (resp., $(\hat{\varphi} \vee \hat{\psi})$) also defines a triangle relation. This follows from the fact that the free variables of $(\hat{\varphi} \wedge \hat{\psi})$ (resp., $(\hat{\varphi} \vee \hat{\psi})$) are free variables in $\hat{\varphi}$ or $\hat{\psi}$. The universe of all triangles is trivially consistent. If a consistent subset is removed from this universe, the remaining part is still consistent. Therefore, $\neg\hat{\varphi}$ is well-defined. Finally, because consistency is defined argument-wise, the projection $\exists\mathsf{T}_1\, \hat{\varphi}(\mathsf{T}_1, \mathsf{T}_2, \ldots, \mathsf{T}_{k_\varphi})$ is consistent. □

After proving that the language FO({**PartOf**}) is well-defined, we can analyse the expressive power of the language FO({**PartOf**}). We prove that it is sound and complete for the affine-invariant fragment of first-order logic over the reals, on triangle databases. We prove this by comparing the languages FO({**PartOf**}) and FO({**Between**}).

As we have remarked at the end of Section 2, we already know that FO({**Between**}) is sound and complete for the affine-invariant fragment of first-order logic over the reals, on point databases [12].

The soundness and completeness of the query language FO({**PartOf**}) with respect to the language FO({**Between**}) is proved using two separate lemmas (Lemma 4.2 and Lemma 4.3). In both lemmas, formulas are translated from one language in the other,

by using induction on the structure of FO({**PartOf**})- and FO({**Between**})-formulas, respectively. This proof technique will be used several times in this text.

Therefore, we give the first such proofs in detail (in the Appendix). Later on, we will only develop the crucial points in similar proofs.

Lemma 4.2 (Soundness of FO({PartOf}) with respect to FO({**Between**})) Let $\hat{\sigma} = \{\hat{R}_1, \hat{R}_2, \ldots, \hat{R}_m\}$ be a triangle database schema. Let $\dot{R}_i$ be the corresponding point relation names of arity $3 \cdot ar(\hat{R}_i)$, for $(1 \leq i \leq m)$, and let $\dot{\sigma}$ be the point database schema $\{\dot{R}_1, \dot{R}_2, \ldots, \dot{R}_m\}$. Every FO({**PartOf**}, $\hat{\sigma}$)-expressible query can be expressed equivalently in FO({**Between**}, $\dot{\sigma}$).

To prove completeness, we translate FO({**Between**})-formulas into FO({**PartOf**})-formulas. Again, we prove this by induction on the structure of FO({**Between**})-formulas. However, this translation is not as straightforward as the translation in the other direction. We refer the interested reader to the Appendix for the rather technical proof.

Lemma 4.3 (Completeness of FO({PartOf})) Let $\hat{\sigma} = \{\hat{R}_1, \hat{R}_2, \ldots, \hat{R}_m\}$ be a triangle database schema and $\dot{\sigma}$ be the corresponding point database schema. Every FO({**Between**}, $\dot{\sigma}$)-expressible query can be expressed equivalently in FO({**PartOf**}, $\hat{\sigma}$).

Remark 4.2 So far, we showed that we can simulate any FO({**Between**}, $\dot{\sigma}$) formula $\dot{\varphi}(p_1, p_2, \ldots, p_k)$ by a formula $\varphi'(\triangle_1, \triangle_2, \ldots, \triangle_k)$, where $\mathbf{Point}(\triangle_i)$ is true for all $\triangle_i$ $(1 \leq i \leq k)$. However, when $\dot{\varphi}$ expresses a k-ary triangle database query Q (that is, $\dot{\varphi}$ has $3k$ free variables), we can do better.

Let $\dot{\varphi}$ be the FO({**Between**}, $\dot{\sigma}$)-formula expressing a k-ary triangle database query $\hat{Q}$. The free variables of $\dot{\varphi}$ are $p_{1,1}, p_{1,2}, p_{1,3}, p_{2,1}, p_{2,2}, p_{2,3}, \ldots, p_{k,1}, p_{k,2}, p_{k,3}$.

We now construct the FO({**PartOf**}, $\hat{\sigma}$)-formula $\hat{\varphi}$ expressing the query $\hat{Q}$ as follows:

$$\begin{aligned}
&\hat{\varphi}(\triangle_1, \triangle_2, \ldots, \triangle_k) \equiv \\
&\quad \exists\, \triangle_{1,1}\, \exists\, \triangle_{1,2}\, \exists\, \triangle_{1,3}\, \exists\, \triangle_{2,1}\, \exists\, \triangle_{2,2}\, \exists\, \triangle_{2,3} \ldots \exists\, \triangle_{k,1}\, \exists\, \triangle_{k,2}\, \exists\, \triangle_{k,3}\, (\\
&\qquad \bigwedge_{i=1}^{k} \mathbf{CornerP}(\triangle_{i,1}, \triangle_{i,2}, \triangle_{i,3}, \triangle_i) \wedge \\
&\qquad\quad \hat{\varphi}'(\triangle_{1,1}, \triangle_{1,2}, \triangle_{1,3}, \triangle_{2,1}, \triangle_{2,2}, \triangle_{2,3}, \ldots, \triangle_{k,1}, \triangle_{k,2}, \triangle_{k,3})).
\end{aligned}$$

For each triple of points, there are 6 different representations for the triangle having those points as its corner points. Therefore, for each tuple returned by $\hat{\varphi}'$, 6^k tuples will be returned by $\hat{\varphi}$. But we know that $\dot{\varphi}$ is a well-defined triangle query. This means that, for each $3k$-tuple of points $((\mathsf{p}_{1,1}, \mathsf{p}_{1,2}, \mathsf{p}_{1,3}), (\mathsf{p}_{2,1}, \mathsf{p}_{2,2}, \mathsf{p}_{2,3}), \ldots, (\mathsf{p}_{k,1}, \mathsf{p}_{k,2}, \mathsf{p}_{k,3}))$ satisfying $\dot{\varphi}$, also the tuples $((\mathsf{p}_{1,j_{1,1}}, \mathsf{p}_{1,j_{1,2}}, \mathsf{p}_{1,j_{1,3}}), (\mathsf{p}_{2,j_{2,1}}, \mathsf{p}_{2,j_{2,2}}, \mathsf{p}_{2,j_{2,3}}), \ldots, (\mathsf{p}_{k,j_{k,1}}, \mathsf{p}_{k,j_{k,2}}, \mathsf{p}_{k,j_{k,3}}))$, whith $\sigma_i(1,2,3) = (j_{i,1}, j_{i,2}, j_{i,3})$, with $1 \leq i \leq k$; $\sigma_i \in \mathscr{S}_3$, where $\mathscr{S}_3$ is the set of all permutations

of $\{1, 2, 3\}$, satisfy $\dot{\varphi}$. Therefore, $\hat{\varphi}$ and $\dot{\varphi}$ are equivalent according to Definition 3.4. □

We now combine the soundness and completeness lemmas, and use them to prove our main theorem for this section.

Theorem 4.1 (Expressiveness of FO({PartOf})) Let $\hat{\sigma} = \{\hat{R}_1, \hat{R}_2, \ldots, \hat{R}_m\}$ be a triangle database schema. Let R_i be the corresponding semi-algebraic relation names of arity $6 \cdot ar(\hat{R}_i)$, for $1 \leq i \leq m$, and let σ be the semi-algebraic database schema $\{R_1, R_2, \ldots, R_m\}$. The language FO($\{$**PartOf**$\}$, $\hat{\sigma}$) is sound and complete for the affine-generic FO(+, ×, <, 0, 1, σ)-queries on triangle databases.

Proof. Let $\hat{\sigma} = \{\hat{R}_1, \hat{R}_2, \ldots, \hat{R}_m\}$ be a triangle database schema. Let $\dot{R}_i$ be the corresponding point relation names of arity $3 \cdot ar(\hat{R}_i)$, for $1 \leq i \leq m$, and let $\dot{\sigma}$ be the point database schema $\{\dot{R}_1, \dot{R}_2, \ldots, \dot{R}_m\}$. Let $R_i (1 \leq i \leq m)$ be the corresponding semi-algebraic relation names of arity $6 \cdot ar(\hat{R}_i)$ and let σ be the semi-algebraic database schema $\{R_1, R_2, \ldots, R_m\}$.

From Lemma 4.2 and Lemma 4.3, we deduce that FO($\{$**PartOf**$\}$, $\hat{\sigma}$) is sound and complete for the FO($\{$**Between**$\}$, $\dot{\sigma}$)-queries on triangle databases.

