The Area Method: a Recapitulation

Predrag Janicic, Julien Narboux, Pedro Quaresma

The Area Method: a Recapitulation

Predrag Janičić
Faculty of Mathematics, University of Belgrade
Studentski trg 16, 11000 Belgrade, Serbia
e-mail: janicic@matf.bg.ac.rs

Julien Narboux
LSIIT, UMR 7005 CNRS-ULP, University of Strasbourg,
Pôle API, Bd Sébastien Brant, BP 10413, 67412 Illkirch, France
e-mail: Julien.Narboux@lsiit-cnrs.unistra.fr

Pedro Quaresma
CISUC/Department of Mathematics, University of Coimbra
3001-454 Coimbra, Portugal
e-mail: pedro@mat.uc.pt

October 6, 2009

Abstract. The area method for Euclidean constructive geometry was proposed by Chou et al. in the early 1990's. The method produces human-readable proofs and can efficiently prove many non-trivial geometry theorems. It is one of the most interesting and most successful methods for automated theorem proving in geometry and probably the most successful in the domain of automated production of readable proofs in geometry.

In this paper, we provide a first complete presentation of the method. We provide both algorithmic and implementation details that were omitted in the original presentations. We also give a variant of Chou, Gao and Zhang's axiom system. Based on this axiom system, we proved formally all the lemmas needed by the method and its soundness using the *Coq* proof assistant.

To our knowledge, apart from the original implementation by the authors who first proposed the method, there are only three implementations more. Although the basic idea of the method is simple, implementing it is a very challenging task because of a number of details that has to be dealt with. With the description of the method given in this paper, implementing the method should be still complex, but a straightforward task. In the paper we describe all these implementations and also some of their applications.

Keywords: area method, geometry, automated theorem proving

1. Introduction

There are two major families of methods in automated reasoning in geometry: algebraic style and synthetic style methods.

Algebraic style has its roots in the work of Descartes and in the translation of geometry problems to algebraic problems. The automation of the proving process along this line began with the quantifier elimination method of Tarski (Tarski, 1951) and since then had many improvements (Collins, 1975). The characteristic set method, also known as Wu's method (Wu, 1978; Chou, 1985), the elimination method (Wang, 1995), the Gröbner basis method (Kapur, 1986b; Kapur, 1986a), and the Clifford algebra approach (Li, 2000) are examples of practical methods based on the algebraic approach. All these methods have in common an algebraic style, unrelated to traditional, synthetic geometry methods, and they do not provide human-readable proofs. Namely, they deal with polynomials that are often extremely complex for a human to understand, and also with no direct link to the geometrical contents.

The second approach to the automated theorem proving in geometry focuses on synthetic proofs, with an attempt to automate the traditional proving methods. Many of these methods add auxiliary elements to the geometric configuration considered, so that a certain postulates could apply. This usually leads to a combinatorial explosion of the search space. The challenge is to control the combinatorial explosion and to develop suitable heuristics in order to avoid unnecessary construction steps. Examples of synthetic proof methods include approaches by Gelertner (Gelernter, 1959), Nevis (Nevis, 1975), Elcock (Elcock, 1977), Greeno et al. (Greeno et al., 1979), Coelho and Pereira (Coelho and Pereira, 1986), Chou, Gao, and Zhang (Chou et al., 1993; Chou et al., 1996c).

In this paper we focus on the area method, an efficient semi-algebraic method for a fragment of Euclidean geometry, developed by Chou, Gao, and Zhang (Chou et al., 1993; Chou et al., 1994; Chou et al., 1996b). This method enables implementing efficient provers capable of generating human readable proofs. These proofs often differ from the traditional, Hilbert-style, synthetic proofs, but still they are often concise, consisting of steps that are directly related to the geometrical contents involved and hence can be easily understood by a mathematician.

The main idea of the area method is to express the hypotheses of a theorem using a set of starting ("free") points and a set of constructive statements each of them introducing a new point, and to express the conclusion by an equality between polynomials in some geometric quantities (without considering Cartesian coordinates). The proof is developed by eliminating, in reverse order, the points introduced before,

using for that purpose a set of appropriate lemmas. After eliminating all the introduced points, the conclusion of the theorem collapses to an equation between two rational expressions involving only free points. This equation can be further simplified to involve only independent variables. If the expressions on the two sides are equal, the statement is valid, otherwise it is invalid. All proof steps generated by the area method are expressed in terms of applications of high-level geometry lemmas and expression simplifications.

Although the basic idea of the method is simple, implementing it is a very challenging task because of a number of details that has to be dealt with. To our knowledge, apart from the original implementation by the authors who first proposed the area method, there are only three implementations more. These three implementations were made independently and in different contexts:

- within a tool for storing and exploring mathematical knowledge (Theorema (Buchberger et al., 2006)) — implemented by Judite Robu (Robu, 2002).
- within a generic proof assistant (Coq (The Coq development team, 2009)) — implemented by Julien Narboux (Narboux, 2004);
- within a dynamic geometry tool (GCLC (Janičić, 2006)) — implemented by Predrag Janičić and Pedro Quaresma (Janičić and Quaresma, 2006);

The implementations of the method can efficiently find proofs of a range of non-trivial theorems, including theorems due to Ceva, Menelaus, Gauss, Pappus, and Thales.

In this paper, we present an in-depth description of the area method covering all relevant definitions and lemmas. We also provide some of the implementations details, which are not given or not clearly stated in the original presentations. We follow the original exposition, but in a reorganised, more methodological form. This description of the area method should be sufficient for a complete understanding of the method, and for making a new implementation a straightforward task. This paper also summarises our results, experiences, and descriptions of our software systems related to the area method (Janičić and Quaresma, 2006; Quaresma and Janičić, 2006a; Quaresma and Janičić, 2006b; Janičić and Quaresma, 2007; Narboux, 2004; Narboux, 2007a).

In this paper we consider only the basic variant of the area method for Euclidean geometry, although there are other variants. Additional techniques can also be used to produce shorter proofs and slightly extend the basic domain of the method (Chou et al., 1994). However,

these techniques are applicable only in special cases and not in a uniform way, in contrast to the basic method. It is also possible to extend the area method to deal with inequations in the goal. Then the final inequation can be decided using an CAD algorithm or a heuristic like the sum of squares method. There are also variants of the area method developed for solid Euclidean geometry (Chou et al., 1995) and for hyperbolic plane geometry (Yang et al., 1998). Substantially, the idea of these variants is the same as in the basic method and this demonstrate that the approach has a wide domain. Variants of the method can be implemented in the same way described in this paper.

Overview of the paper. The paper is organised as follows: first, in Section 2, we explain the area method in details. In Section 3, we describe all the existing implementations of the method and some of their applications. In Section 4 we summarise our contributions and we draw final conclusions in Section 5.

2. The Area Method

The area method is a decision procedure for a fragment of Euclidean plane geometry. The method deals with problems stated in terms of sequences of specific geometric construction steps. We begin introducing the method by way of example.

In the rest of the paper, capital letters will denote points in the plane and $\triangle ABC$ will denote the triangle with vertices A, B, and C.

2.1. Introductory Example

The following simple example briefly illustrates some key features of the area method.

EXAMPLE 2.1. (Ceva's Theorem). *Let $\triangle ABC$ be a triangle and P be an arbitrary point in the plane. Let D be the intersection of AP and BC, E be the intersection of BP and AC, and F the intersection of CP and AB. Then it holds that:*

$$\frac{\overline{AF}}{\overline{FB}}\,\frac{\overline{BD}}{\overline{DC}}\,\frac{\overline{CE}}{\overline{EA}} = 1$$

This result can be stated and proved, within the area method setting.

The Construction. The points A, B, C, and P are *free points*, points not defined by construction steps. The point D is the intersection of the line determined by the points A and P and the line determined by

the points B and C. The points E and F are constructed in a similar fashion.

For this problem, an initial *non-degeneracy condition* is, that it holds $F \neq B$, $D \neq C$, and $E \neq A$. Notice also that the point P is not completely arbitrary point in the plane, since it should not belong to the three lines parallel to the sides of the triangle and passing through the opposite vertices (Figure 1).