Gyssens, Van den Bussche and Van Gucht showed that FO($\{$**Between**$\}$, $\dot{\sigma}$) is sound and complete for the affine-generic FO(+, ×, <, 0, 1, σ)-queries on point databases [12].

From the definition of triangle databases, we know that they are point databases. This concludes the proof. □

We remark that in the proofs of Lemma 4.2 and Lemma 4.3 we only use the fact that triangles are convex objects having three corner points. We use no other properties of triangles.

The following corollary follows from the known fact that FO($\{$**Between**$\}$, $\dot{\sigma}$) +**While** is sound and complete for the computable affine-generic queries on geometric databases [12]. The language FO($\{$**PartOf**$\}$, $\hat{\sigma}$) + **While** is a language in which FO($\{$**PartOf**$\}$, $\hat{\sigma}$)-definable relations can be created and which has a while-loop with FO($\{$**PartOf**$\}$, $\hat{\sigma}$)-definable stop conditions. For details on this language, we refer to [12].

Corollary 4.1 (Expressiveness of FO($\{$**PartOf**$\}$, $\hat{\sigma}$)+ **While)** Let $\hat{\sigma}$ be a spatial triangle database schema. The language FO($\{$**PartOf**$\}$, $\hat{\sigma}$)+ **While** is sound and complete for the computable affine-generic queries on triangle databases. □

We now give some examples of FO($\{$**PartOf**$\}$, $\hat{\sigma}$)-queries. We illustrate some geometrical constructions in Example 4.1. Afterwards, we formulate queries on an example triangle database in Example 4.2.

Example 4.1 We illustrate how to express that two triangles are "similar", that is, each side of the first triangle is parallel to a side of the second triangle. We denote the formula expressing this by **Sim**.

We use the predicates **ColSeg** and **ParSeg**, expressing that two line segments are collinear and parallel respectively, to simplify the expression for **Sim**. The predicate **ColSeg** is expressed as

$$\mathbf{ColSeg}(\triangle_1, \triangle_2) := \mathbf{Seg}(\triangle_1) \wedge \mathbf{Seg}(\triangle_2) \wedge \exists\, \triangle_3\ (\mathbf{Seg}(\triangle_3) \wedge \mathbf{PartOf}(\triangle_1, \triangle_3) \wedge \mathbf{PartOf}(\triangle_2, \triangle_3)).$$

Here, $\mathbf{Seg}(\triangle_1)$ is a shorthand for

$$\exists\, \triangle_4\ \exists\, \triangle_5\ (\mathbf{Point}(\triangle_4) \wedge \mathbf{Point}(\triangle_5) \wedge \forall\, \triangle_6\ ((\mathbf{Point}(\triangle_6) \wedge \mathbf{PartOf}(\triangle_6, \triangle_1)) \rightarrow (\mathbf{Between}_{\Delta}(\triangle_4, \triangle_6, \triangle_5)))),$$

and expresses that $\triangle_1$ is a triangle that is degenerated into a line segment. The fact that two line segments are parallel is now defined as follows:

$$\mathbf{ParSeg}(\triangle_1, \triangle_2) := \mathbf{Seg}(\triangle_1) \wedge \mathbf{Seg}(\triangle_2) \wedge \forall\, \triangle_3\ \forall\, \triangle_4\ ((\mathbf{ColSeg}(\triangle_1, \triangle_3) \wedge \mathbf{ColSeg}(\triangle_2, \triangle_4)) \rightarrow \neg\exists\, \triangle_5\ (\mathbf{PartOf}(\triangle_5, \triangle_3) \wedge \mathbf{PartOf}(\triangle_5, \triangle_4))).$$

Now we can write the expression for **Sim**:

$$\begin{aligned}
&\mathbf{Sim}(\triangle_1, \triangle_2) := \\
&\exists\, \triangle_{1,1}\ \exists\, \triangle_{1,2}\ \exists\, \triangle_{1,3}\ \exists\, \triangle_{1,4}\ \exists\, \triangle_{1,5}\ \exists\, \triangle_{1,6}\ \exists\, \triangle_{2,1}\ \exists\, \triangle_{2,2}\ \exists\, \triangle_{2,3}\ \exists\, \triangle_{2,4}\ \exists\, \triangle_{2,5}\ \exists\, \triangle_{2,6}\ (\\
&\quad \bigwedge_{i=1}^{2} (\mathbf{CornerP}(\triangle_{i,1}, \triangle_{i,2}, \triangle_{i,3}, \triangle_i) \wedge \mathbf{CornerP}(\triangle_{i,1}, \triangle_{i,1}, \triangle_{i,2}, \triangle_{i,4}) \wedge \\
&\quad \mathbf{CornerP}(\triangle_{i,2}, \triangle_{i,2}, \triangle_{i,3}, \triangle_{i,5}) \wedge \mathbf{CornerP}(\triangle_{i,3}, \triangle_{i,3}, \triangle_{i,1}, \triangle_{i,6})) \wedge \\
&\quad \bigvee_{\sigma(1,2,3)=(i_1,i_2,i_3), \sigma \in \mathscr{S}_3} (\mathbf{ParSeg}(\triangle_{1,4}, \triangle_{2,(3+i_1)}) \wedge \mathbf{ParSeg}(\triangle_{1,5}, \triangle_{2,(3+i_2)}) \\
&\qquad\qquad \wedge \mathbf{ParSeg}(\triangle_{1,6}, \triangle_{2,(3+i_3)}))),
\end{aligned}$$

where $\mathscr{S}_3$ is the set of all permutations of $\{1, 2, 3\}$. □

We proceed with an example of a spatial database containing information about butterflies, and give some examples of FO($\{$**PartOf**$\}$)-queries that can be asked to such a database.

Example 4.2 Consider a triangle database $\hat{\mathscr{D}}$ over the schema $\hat{\sigma} = \{ButterflyB, PlantP, Rural\}$ that contains information about butterflies and flowers. The unary triangle relation $ButterflyB$ contains all regions where some butterfly B is spotted. The unary triangle relation $PlantP$ contains all regions where some specific plant P grows. We also have a unary triangle relation $Rural$, containing rural regions. It is known in

biology that each butterfly appears close to some specific plant, as caterpillars only eat the leaves of their favorite plant. Suppose that it is also investigated that butterflies like to live in rural areas.

• $\hat{Q}_5$: *Are all butterflies B spotted in regions where the plant P grows?* This query can be used to see if it is possible that a butterfly was spotted in a certain region. The query $\hat{Q}_5(\triangle)$ can be expressed by the formula

$$\neg(\exists\ \triangle_1\ \exists\ \triangle_2\ (ButterflyB(\triangle_1) \wedge \mathbf{RealTriangle}(\triangle_2) \wedge \mathbf{PartOf}(\triangle_2, \triangle_1) \wedge \neg(\exists\ \triangle_3\ (PlantP(\triangle_3) \wedge \mathbf{PartOf}(\triangle_2, \triangle_3))))).$$

Here, $\mathbf{RealTriangle}(\triangle)$ is a shorthand for $\neg\mathbf{Point}(\triangle) \wedge \neg\mathbf{Seg}(\triangle)$.

• $\hat{Q}_6$: *Give the region(s) where we have to search if we want to see butterfly B.* The query $\hat{Q}_6(\triangle)$ can be expressed by the formula

$$\exists\ \triangle_2\ \exists\ \triangle_3\ (PlantP(\triangle_2) \wedge Rural(\triangle_3) \wedge \mathbf{PartOf}(\triangle, \triangle_2) \wedge \mathbf{PartOf}(\triangle, \triangle_3)).$$

• $\hat{Q}_7$: *Give the region inside the convex hull of the search region for butterfly B.* It is much more convenient to search a convex region than having to deal with a very irregularly shaped region.