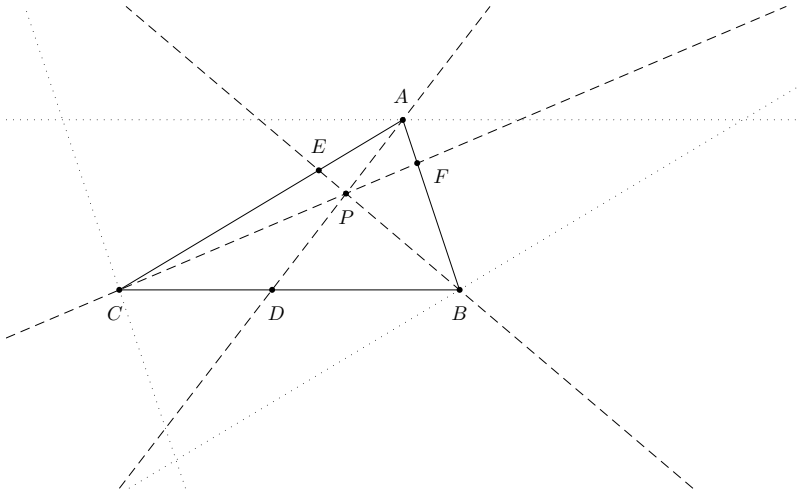


Figure 1. Illustration for Ceva's theorem

Stating the Conjecture. One of the key problems in automated theorem proving in geometry is the control of the combinatorial explosion that arises from the number of similar, but still different, cases that have to be analysed. For instance, given three points A, B, and C, how many triangles they define? One can argue that the answer is one, but from a syntactic point of view the $\triangle ABC$ is not equal to the $\triangle ACB$. For reducing such combinatorial explosion, but also for ensuring rigorous reasoning, one has to deal with arrangement relations, such as *on the same side of a line*, *two triangles have the same positive orientation*, etc. Note that, in Euclidean geometry, positive and negative orientation are just two names used to distinguish between the two orientations and one can select any triangle in the plane and proclaim that it has the orientation that will be called *positive* (and it is similar with orientation of segments on a line). In other words, in Euclidean geometry the notion of orientation is relative rather then absolute, and one can prove that a triangle has positive orientation, only if positive (and negative) orientation was already defined via some triangle in the same plane. In the Cartesian model of Euclidean geometry, the two orientations

are distinguished as *clockwise* and *counterclockwise* orientations. These two names should not be used for Euclidean geometry, because they cannot be defined there. Unfortunately, these terms are widely used in geometrical texts, including the description of the area method (Zhang et al., 1995).

For stating and proving conjectures, the area method uses a set of specific *geometric quantities*. The geometric quantities enable treating arrangement relations.

Within the area method the following geometric quantities are used:

- *ratio of parallel directed segments*, denoted $\overline{AB}/\overline{CD}$. If the points A, B, C, and D are collinear, $\overline{AB}/\overline{CD}$ is the ratio between lengths of directed segments AB and CD. If the points A, B, C, and D are not collinear, and it holds $AB \| CD$, there is a parallelogram $ABPQ$ such that P, Q, C, and D are collinear and then $\frac{\overline{AB}}{\overline{CD}} = \frac{\overline{QP}}{\overline{CD}}$.
- *signed area* for a triangle ABC, denoted $\mathcal{S}_{ABC}$;
- *Pythagoras difference*, denoted $\mathcal{P}_{ABC}$, for the points A, B, C, defined as $\mathcal{P}_{ABC} = \overline{AB}^2 + \overline{CB}^2 - \overline{AC}^2$.

These three geometric quantities allow expressing (in form of equalities) geometry properties such as collinearity of three points, parallelism of two lines, equality of two points, perpendicularity of two lines, etc. (see section 2.2.1). In the example, the conjecture is expressed using ratios of parallel directed segments.

Proof. The proof of a conjecture is based on eliminating all the constructed points, in reverse order, using for that propose the properties of the geometric quantities, until an equality in only the free points is reached. If the equality is valid, then the original conjecture is valid too. For the given example, a proof can be as follows:

It can be proved that $\frac{\overline{AF}}{\overline{FB}} = \frac{\mathcal{S}_{APC}}{\mathcal{S}_{BCP}}$. By analogy $\frac{\overline{BD}}{\overline{DC}} = \frac{\mathcal{S}_{BPA}}{\mathcal{S}_{CAP}}$ and $\frac{\overline{CE}}{\overline{EA}} = \frac{\mathcal{S}_{CPB}}{\mathcal{S}_{ABP}}$. Therefore:

$$\begin{aligned}
\frac{\overline{AF}}{\overline{FB}}\ \frac{\overline{BD}}{\overline{DC}}\ \frac{\overline{CE}}{\overline{EA}} &= \frac{\mathcal{S}_{APC}}{\mathcal{S}_{BCP}}\ \frac{\overline{BD}}{\overline{DC}}\ \frac{\overline{CE}}{\overline{EA}} && \text{the point } F \text{ is eliminated} \\
&= \frac{\mathcal{S}_{APC}}{\mathcal{S}_{BCP}}\ \frac{\mathcal{S}_{BPA}}{\mathcal{S}_{CAP}}\ \frac{\overline{CE}}{\overline{EA}} && \text{the point } D \text{ is eliminated} \\
&= \frac{\mathcal{S}_{APC}}{\mathcal{S}_{BCP}}\ \frac{\mathcal{S}_{BPA}}{\mathcal{S}_{CAP}}\ \frac{\mathcal{S}_{CPB}}{\mathcal{S}_{ABP}} && \text{the point } E \text{ is eliminated} \\
&= 1
\end{aligned}$$

Q.E.D.

The example illustrates how to express a problem using the given geometric quantities and how to prove it, and moreover, how to give a proof that is concise and very easy to understand.

The complete proof procedure will be given in Section 2.5. Before that, the underlying axiom system will be introduced.

2.2. Axiomatic Grounds for the Area Method

There is a number of axiom systems for Euclidean geometry. Euclid's system (Heath, 1956), partly naive from today's point of view, was used for centuries. In early twenty century, Hilbert provided a more rigorous axiomatisation (Hilbert, 1977), one of the landmarks for modern mathematics, but still not up to modern standards (Dehlinger et al., 2000; Meikle and Fleuriot, 2003). In mid-twenty century, Tarski presented a new axiomatisation for elementary geometry (without all continuity features ensured), along with a decision procedure for that theory (Tarski, 1959). Although there are other variations of these systems (Janičić, 1996; Narboux, 2006), these three are the most influential and most popular axiomatic systems for geometry.

Modern courses on classical Euclidean geometry are most often based on Hilbert's axioms. In Hilbert-style geometry, the primitive (not defined) objects are: *point*, *line*, *plane*. The primitive (not defined) predicates are those of congruence and order (with addition of equality and incidence[1]). Properties of the primitive objects and predicates are introduced by five groups of axioms, such as: "For two points A, B there exists a line a such that both A and B are incident with it".

In the following text we briefly discuss how axiomatic grounds can be built for the fragment of geometry treated by the area method. We will present two approaches, both enabling proving properties of geometric quantities required by the area method.

2.2.1. *A Hilbert Style Axiomatisation*

The geometric quantities used within the area method can be defined in Hilbert style geometry, but they also require axioms of the theory of fields.

The notions of the ratio of parallel directed segments and of the signed area involve the notion of orientation of segments on a line and the notion of orientation of triangles in a plane (discussed in section 2.1).

DEFINITION 1. (Ratio of parallel directed segments).

[1] See von Plato's discussion about incidence in Hilbert's geometry (von Plato, 1997).

Table I. Expressing geometry predicates in terms of the three geometric quantities.

property	**in terms of geometric quantities**
points A and B are identical	$\mathcal{P}_{ABA} = 0$
points A, B, C are collinear	$\mathcal{S}_{ABC} = 0$
AB is perpendicular to CD	$\mathcal{P}_{ACD} = \mathcal{P}_{BCD}$
AB is parallel to CD	$\mathcal{S}_{ACD} = \mathcal{S}_{BCD}$
O is the midpoint of AB	$\frac{\overline{AO}}{\overline{OB}} = 1$
AB has the same length as CD	$\mathcal{P}_{ABA} = \mathcal{P}_{CDC}$
points A, B, C, D are harmonic	$\frac{\overline{AC}}{\overline{CB}} = \frac{\overline{DA}}{\overline{DB}}$
angle ABC has the same measure as DEF	$\mathcal{S}_{ABC} \cdot \mathcal{P}_{DEF} = \mathcal{S}_{DEF} \cdot \mathcal{P}_{ABC}$
A, B, C and D belong to the same circle	$\mathcal{S}_{CAD} \cdot \mathcal{P}_{CBD} = \mathcal{S}_{CBD} \cdot \mathcal{P}_{CAD}$