We first express how to test whether the region is convex ($\hat{Q}'_7$), this will help understand the formula that computes the convex hull. The query $\hat{Q}'_7$ can be expressed by the formula

$$\forall\ \triangle_1\ \forall\ \triangle_2\ \forall\ \triangle_3\ \forall\ \triangle_4\ ((\bigwedge_{i=1}^{3} \mathbf{Point}(\triangle_i) \wedge \bigwedge_{i=1}^{3} Q_{11}(\triangle_i) \wedge \mathbf{CornerP}(\triangle_1, \triangle_2, \triangle_3, \triangle_4)) \Rightarrow (\hat{Q}_6(\triangle_4))).$$

Hence, the expression

$$\exists\ \triangle_1\ \exists\ \triangle_2\ \exists\ \triangle_3\ \exists\ \triangle_4\ \exists\ \triangle_5\ \exists\ \triangle_6\ (\bigwedge_{i=1}^{3} \mathbf{Point}(\triangle_i) \wedge \bigwedge_{i=1}^{3} \mathbf{PartOf}(\triangle_i, \triangle_{i+3}) \wedge \bigwedge_{i=4}^{6} \mathbf{Q}_{11}(\triangle_i) \wedge \mathbf{CornerP}(\triangle_1, \triangle_2, \triangle_3, \triangle))$$

defines the query $\hat{Q}_7(\triangle)$. For any three points in some triangles in $\hat{Q}_6$, the triangle connecting them is added to $\hat{Q}_7$. Figure 4 illustrates this. □

Remark 4.3 The first two queries of Example 4.2 ask for relations between regions that can be expressed by the so-called 9-intersection model [10]. This model defines a relation

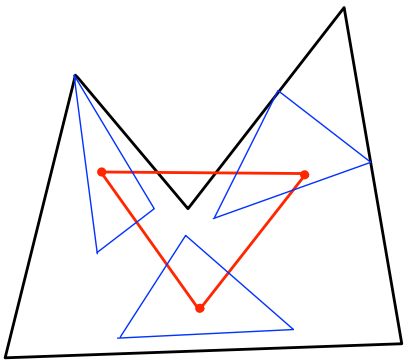

Fig. 4. The convex hull of a set S of triangles is computed by adding all triangles (in red) constructed from three points that are inside (shown with the blue triangles) three triangles of S.

between two regions by investigating the intersections between their boundaries, interiors and exteriors. As the boundary, interior and exterior of a region can be expressed in FO(+, $\times$, $<$, 0, 1, σ), and are affine invariant concepts [1], all relations that can be expressed by the 9-intersection model, can be expressed in FO($\{$**PartOf**$\}$, $\hat{\sigma}$). □

Remark 4.4 As we have remarked before, in the proofs of Lemma 4.2 and Lemma 4.3, we only used the fact that triangles are convex objects having three corner points. It is not difficult to prove that the predicate **PartOf** can be generalized to a predicate $\mathbf{PartOf}^{(n,k)}$, which arguments are n-dimensional convex objects with k corner points (called (n,k)-objects) and that the language FO($\{\mathbf{PartOf}^{(n,k)}\}$) is sound and complete for the first-order affine-generic queries on (n,k)-objects. □

In the context of this remark, we also want to refer to the work of Aiello and van Benthem [1,2] on modal logics of space. They first propose a topological modal logic over regions, which can express "connectedness" and "parthood". By adding a "convexity" operator (expressed using a "betweenness" operator), they obtain an affine modal logic. In [2], the authors also motivate the use of finite unions of convex sets as basic elements for spatial reasoning. They argue that it is a very natural way for people to reason about objects. A fork, for example, is described as the union of its prongs and its handle.

5 Affine-Invariant Similarity of Patterned Triangles

In many applications, we have to consider not only the shape and the color but also the patterns that may be present on the surface of the objects. Barcodes provide an inspiration for robust affine-invariant similarity measures between pairs of patterned objects.

[1] To be exact, they are topological concepts. The affinities of the plane are a subgroup of the homeomorphisms of the plane, so the invariance under the boundary and interior operations carry over naturally.

Barcodes encode information in a robust machine-readable way using sequences of dark and light bars with different widths. The optical scanners which read barcodes are quite robust and already allow the presentation of barcodes from slightly different angles.

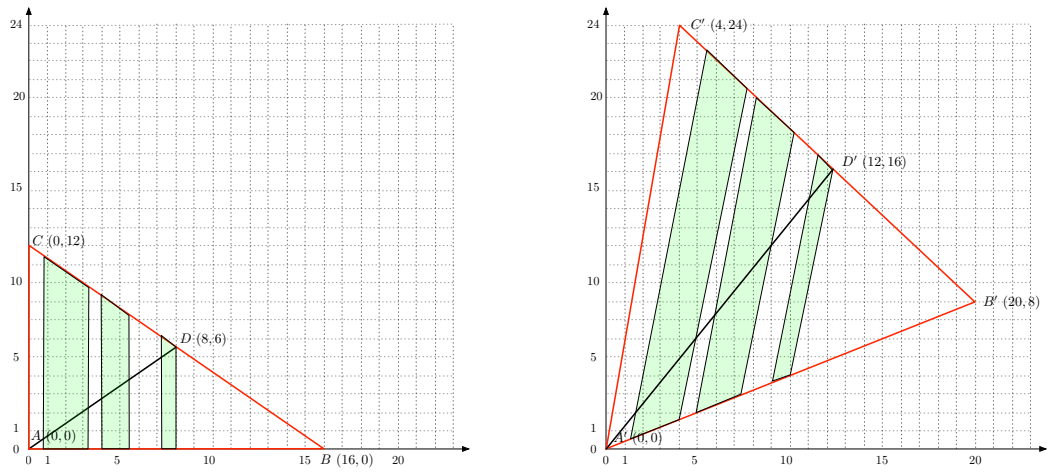

Fig. 5. A striped triangle ABC (left) and its affine-transformation $A'B'C'$ (right).

Although natural objects do not have barcodes, they have rich patterned and textured surfaces with a variety of colors. In such cases, the surface of objects can be broken into a set of triangles that each contain some interesting pattern. For example, consider the left side of Figure 5,which shows a triangular land area ABC that contains corn and wheat parcels, which are seen as green and white stripes, respectively, in an areal image. Suppose that we view ABC from a different point in the air, where the new image $A'B'C'$ (see the right side of Figure 5) will be the following affine transformation of the previous image:

$$f(x,y) = \begin{pmatrix} \frac{5}{4} & \frac{1}{3} \\ \\ \frac{1}{2} & 2 \end{pmatrix} \begin{pmatrix} x \\ \\ y \end{pmatrix}$$

Although the shape of triangle ABC becomes highly distorted, one can identify the point D as the midpoint on the edge CD in triangle ABC. Similarly, one can identify also the corresponding midpoint D' on the edge $C'D'$.

Now let us consider scanning the line segment AD from point A to point D. During the scan one sees a series of lighter and darker areas. In particular, one sees in sequence 1 unit light area, 3 units dark area, 1 unit light area, 2 units dark area, 2 units light area, and 1 unit dark area. Hence the barcode of the AD segment can be represented as $(1, \mathbf{3}, 1, \mathbf{2}, 2, \mathbf{1})$, with the boldface numbers representing the dark areas.

Interestingly, when one scans the line segment $A'D'$, one finds also a similar sequence of

light and dark areas. In particular, one sees 2 units light area, 6 units dark area, 2 unit light area, 4 units dark area, 4 units light area, and 2 units dark area. Hence the barcode of $A'D'$ can be represented as $(2, \mathbf{6}, 2, \mathbf{4}, 4, \mathbf{2})$.

The two barcodes are similar because they have the same number of light and dark areas and those have the same length ratios, which are equal to the ratio of the lengths of the two line segments.

Let the length of a line segment l be denoted as $length(l)$. The similarity of barcodes is a general feature of affine transformations as expressed in Theorem 5.1.

Theorem 5.1 *Let $(a_1, \ldots, a_n)$ and $(b_1, \ldots, b_m)$ be the barcodes of two corresponding line segments l_1 and l_2 in two affine transformations of a patterned triangle. Then $n = m$, and the following hold for each $1 \leq i \leq n$:*

$$\frac{a_i}{b_i} = \frac{length(l_1)}{length(l_2)}$$

Example 5.1 Consider again the barcodes of AD and $A'D'$ of the two affine transformations shows in Figure 5. In this case the barcode of AD is:

$$(a_1, a_2, a_3, a_4, a_5, a_6) = (1, \mathbf{3}, 1, \mathbf{2}, 2, \mathbf{1})$$

and the barcode of $A'D'$ is:

$$(b_1, b_2, b_3, b_4, b_5, b_6) = (2, \mathbf{6}, 2, \mathbf{4}, 4, \mathbf{2})$$

We have that $length(AD) = 10$ and $length(A'D') = 20$. Further, as expected by Theorem 5.1,

$$\frac{a_i}{b_i} = \frac{10}{20} = 0.5 \qquad \text{for all } i, \quad 1 \leq i \leq 6$$

5.1 Similarity Measures for Barcodes

Let us now consider a similarity measure for two barcodes. Let $\overline{a} = (a_1, \ldots, a_n)$ and $\overline{b} = (b_1, \ldots, b_m)$ be the barcodes of two corresponding line segments l_1 and l_2 in two affine transformations of a patterned triangle. If $n \neq m$, then we simply say that the two barcodes are not similar. Otherwise, let

$$d = \frac{length(l_1)}{length(l_2)}$$

Equalize the total length of the two line segments by scaling l_2 by the factor d. After scaling we obtain a barcode $\overline{c} = (c_1, \ldots, c_n)$ where each $c_i = b_i \times d$ for $1 \leq i \leq n$.