If the points A, B, C, and D are collinear, $\frac{\overline{AB}}{\overline{CD}}$ is the ratio between lengths of directed segments AB and CD. If the points A, B, C, and D are not collinear, and it holds $AB \| CD$, there is a parallelogram $ABPQ$ such that P, Q, C, and D are collinear and then $\frac{\overline{AB}}{\overline{CD}} = \frac{\overline{QP}}{\overline{CD}}$.

DEFINITION 2. (Signed Area). *The* signed area *of the triangle ABC, denoted $\mathcal{S}_{ABC}$.*

The *Pythagoras difference* is a generalisation of the Pythagoras equality regarding the three sides of a right triangle, to an expression applicable to any triangle (for a triangle ABC with the right angle at B, it holds that $\mathcal{P}_{ABC} = 0$.).

DEFINITION 3. (Pythagoras difference). *For three points A, B, and C, the* Pythagoras difference*, denoted $\mathcal{P}_{ABC}$, is defined in the following way:*

$$\mathcal{P}_{ABC} = \overline{AB}^2 + \overline{CB}^2 - \overline{AC}^2.$$

Using these three geometric quantities it is possible to express a range of geometry predicates as shown in Table 2.2.1.

Proofs generated by the area method use a set of specific lemmas. These lemmas can be proved within the Hilbert's geometry (i.e., within its fragment for plane geometry), but the full, formal proofs would be very long. That is why it is suitable to have an alternative axiomatisation, suitable for the area method. Chou, Gao and Zhang (Chou et al., 1993) proposed such a system for affine geometry, and in the next section we propose a variant of this system.

2.2.2. *A New Axiom System for the Area Method*
The axiom system used by Chou, Gao and Zhang (Chou et al., 1994) is a Hilbert style axiom system, i.e. a semi-analytic axiom system with (only) points as primitive objects (lines are not primitive objects as in Hilbert's axiom system). The axiom system contains the axioms of field, so the system uses the concept of numbers, but it is still coordinate free. The field is not assumed to be ordered, hence the axiom system has the property of representing an unordered geometry. This means that, for instance, one cannot express the concept of a point being between two points (unlike in Hilbert's system).

In the following, we present our special-purpose axiom system for Euclidean plane geometry (within first order logic with equality), a modified version of the axiomatic system of Chou, Gao and Zhang. There are several reasons why we modified the original axiom system. Compared to the original version, ours has the advantage to be more concise and organised. Moreover, we *formally verified* (within the *Coq* proof assistant (The Coq development team, 2009)) all the properties of the geometric quantities required by the area method, demonstrating the correctness of the system and eliminating all concerns about validity of the lemmas.

In our axiom system, there is just one type of objects: points. The system uses a field $(F, +, \cdot, 0, 1)$ of characteristic different from 2.[2] The axioms of the theory of fields are standard and hence omitted.

There is one primitive binary function symbol $(\overline{\cdot\cdot})$ and one ternary function symbols $(\mathcal{S}_{\cdot\cdot\cdot})$ from points to F. The first depicts the signed distance between two points, the second represents the signed area of a triangle. All axioms given in Table II are implicitly universally quantified. To improve readability (of the last three axioms), the following shortcuts are used:

$$\begin{aligned} \mathcal{P}_{ABC} &\equiv \overline{AB}^2 + \overline{BC}^2 - \overline{AC}^2 \\ AB \parallel CD &\equiv \mathcal{S}_{ACB} + \mathcal{S}_{ABD} = 0 \\ AB \perp CD &\equiv \mathcal{P}_{ACD} + \mathcal{P}_{BCD} = 0 \end{aligned}$$

The axiom system we propose differs from the axiom system of Chou, Gao and Zhang in several aspects.

First, our system does not use collinearity as a primitive notion and instead, collinearity is defined by the signed area. Chou, Gao and

[2] The fact that the characteristic of F is different from 2 is used to simplify the axiom system. Indeed, if $0 \neq 2$ since $\forall ABC, \mathcal{S}_{ABC} = -\mathcal{S}_{BAC}$ (by axiom 3) then $\forall AC, \mathcal{S}_{AAC} = -\mathcal{S}_{AAC}$ and hence $\forall AC, \mathcal{S}_{AAC} = 0$, so we can omit the axiom $\mathcal{S}_{AAC} = 0$ which appears in the system proposed by Chou et al. In addition, this assumption allows, for instance, construction of the midpoint (using the construction axiom with $r = \frac{1}{2}$) of a segment without explicitly stating the assumption $0 \neq 2$.

Table II. The axiom system

1. $\overline{AB} = 0$ if and only if the points A and B are identical
2. $\mathcal{S}_{ABC} = \mathcal{S}_{CAB}$
3. $\mathcal{S}_{ABC} = -\mathcal{S}_{BAC}$
4. If $\mathcal{S}_{ABC} = 0$ then $\overline{AB} + \overline{BC} = \overline{AC}$ (Chasles's axiom)
5. There are points A, B, C such that $\mathcal{S}_{ABC} \neq 0$ (dimension; not all points are collinear)
6. $\mathcal{S}_{ABC} = \mathcal{S}_{DBC} + \mathcal{S}_{ADC} + \mathcal{S}_{ABD}$ (dimension; all points are in the same plane)
7. For each element r of F, there exists a point P, such that $\mathcal{S}_{ABP} = 0$ and $\overline{AP} = r\overline{AB}$ (construction of a point on the line)
8. If $A \neq B, \mathcal{S}_{ABP} = 0, \overline{AP} = r\overline{AB}, \mathcal{S}_{ABP'} = 0, \overline{AP'} = r\overline{AB}$, then $P = P'$ (unicity)
9. If $\mathcal{S}_{PCQ} + \mathcal{S}_{PQD} = 0$, $C \neq D$ $\mathcal{S}_{CDQ} \neq 0$, and $\frac{\overline{PQ}}{\overline{CD}} = 1$, then $\frac{\mathcal{S}_{PDQ}}{\mathcal{S}_{CDQ}} = 1$ (parallelogram)
10. If $\mathcal{S}_{PAC} \neq 0$ and $\mathcal{S}_{ABC} = 0$ then $\frac{\overline{AB}}{\overline{AC}} = \frac{\mathcal{S}_{PAB}}{\mathcal{S}_{PAC}}$ (proportions)
11. If $C \neq D$ and $AB \perp CD$ and $EF \perp CD$ then $AB \parallel EF$.
12. If $A \neq B$ and $AB \perp CD$ and $AB \parallel EF$ then $EF \perp CD$.
13. If $FA \perp BC$ and $\mathcal{S}_{FBC} = 0$ then $4\mathcal{S}^2_{ABC} = \overline{AF}^2\overline{BC}^2$ (area of a triangle).

Zhang's system has axioms introducing properties of collinearity, and these axioms are then used for proving that three points are collinear if and only if $\mathcal{S}_{ABC} = 0$ (Chou et al., 1994).

Second, while Chou, Gao and Zhang's axiom system restricts to ratios of directed parallel segments $\frac{\overline{AB}}{\overline{CD}}$ where the lines AB and CD are parallel, we skip this syntactical restriction and can use ratios for arbitrary points. The consistency of the axiom system is preserved because the concept of oriented distance can be interpreted in the standard Cartesian model. The area method requires explicitly that for every ratio of directed segments $\frac{\overline{AB}}{\overline{CD}}$, AB is parallel to CD. Therefore, the area method is not a decision procedure for this theory, as it can not prove or disprove all conjectures stated in the introduced language because the method can not deal with ratios of the form $\frac{\overline{AB}}{\overline{CD}}$ if $AB \nparallel CD$ (however, it is a decision procedure for the set of formulae from the restricted version of the language).

Third, while Chou, Gao and Zhang's axiom system deals with affine geometry, we extend the system to deal with Euclidean geometry as we make explicit the axioms about Pythagoras difference (axioms 11, 12, and 13).