Next compare $\overline{a}$ and $\overline{c}$. If one is an affine transformation of the other, then by Theorem 5.1 and the choice of the scaling factor d, the following holds:

$$\frac{a_i}{c_i} = \frac{a_i}{b_i \times d} = \frac{length(l_1)}{length(l_2) \times d} = 1 \qquad \forall \;\; 1 \leq i \leq n$$

In general, one cannot expect two barcodes to be perfect affine transformations of each other, that is to have $a_i = c_i$ for each $1 \leq i \leq n$. Hence one needs to consider how much a_i and c_i deviate from each other. One can use a root mean square error measurement as follows:

$$E(\overline{a}, \overline{c}) = \sqrt{\frac{\sum_{k=1}^{n} (a_i - c_i)^2}{n}}$$

Example 5.2 Let $\overline{a} = (1, \mathbf{2.5}, 1, \mathbf{2}, 2.5, \mathbf{1})$ and $\overline{b} = (2, \mathbf{5.5}, 2, \mathbf{4.5}, 4, \mathbf{2})$. Then

$$length(\overline{a}) = 1 + 2.5 + 1 + 2 + 2.5 + 1 = 10$$

and

$$length(\overline{b}) = 2 + 5.5 + 2 + 4.5 + 4 + 2 = 20.$$

Hence

$$d = \frac{length(a)}{length(b)} = \frac{10}{20} = 0.5$$

and

$$\overline{c} = (1, \mathbf{2.75}, 1, \mathbf{2.25}, 2, \mathbf{1}).$$

Further,

$$E(\overline{a}, \overline{c}) = \sqrt{\frac{0^2 + (-.25)^2 + 0^2 + (-.25)^2 + .5^2 + 0^2}{6}} = 0.375$$

Since the root mean square error is small, the two barcodes are quite similar.

5.2 *Different Patterns*

The barcode-based similarity measure can accommodate other patterns beside striped patterns. For example, consider again the land area within triangle ABC but suppose that it is a grass land with a few oval shaped watering pond areas for animals as shown by green color in Figure 6. In this case, the barcodes are still similar after the same affine transformation.

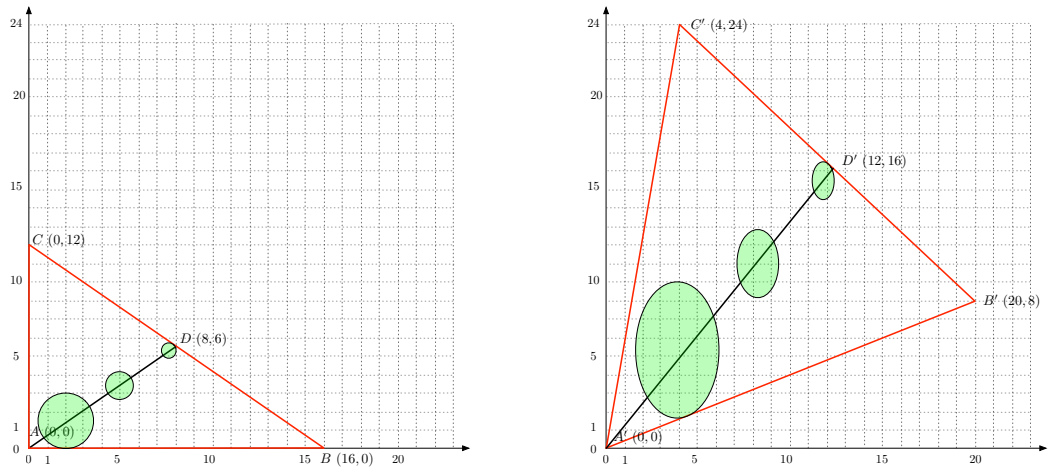

Fig. 6. A spotted triangle ABC (left) and its affine-transformation $A'B'C'$ (right).

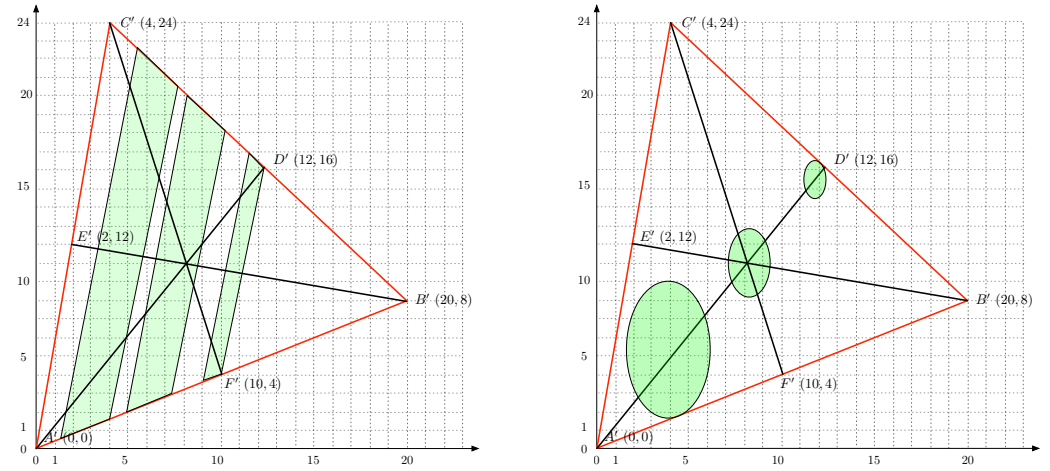

Fig. 7. Midpoints of a striped triangle ABC (left) and a spotted triangle $A'B'C'$ (right).

Note that the barcodes in Figures 5 and 6 are the same. That means that a single barcode for a triangle is incapable of distinguishing between the striped and the spotted patterns within the triangle. However, one can improve the situation by considering not one but two or three barcodes for a single triangle. Each barcode is scanned along the line segment whose endpoints are a corner vertex of the triangle and the midpoint vertex on the opposite side of the triangle. Figure 7 shows the three line segments $A'D'$, $B'E'$ and $C'F'$ in the striped and the spotted triangles.

5.3 Colored Patterns

There are several ways to add more colors than just two to the barcode-based similarity measure. For example, the barcode can be enhanced with a sensor of the color (or average color) in each "bar" instead of sensing only white and black. Assume that in Figure 5 non-white stripes (shown as black) are from left to right red, blue, and red. Then a barcode representation of the striped triangle would be $(1-white, 3-red, 1-white, 2-blue, 2-white, 1-red)$. Such a colored barcode representation would be distinguishable from another colored barcode representation such as $(1-white, 3-blue, 1-white, 2-red, 2-white, 1-blue)$ based purely on the color differences.

5.4 Complex Objects

The above discussion focused on the handling of triangles. Of course, many objects are more complex than triangles. However, the more complex objects can be decomposed into a set of triangles. In fact, a particular triangulation, or subdivision into non-overlapping triangles, can be found that can be shown to be affine-invariant. That is, the same object, no matter what particular affine invariant image is given, can be triangulated into the same set of triangles, such that, the set of triangles are also affine-invariant to each other.

Applying the above affine-invariant triangulation method should be the first step in handling complex objects, once they are identified and separated from the background and other objects. Since each complex object is triangulated also into a set of affine-invariant triangles, to find whether two images are similar, we need to find the best mapping from triangles to triangles in the two sets, then we need to sum the similarity scores.

Example 5.3 *Suppose that a complex object I_1 consists of the following three affine-invariant triangles:*

$$\Delta_{1,1}, \Delta_{1,2}, \Delta_{1,3}.$$

Similarly, suppose that another complex object I_2 consists of the following three affine-invariant triangles:

$$\Delta_{2,1}, \Delta_{2,2}, \Delta_{2,3}.$$

Further suppose that the best mapping (the mapping that yields the lowest sum of the triangle-triangle similarity scores) is the following:

$$\Delta_{1,1} \leftrightarrow \Delta_{2,3}, \quad \Delta_{1,2} \leftrightarrow \Delta_{2,1}, \quad \Delta_{1,3} \leftrightarrow \Delta_{2,2}.$$

Then the similarity score $\mathscr{S}im$ between the two complex objects is the following.

$$\mathscr{S}im(I_1, I_2) = \mathscr{S}im(\Delta_{1,1}, \Delta_{2,3}) + \mathscr{S}im(\Delta_{1,2}, \Delta_{2,1}) + \mathscr{S}im(\Delta_{1,3}, \Delta_{2,2}).$$

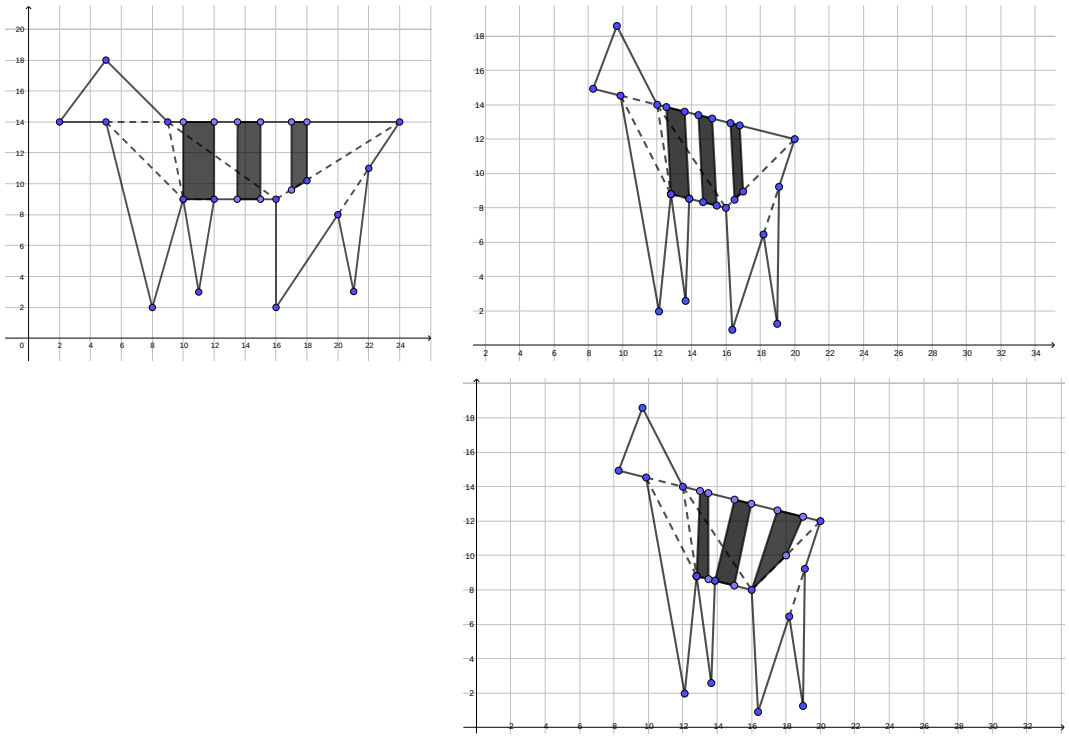

Fig. 8. Three zebra images top left (A), top right (B) and bottom right (C).

When the image is a convex polygon that is the silhouette of an object, then we can apply Delauney triangulation that generates a unique triangulation for every polygon. Even when the polygon is concave, there may be obvious choice in breaking it into a set of convex polygons, and then applying to each a Delauney triangulation. Hence two silhouette images can be compared as two sets of triangles. If a large number of the triangles can be shown to be affine invariant copies of each other, then we can say that the two images are affine-invariant.

When the images are not only silhouettes but also contain some patterns, then the affine-invariance test has two be performed into steps. In the first step, we test only the affine-invariance of the silhouettes. If that succeeds, then we can proceed to the second step, which is testing the affine-invariance with respect to the patterns. For corresponding triangles, the patterns that they contain have to be shown to be affine-invariant of each other.

Example 5.4 *There are cases when the first step of the affine-invariance test succeeds, but the second step fails. Consider the three zebra-like images shown in Figure 8. Like in Figure 5, in these three areal images the zebra stripes are corn fields and the other areas of the zebra are wheat fields. Hence these areal images are similar to Nazca Indian drawings, which can be seen from great distances. Here the A and B are affine-invariant images of each other according to both steps, while A and C are not affine-invariant because they*

fail the second step.

As a practical matter, when the complex images are composed of a large number of triangles, then we cannot expect the affine-invariant triangulation to find for two different images two different triangulations that are perfectly mappable into each other. It could well happen, for instance, that one triangulation yields 90 triangles and another triangulation yields 100. Obviously, in this case at least ten triangles in the first triangulation cannot be mapped into the other. Such an imperfect match can be expected between two images of the land surface, when in the second image a flood occurs. However, notice that even in the case of partial flooding, many of the triangles associated with the dry areas can be identified and mapped to each other. Therefore, it may be possible to find in general a number k, such that if k triangles can be mapped into each other by an affine transformation, then they are images of the same land area. It is at that point that the study of their differences, i.e., the changes over time, become interesting. In the flood example, we may naturally want to compute the total area flooded. That natural problem would need some extra operators beyond the ones we already have in our language, in particular an area-ratio operator. Note that affine-invariance preserves the area ratios. Hence if in any of the images we can calculate the ratio of the area flooded to the total area of a state, which can be assumed to be a fixed already known constant, then the total flooded area can be calculated too.

6 Conclusions and Future Work

The use of triangles instead of points or real numbers is motivated by the spatial practice, where data is often represented as a collection of triangles. In this paper, we introduced the new triangle-based query language FO({**PartOf**}) that fits well with this practice. We showed that our query language has the same expressiveness as the affine-invariant FO({**Between**})-queries on triangle databases. We did this by showing that our language is sound and complete for the FO({**Between**})-queries on triangle databases. In addition, we gave several examples to illustrate the expressiveness of the triangle-based language and the ease of use of manipulating triangles.

We then turned to the notion of safety. We showed that, although we cannot decide whether a particular Tquery returns a finite output given a finite input, we can decide whether the output is finite. We also extended this finiteness to the more intuitive notion of sets that have a finite representation. We proved that we can decide whether the output of a query has a finite representation and compute such a finite representation in FO({**PartOf**}).

Geerts, Haesevoets and Kuijpers [11] already proposed point-based languages for several classes of spatio-temporal queries. The data model used there represented a moving two-dimensional object as a collection of points in three-dimensional space. There exist however, data models that represent spatio-temporal data as a collection of moving

objects (see for example [6,7]), which is more natural. Hence, a *moving triangle*-based language with the same expressiveness as the spatio-temporal point languages mentioned above would be much more useful in practice, but that remains an open problem.

References

[1] M. Aiello and J. van Benthem. Logical patterns in space. In D. Barker-Plummer, D. Beaver, J. van Benthem, and P. Scotto di Luzio, editors, *Words, Proofs, and Diagrams*, pages 5–25. CSLI, 2002.

[2] M. Aiello and J. van Benthem. A modal walk through space. *Journal of Applied Non-Classical Logics*, 12(3-4):319–363, 2002.

[3] S. Anderson and P. Z. Revesz. Efficient maxcount and threshold operators of moving objects. *Geoinformatica*, 13(4):355–396, 2009.

[4] M. De Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf. *Computational Geometry.* Springer-Verlag, 2000.

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

[6] C. X. Chen and C. Zaniolo. SQLST: A spatio-temporal data model and query language. In V. C. Storey A. H. F. Laender, S. W. Liddle, editor, *Conceptual Modeling, 19th International Conference on Conceptual Modeling (ER'00)*, volume 1920 of *Lecture Notes in Computer Science*, pages 96–111. Springer-Verlag, 2000.

[7] J. Chomicki, S. Haesevoets, B. Kuijpers, and P. Z. Revesz. Classes of spatiotemporal objects and their closure properties. *Annals of Mathematics and Artificial Intelligence*, 39(4):431–461, 2003.

[8] J. Chomicki and P. Z. Revesz. Constraint-based interoperability of spatiotemporal databases. *Geoinformatica*, 3(3):211–243, 1999.

[9] S. Daggumati, P. Z. Revesz, and C. Svehla. The A-Maze-D advanced maze development system for fast game design and implementation. *International Journal of Systems Applications, Engineering and Development*, 10:195–204, 2016.

[10] M. Egenhofer and J. Herring. A mathematical framework for the definition of topological relationships. In K. Brassel and H. Kishimoto, editors, *Proceedings of the Fourth International Symposium on Spatial Data Handling*, pages 803–813, 1990.