2.3. Geometric Constructions

The area method is used for proving constructive geometry conjectures: statements about properties of objects constructed by some fixed set of elementary constructions. In this section we first describe the set of available construction steps and then the set of conjectures that can be expressed.

2.3.1. *Elementary Construction Steps*

Constructions covered by the area method are closely related, but still different, from constructions by ruler and compass. These are the elementary constructions by ruler and compass:

- construction of an arbitrary point;
- construction of an arbitrary line;
- construction (by ruler) of a line such that two given points belong to it;
- construction (by compass) of a circle such that its centre is one given point and such that the second given point belongs to it;
- construction of a point such that it is the intersection of two lines (if such a point exists);
- construction of the intersections of a given line and a given circle (if such points exist).
- construction of the intersections of two given circles (if such points exist).

The area method cannot deal with all geometry theorems involving the above constructions. It does not support construction of an arbitrary line, and support intersections of two circles and intersections of a line and a circle only in a limited way.

Instead of support for intersections of two circles or a line and a circle (critical for describing many geometry theorems), there are new, specific construction steps. All constructions supported by the area method are expressed in terms of the involved points.[3] Therefore, only

[3] Elementary construction steps used by the area method do not use the concept of line and plane explicitly. This is convenient from the formalisation and automatization point of view. Indeed, in an axiom system based only on the concept of points (as in Tarski's axiom system (Tarski, 1959)), the dimension implied can be easily changed by adding or removing some appropriate axioms (stated in the original

lines and circles determined by specific points can be used (rather than arbitrarily chosen lines and circles) and the key constructions steps are those introducing new points. For a construction steps to be well-defined, certain conditions may be required. These conditions are called *non-degeneracy condition* (ndg-conditions). The *degree of freedom* tells if a point is free (degree bigger than 0), or not.

In the following text, (LINE U V) will denote a line such that the points U and V belong to it, and (CIRCLE O U) will denote a circle such that its centre is point O and such that the point U belongs to it.

Some of the constructions steps are formulated using the fixed field $(F, +, \cdot, 0, 1)$, employed by the used axiom system.

Given below is the list of elementary constructions in the area method, along with the corresponding ndg-conditions and the degrees of freedom of the constructed points.

ECS1 construction of an arbitrary point U; this construction step is denoted by (POINT U).

ndg-condition: –

degree of freedom for U: 2

ECS2 construction of a point Y such that it is the intersection of two lines (LINE U V) and (LINE P Q); this construction step is denoted by (INTER Y (LINE U V) (LINE P Q))

ndg-condition: $UV \nparallel PQ$; $U \neq V$; $P \neq Q$.

degree of freedom for Y: 0

ECS3 construction of a point Y such that it is the foot from a given point P to (LINE U V); this construction step is denoted by (FOOT Y P (LINE U V)).

ndg-condition: $U \neq V$

degree of freedom for Y: 0

ECS4 construction of a point Y on the line passing through a point W and is parallel to (LINE U V), such that $\overline{WY} = r\overline{UV}$, where r is an element of F, a rational expression in geometric quantities, or a variable; this construction step is denoted by (PRATIO Y W (LINE U V) r).

signature). On the other hand, in an axiom system based on the concepts of points and lines, such as Hilbert's axiom system, in order to extent the system to the third dimension ones needs both to update some axioms, to introduce some new axioms and to *change the signature of the theory* by introducing the sort of planes.

ndg-condition: $U \neq V$; if r is a rational expression in the geometric quantities, the denominator of r should not be zero.

degree of freedom for Y: 0, if r is a fixed quantity; 1, if r is a variable.

ECS5 construction of a point Y on the line passing through a point U and perpendicular to (LINE U V), such that $\frac{4\mathcal{S}_{UVY}}{\mathcal{P}_{UVU}} = r$, where r is a rational number, a rational expression in geometric quantities, or a variable; this construction step is denoted by (TRATIO Y (LINE U V) r).

ndg-condition: $U \neq V$; if r is a rational expression in geometric quantities then the denominator of r should not be zero.

degree of freedom for Y: 0, if r is a fixed quantity; 1, if r is a variable.