[11] F. Geerts, S. Haesevoets, and B. Kuijpers. A theory of spatio-temporal database queries. In G. Ghelli and G. Grahne, editors, *Database Programming Languages, 8th International Workshop, DBPL 2001*, volume 2397 of *Lecture Notes in Computer Science*, pages 198–212. Springer, 2002.

[12] M. Gyssens, J. Van den Bussche, and D. Van Gucht. Complete geometric query languages. *Journal of Computer and System Sciences*, 58(3):483–511, 1999.

[13] S. Haesevoets, B. Kuijpers, and P. Z. Revesz. Affine-invariant triangulation of spatio-temporal data with an application to image retrieval. *ISPRS International Journal of Geo-Information*, 6(4):100, 2017.

[14] M. Hagedoorn and R. C. Veldkamp. Reliable and efficient pattern matching using an affine invariant metric. *International Journal of Computer Vision*, 31:203–225, 1999.

[15] D.P. Huttenlocher, G.A. Klauderman, and W.J. Rucklidge. Comparing images using the Hausdorff distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15:850–863, 1998.

[16] P. C. Kanellakis, G. M. Kuper, and P.Z. Revesz. Constraint query languages. *Journal of Computer and System Sciences*, 51:26–52, 1995.

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

[18] Y. Lamdan, J.T. Schwartz, and H.J. Wolfson. Affine-invariant model-based object recognition. *IEEE Journal of Robotics and Automation*, 6:578–589, 1990.

[19] R. Laurini and D. Thompson. *Fundamentals of Spatial Information Systems*. Number 37 in APIC Series. Academic Press, 1992.

[20] L. Li and P. Z. Revesz. Interpolation methods for spatio-temporal geographic data. *Computers, Environment and Urban Systems*, 28(3):201–227, 2004.

[21] L. Matos, J. Moreira, and A. Carvalho. Representation and management of spatiotemporal data in object-relational databases. In *Proceedings of the 27th Annual ACM Symposium on Applied Computing*, New York, NY, USA, 2012. ACM Press.

[22] G. Nielson. A characterization of an affine invariant triangulation. In G. Farin, H. Hagen, and H. Noltemeier, editors, *Geometric Modelling, Computing Supplementum 8*, pages 191–210, 1993.

[23] C.H. Papdimitriou, D. Suciu, and V. Vianu. Topological queries in spatial databases. In *Proceedings of the 15th ACM Symposium on Principles of Database Systems*, pages 81–92. ACM Press, 1996.

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

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

[26] Dan Raviv and Ron Kimmel. Affine invariant geometry for non-rigid shapes. *International Journal of Computer Vision*, 111(1):1–11, 2015.

[27] P. Z. Revesz. *Introduction to Constraint Databases*. Springer-Verlag, 2002.

[28] P. Z. Revesz. Robust affine-invariant similarity measures for patterned triangles. In *10th IASTED Int. Conf. on Computer Graphics and Imaging*, pages 292–297. ACTA Press, 2008.

[29] P. Z. Revesz. *Introduction to Databases: From Biological to Spatio-Temporal.* Springer-Verlag, 2010.

[30] Oncel Tuzel, Tim K. Marks, and Salil Tambe. Robust face alignment using a mixture of invariant experts. In *European Conference on Computer Vision*, pages 825–841. Springer-Verlag, 2016.

[31] H. Yue, L. R. Rilett, and P. Z. Revesz. Spatio-temporal traffic video data archiving and retrieval system. *Geoinformatica*, 20(1):59–94, 2016.

Appendix

We collect below some of the technical proofs, starting with the proof of the soundness lemma.

Proof of Lemma 4.2. Let $\hat{\sigma} = \{\hat{R}_1, \hat{R}_2, \ldots, \hat{R}_m\}$ be a triangle database schema. Let $\dot{R}_i$ be the corresponding spatial point relation names of arity $3 \cdot ar(\hat{R}_i)$, for $1 \leq i \leq m$, and let $\dot{\sigma}$ be the corresponding point database schema $\{\dot{R}_1, \dot{R}_2, \ldots, \dot{R}_m\}$. We translate each formula $\hat{\varphi}$ of FO($\{$**PartOf**$\}$, $\hat{\sigma}$) into an equivalent formula in FO($\{$**Between**$\}$, $\dot{\sigma}$). We do this by induction on the structure of FO($\{$**PartOf**$\}$, $\hat{\sigma}$)-formulas.

First, we translate the variables of $\hat{\varphi}$. Each triangle variable $\triangle$ is naturally translated into three spatial point variables p_1, p_2 and p_3. We allow one or more of the corner points of a triangle to coincide, so there are no further restrictions on the variables p_j, for $1 \leq i \leq 3$.

The atomic formulas of FO($\{$**PartOf**$\}$, $\hat{\sigma}$) are equality expressions on triangle variables, expressions of the form **PartOf**$(\triangle_1, \triangle_2)$, and expressions of the form $\hat{R}_i(\triangle_1, \triangle_2, \ldots, \triangle_{k_i})$, where $k_i = ar(\hat{R}_i)$, for $1 \leq i \leq m$. More complex formulas can be constructed using the Boolean operators $\wedge$, $\vee$ and $\neg$ and existential quantification.

The translation of atomic formulas.

We first show that all atomic formulas of FO($\{$**PartOf**$\}$, $\hat{\sigma}$) can be expressed in the language FO($\{$**Between**$\}$, $\dot{\sigma}$).

(i) The translation of $(\triangle_1{=}\triangle_2)$, where $\triangle_1$ is translated into $p_{1,1}$, $p_{1,2}$ and $p_{1,3}$ and $\triangle_2$ is translated into $p_{2,1}$, $p_{2,2}$ and $p_{2,3}$, equals

$$\bigvee_{\sigma(1,2,3)=(j_1,j_2,j_3),\sigma\in\mathscr{S}_3} (p_{1,1} = p_{2,j_1} \wedge p_{1,2} = p_{2,j_2} \wedge p_{1,3} = p_{2,j_3}),$$

where $\mathscr{S}_3$ is the set of all permutations of $\{1, 2, 3\}$. This formula expresses, that the corner points of $\triangle_1$ and $\triangle_2$ coincide, when taken in the right order. The correct-

ness of this translation follows trivially from the definition of triangle equality (see Definition 4.2).

(ii) The translation of $\mathbf{PartOf}(\triangle_1, \triangle_2)$, where $\triangle_1$ is translated into $p_{1,1}$, $p_{1,2}$ and $p_{1,3}$ and $\triangle_2$ is translated into $p_{2,1}$, $p_{2,2}$ and $p_{2,3}$, is

$$\bigwedge_{i=1}^{3} \mathbf{InTriangle}(p_{1,i}, p_{2,1}, p_{2,2}, p_{2,3}),$$

where the definition of **InTriangle** is:

$$\mathbf{InTriangle}(p, p_1, p_2, p_3) := \exists\, p_4(\mathbf{Between}(p_1, p_4, p_2) \wedge \mathbf{Between}(p_4, p, p_3)).$$

Figure 9 illustrates the corresponding geometric construction. The correctness of this translation follows from the definition of the predicate **PartOf** (see Definition 4.1).

(iii) The translation of $\hat{R}_i(\triangle_1, \triangle_2, \ldots, \triangle_k)$, where $\triangle_i$ is translated into $p_{i,1}$, $p_{i,2}$ and $p_{i,3}$ for $1 \leq i \leq k$, is

$$\dot{R}_i(p_{1,1}, p_{1,2}, p_{1,3}, p_{2,1}, p_{2,2}, p_{2,3}, \ldots, p_{k,1}, p_{k,2}, p_{k,3}).$$

The correctness of this translation follows from Definition 3.2 and Remark 3.1.

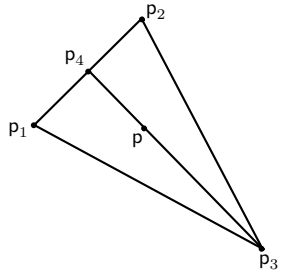

Fig. 9. In this illustration, the expression $\mathbf{InTriangle}(\mathsf{p}, \mathsf{p}_1, \mathsf{p}_2, \mathsf{p}_3)$ is true because there exists a point p_4 between p_1 and p_2 such that p lies between p_4 and p_3.

The translation of composed formulas.

Assume that we already correctly translated the FO($\{\mathbf{PartOf}\}$, $\hat{\sigma}$)-formulas $\hat{\varphi}$ and $\hat{\psi}$ into the FO($\{\mathbf{Between}\}$, $\dot{\sigma}$)-formulas $\dot{\varphi}$ and $\dot{\psi}$. Suppose that the number of free variables in $\hat{\varphi}$ is k_φ and that of $\hat{\psi}$ is k_ψ. Therefore, we can assume that, for each triangle database $\hat{\mathscr{D}}$ over the input schema $\hat{\sigma}$, and for each k_φ-tuple of triangles $(\mathsf{T}_1, \mathsf{T}_2, \ldots, \mathsf{T}_{k_\varphi})$ given as $((\mathsf{p}_{1,1}, \mathsf{p}_{1,2}, \mathsf{p}_{1,3}), (\mathsf{p}_{2,1}, \mathsf{p}_{2,2}, \mathsf{p}_{2,3}), \ldots, (\mathsf{p}_{k_\varphi,1}, \mathsf{p}_{k_\varphi,2}, \mathsf{p}_{k_\varphi,3}))$ that

$$\hat{\mathscr{D}} \models \hat{\varphi}[\mathsf{T}_1, \mathsf{T}_2, \ldots, \mathsf{T}_{k_\varphi}] \text{ if and only if}$$
$$\dot{\mathscr{D}} \models \dot{\varphi}[\mathsf{p}_{1,1}, \mathsf{p}_{1,2}, \mathsf{p}_{1,3}, \mathsf{p}_{2,1}, \mathsf{p}_{2,2}, \mathsf{p}_{2,3}, \ldots, \mathsf{p}_{k_\varphi,1}, \mathsf{p}_{k_\varphi,2}, \mathsf{p}_{k_\varphi,3}]$$

is true when $\dot{\mathscr{D}}$ is the point database over the input schema $\dot{\sigma}$, obtained from $\hat{\mathscr{D}}$ by applying the canonical bijection can_{tr} between $\left((\mathbb{R}^2)^3\right)^{k_\varphi}$ and $(\mathbb{R}^2)^{3k_\varphi}$, on $\hat{\mathscr{D}}$. For the formula $\hat{\psi}$ the analog holds.

In the following, we omit the k_φ-tuples (resp., k_ψ-tuples) of triangles and $3k_\varphi$-tuples (resp., $3k_\psi$-tuples) of points on which the formulas are applied, to make the proofs more readable.

(i) The translation of $\hat{\varphi} \wedge \hat{\psi}$ is $\dot{\varphi} \wedge \dot{\psi}$. Indeed,

$\dot{\mathscr{D}} \models (\dot{\varphi} \wedge \dot{\psi})$

iff. $\dot{\mathscr{D}} \models \dot{\varphi}$ and $\dot{\mathscr{D}} \models \dot{\psi}$

iff. $\hat{\mathscr{D}} \models \hat{\varphi}$ and $\hat{\mathscr{D}} \models \hat{\psi}$

iff. $\hat{\mathscr{D}} \models (\hat{\varphi} \wedge \hat{\psi})$.

(ii) The translation of $\hat{\varphi} \vee \hat{\psi}$ is $\dot{\varphi} \vee \dot{\psi}$. Indeed,

$\dot{\mathscr{D}} \models (\dot{\varphi} \vee \dot{\psi})$

iff. $\dot{\mathscr{D}} \models \dot{\varphi}$ or $\dot{\mathscr{D}} \models \dot{\psi}$

iff. $\hat{\mathscr{D}} \models \hat{\varphi}$ or $\hat{\mathscr{D}} \models \hat{\psi}$

iff. $\hat{\mathscr{D}} \models (\hat{\varphi} \vee \hat{\psi})$.

(iii) The translation of $\neg\hat{\varphi}$ is $\neg\dot{\varphi}$. Indeed,

$\dot{\mathscr{D}} \models \neg\dot{\varphi}$

iff. it is not true that $\dot{\mathscr{D}} \models \dot{\varphi}$

iff. it is not true that $\hat{\mathscr{D}} \models \hat{\varphi}$

iff. $\hat{\mathscr{D}} \models \neg\hat{\varphi}$.

(iv) Assume that $\hat{\varphi}$ has free variables $\triangle, \triangle_1, \ldots, \triangle_k$ and $\triangle$ is translated into p_1, p_2, p_3 and $\triangle_i$ is translated into $p_{i,1}$, $p_{i,2}$, $p_{i,3}$, for $1 \leq i \leq k$. The translation of

$$\exists\ \triangle\ \ \hat{\varphi}(\triangle, \triangle_1, \triangle_2, \ldots, \triangle_k)$$

$$\text{is } \exists p_1\, \exists p_2\, \exists p_3\, \dot{\varphi}(p_1, p_2, p_3, p_{1,1}, p_{1,2}, p_{1,3}, \ldots, p_{k,1}, p_{k,2}, p_{k,3}).$$

Indeed,

$\dot{\mathscr{D}} \models \exists p_1\, \exists p_2\, \exists p_3\, \dot{\varphi}(p_1, p_2, p_3)[\mathsf{p}_{1,1}, \mathsf{p}_{1,2}, \mathsf{p}_{1,3}, \ldots, \mathsf{p}_{k,1}, \mathsf{p}_{k,2}, \mathsf{p}_{k,3}]$

iff. there exist points p_1, p_2, p_3 in $\mathbb{R}^2$ such that

$\dot{\mathscr{D}} \models \dot{\varphi}[\mathsf{p}_1, \mathsf{p}_2, \mathsf{p}_3, \mathsf{p}_{1,1}, \mathsf{p}_{1,2}, \mathsf{p}_{1,3}, \ldots, \mathsf{p}_{k,1}, \mathsf{p}_{k,2}, \mathsf{p}_{k,3}]$

iff. there exists a triangle T such that $\hat{\mathscr{D}} \models \hat{\varphi}[\mathsf{T}, \mathsf{T}_1, \ldots, \mathsf{T}_k]$, where

T_i is the triangle with corner points $\mathsf{p}_{i,1}$, $\mathsf{p}_{i,2}$ and $\mathsf{p}_{i,3}$ for $1 \leq i \leq k$

iff. $\hat{\mathscr{D}} \models \exists\mathsf{T}\, \hat{\varphi}(\mathsf{T})[\mathsf{T}_1, \ldots, \mathsf{T}_k]$.

To summarize the proof strategy, let $\hat{\sigma} = \{\hat{R}_1, \hat{R}_2, \ldots, \hat{R}_m\}$ be a triangle database schema. Let $\dot{\sigma} = \{\dot{R}_1, \dot{R}_2, \ldots, \dot{R}_m\}$ be the corresponding point database schema. Each formula $\hat{\varphi}$ in FO($\{\mathbf{PartOf}\}$, $\hat{\sigma}$), with free variables $\triangle_1, \triangle_2, \ldots, \triangle_k$ can be translated into

a FO({**Between**}, $\dot{\sigma}$)-formula $\dot{\varphi}$ with free variables $p_1, p_2, p_3, p_{1,1}, p_{1,2}, p_{1,3}, p_{2,1}, p_{2,2}, p_{2,3}$, $\ldots, p_{k,1}, p_{k,2}, p_{k,3}$. This translation is such that, for all triangle databases $\hat{\mathscr{D}}$ over $\hat{\sigma}$, $\hat{\mathscr{D}} \models \hat{\varphi}$ iff. $\dot{\mathscr{D}} \models \dot{\varphi}$. Here, $\dot{\mathscr{D}}$ is the point database over $\dot{\sigma}$ which is the image of $\hat{\mathscr{D}}$ under the canonical bijection between $\left((\mathbb{R}^2)^3\right)^k$ and $(\mathbb{R}^2)^{3k}$. This completes the soundness proof. □

Next, we give the proof of the completeness lemma.

Proof of Lemma 4.3. Let $\hat{\sigma} = \{\hat{R}_1, \hat{R}_2, \ldots, \hat{R}_m\}$ be a triangle database schema and $\dot{\sigma}$ be the corresponding point database schema. We have to prove that we can translate every triangle database query, expressed in the language FO({**Between**}, $\dot{\sigma}$), into a triangle database query in the language FO({**PartOf**}, $\hat{\sigma}$) over triangle databases.

We first show how we can simulate point variables by a degenerated triangle, and any FO({**Between**}, $\dot{\sigma}$)-formula $\dot{\varphi}(p_1, p_2, \ldots, p_k)$ by a formula $\varphi(\triangle_1, \triangle_2, \ldots, \triangle_k)$, where $\triangle_1$, $\triangle_2, \ldots, \triangle_k$ represent triangles that are degenerated into points. We prove this by induction on the structure of FO({**Between**}, $\dot{\sigma}$)-formulas. Initially, each FO({**Between**}, $\dot{\sigma}$)-formula $\dot{\varphi}(p_1, p_2, \ldots, p_k)$ will be translated into a FO({**PartOf**}, $\hat{\sigma}$)-formula $\hat{\varphi}(\triangle_1, \triangle_2$, $\ldots, \triangle_k)$ with the same number of free variables.

The translation of a point variable p is the triangle variable $\triangle$, and we add the condition **Point**($\triangle$) as a conjunct to the beginning of the translation of the formula. The definition of **Point**($\triangle$) is

$$\forall\ \triangle'\ (\mathbf{PartOf}(\triangle', \triangle) \rightarrow (\triangle{=}\triangle')).$$

In the following, we always assume that such formulas **Point**($\triangle$) are already added to the translation as a conjunct.

The translation of atomic formulas.

The atomic formulas of the language FO({**Between**}, $\dot{\sigma}$) are equality constraints on point variables, formulas of the form **Between**(p_1, p_2, p_3), and formulas of the type $\dot{R}_i(p_1, p_2, \ldots, p_{k_i})$, where $k_i = 3 \cdot ar(\hat{R}_i)$. We show that all of those can be simulated by an equivalent formula of FO({**PartOf**}, $\hat{\sigma}$).

(i) The translation of $(p_1 = p_2)$ is $(\triangle_1{=}_\triangle\triangle_2)$.

(ii) The translation of **Between**(p_1, p_2, p_3), where $\triangle_1$, $\triangle_2$ and $\triangle_3$ (which as assumed are already declared points) are the translations of p_1, p_2 and p_3, respectively, is expressed by saying that all triangles that contain both $\triangle_1$ and $\triangle_3$ should also contain $\triangle_2$. It then follows from the convexity of triangles (or line segments, in the degenerated case) that $\triangle_2$ lies on the line segment between $\triangle_1$ and $\triangle_3$. Figure 10 illustrates this principle. We now give the formula translating **Between**(p_1, p_2, p_3):

$$\forall\ \triangle_4\ ((\mathbf{PartOf}(\triangle_1, \triangle_4) \wedge \mathbf{PartOf}(\triangle_3, \triangle_4)) \rightarrow \mathbf{PartOf}(\triangle_2, \triangle_4)).$$

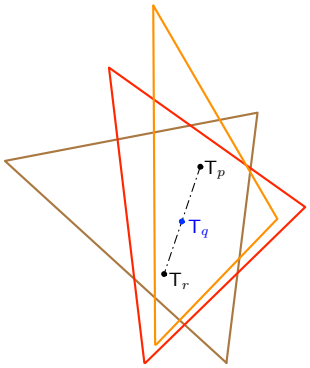

Fig. 10. Illustration of the translation of the predicate **Between**. The (degenerated) triangle T_q (in blue) lies between the (degenerated) triangles T_p and T_r if and only if all triangles (illustrated by the orange, red and brown triangles) that contain both T_p and T_r, also contain T_q.

The correctness of this translation follows from the fact that triangles are convex objects.

(iii) Let $\dot{R}_j$ be a relation name from $\dot{\sigma} = \{\dot{R}_1, \dot{R}_2, \ldots, \dot{R}_m\}$. Let $ar(\hat{R}_j) = k_j$ and thus $ar(\dot{R}_j) = 3k_j$, for $1 \leq j \leq m$. The translation of $\dot{R}_j(p_{1,1}, p_{1,2}, p_{1,3}, p_{2,1}, p_{2,2}, p_{2,3}, \ldots, p_{k,1}, p_{k,2}, p_{k,3})$ is:

$$\exists\, \triangle_1\, \exists\, \triangle_2 \ldots \exists\, \triangle_k\, (\hat{R}_j(\triangle_1, \triangle_2, \ldots, \triangle_k) \wedge \bigwedge_{i=1}^{k} \mathbf{CornerP}(\triangle_{i,1}, \triangle_{i,2}, \triangle_{i,3}, \triangle_i)).$$

The definition of **CornerP** is:

$$\mathbf{CornerP}(\triangle_1, \triangle_2, \triangle_3, \triangle) := \forall\, \triangle_4\ ((\mathbf{Point}(\triangle_4) \wedge \mathbf{PartOf}(\triangle_4, \triangle)) \rightarrow \mathbf{Intriangle}_{\triangle}(\triangle_4, \triangle_1, \triangle_2, \triangle_3)).$$

The predicate $\mathbf{InTriangle}_{\triangle}$ is the translation of the predicate **InTriangle** of the language FO({**Between**}) as described in the proof of Lemma 4.2, into FO({**PartOf**}). The FO({**Between**}) formula expressing **InTriangle** only uses **Between**. In the previous item of this proof, we already showed how this can be translated into FO({**PartOf**}).

Given a $(3k)$-tuple of points $(\mathsf{p}_{1,1}, \mathsf{p}_{1,2}, \mathsf{p}_{1,3}, \mathsf{p}_{2,1}, \mathsf{p}_{2,2}, \mathsf{p}_{2,3}, \ldots, \mathsf{p}_{a,1}, \mathsf{p}_{k,2}, \mathsf{p}_{k,3})$ in $\mathbb{R}^2$. There will be 6^k k-tuples of triangles $(\mathsf{T}_1, \mathsf{T}_2, \ldots, \mathsf{T}_k)$ such that, for each of the T_i, $1 \leq i \leq k$, the condition $\mathbf{CornerP}(\mathsf{T}_{i,1}, \mathsf{T}_{i,2}, \mathsf{T}_{i,3}, \mathsf{T}_i)$ is true. However, there will only be one tuple of triangles that is the image of the $3k$-tuple of points $(\mathsf{p}_{1,1}, \mathsf{p}_{1,2}, \mathsf{p}_{1,3}, \mathsf{p}_{2,1}, \mathsf{p}_{2,2}, \mathsf{p}_{2,3}, \ldots, \mathsf{p}_{a,1}, \mathsf{p}_{k,2}, \mathsf{p}_{k,3})$ under the inverse of the canonical bijection can_{tr}. This shows that the simulation is correct.

The translation of composed formulas.

Now suppose that we already simulated the FO({**Between**}, $\dot{\sigma}$) formulas $\dot{\varphi}(p_1, p_2, \ldots, p_{k_\varphi})$ and $\dot{\psi}(p_1, p_2, \ldots, p_{k_\psi})$ into formulas $\hat{\varphi}$ and $\hat{\psi}$ in FO({**PartOf**}, $\hat{\sigma}$) with free vari-

ables $\triangle_1$, $\triangle_2$, ..., $\triangle_{k_\varphi}$ and $\triangle'_1$, $\triangle'_2$, ..., $\triangle'_{k_\psi}$, respectively. Hence, we can assume that, for each triangle database $\hat{\mathscr{D}}$ over $\hat{\sigma}$ and for each k_φ-tuple of triangles $(\mathsf{T}_1, \mathsf{T}_2, \ldots, \mathsf{T}_{k_\varphi})$ $= ((\mathsf{p}_1, \mathsf{p}_1, \mathsf{p}_1), (\mathsf{p}_2, \mathsf{p}_2, \mathsf{p}_2), \ldots, (\mathsf{p}_{k_\varphi}, \mathsf{p}_{k_\varphi}, \mathsf{p}_{k_\varphi}))$, which are required to be degenerated into points, that

$$\hat{\mathscr{D}} \models \hat{\varphi}[\mathsf{T}_1, \mathsf{T}_2, \ldots, \mathsf{T}_{k_\varphi}] \text{ iff. } \dot{\mathscr{D}} \models \dot{\varphi}[\mathsf{p}_1, \mathsf{p}_2, \ldots, \mathsf{p}_{k_\psi}].$$

For $\hat{\psi}$ we have analogue conditions.

The composed formulas $\dot{\varphi} \wedge \dot{\psi}$, $\dot{\varphi} \vee \dot{\psi}$, $\neg\dot{\varphi}$ and $\exists p\, \dot{\varphi}$, are translated into $\hat{\varphi} \wedge \hat{\psi}$, $\hat{\varphi} \vee \hat{\psi}$, $\neg\hat{\varphi}$ and $\exists \triangle\ (\hat{\varphi})$, respectively if we assume that p is translated into $\triangle$. The correctness proofs for these translations are similar to the proofs in Lemma 4.2. Therefore, we do not repeat them here. This concludes the proof of Lemma 4.3. $\square$