The above set of constructions is sufficient for expressing many constructions based on ruler and compass, but not all of them. For instance, an arbitrary line cannot be constructed by the above construction steps. Still, one can construct two arbitrary points and then (implicitly) the line going through these points.

Also, intersections of two circles and intersections of a line and a circle are not supported in a general case. However, it is still possible to construct intersections of two circles and intersections of a line and a circle in some special cases. For example:

- construction of a point Y such that it is the intersection (other than point U) of a line (LINE U V) and a circle (CIRCLE O U) can be represented as a sequence of two construction steps: (FOOT N O (LINE U V)), (PRATIO Y N (LINE N U) -1).

- construction of a point Y such that it is the intersection (other than point P) of a circle (CIRCLE O1 P) and a circle (CIRCLE O2 P) can be represented as a sequence of two construction steps: (FOOT N P (LINE O1 O2)), (PRATIO Y N (LINE N P) -1).

In addition, many other constructions (expressed in terms of constructions by ruler and compass) can be performed by the elementary constructions of the area method. Some of them are:

- construction of a line such that a given point W belongs to it and it is parallel to a line (LINE U V); it can be represented as a sequence of two steps: (PRATIO N W (LINE U V) 1), (LINE W N).

- construction of a line such that a given point W belongs to it and it is perpendicular to a line (LINE U V); if W, U, V are collinear, then this construction can be represented as (TRATIO N (LINE W U) 1), (LINE N W), otherwise it can be represented as (FOOT N W (LINE U V)), (LINE N W).
- construction of a perpendicular bisector of a segment with endpoints U and V; this construction can be represented as (PRATIO M (LINE U U) V 1/2), (TRATIO N (LINE M U) 1), (LINE N M).

Also, it is possible to construct an arbitrary point on a line (LINE U V), by (PRATIO Y U (LINE U V) r) where r is an indeterminate, or on a circle (CIRCLE O P), by (POINT Q), (FOOT N O (LINE P Q)), (PRATIO Y N (LINE N P) -1).

Within a wider system (e.g., within a dynamic geometry tool), a richer set of construction steps can be used for describing geometry conjectures as long as all of them can be represented by the elementary construction steps of the area method.

As said, the set of elementary construction steps in the area method cannot cover all constructions based on ruler and compass. On the other end, there are also some constructions that can be performed by the above construction steps and that cannot be performed by ruler and compass. For instance, if $\sqrt[3]{2} \in F$ then, given two distinct points A and B, one can construct a third point C such that $\overline{AC} = \sqrt[3]{2}\,\overline{AB}$, since one can use this number (whereas it is not possible using ruler and compass).

EXAMPLE 2.2. *The construction given in Example 2.1 can be represented in terms of the given construction steps as follows:*

A, B, C, P *are free points*	*(ECS1)*
INTER D (LINE A P) (LINE B C))	*(ECS2)*
INTER E (LINE B P) (LINE A C))	*(ECS2)*
INTER F (LINE C P) (LINE A B))	*(ECS2)*

2.3.2. *Constructive Geometry Statements*

In the area method, geometry statements have a specific form.

DEFINITION 4. (Constructive Geometry Statement). *A* constructive geometry statement*, is a list* $S = (C_1, C_2, \ldots, C_n, G)$ *where* C_i*, for* $1 \leq i \leq n$*, are elementary construction steps, and the conclusion of the statement,* G *is of the form* $E_1 = E_2$*, where* E_1 *and* E_2 *are polynomials in geometric quantities of the points introduced by the steps* C_i*. In each of* C_i*, the points used in the construction steps must be already introduced by the preceding construction steps.*

The class of all constructive geometry statements is denoted by $\mathbf{C}$.

Note that, in its basic form, the area method does not deal with inequalities in its conclusion statement, G (for another variant of the method see section 3.3.2).

For a statement $S = (C_1, C_2, \ldots, C_n, (E_1 = E_2))$ from $\mathbf{C}$, the ndg-condition is the set of ndg-conditions of the steps C_i plus the condition that the denominators of the length ratios in E_1 and E_2 are not equal to zero, and the conditions that line appearing in the length ratios in E_1 and E_2 are parallel. The logical meaning of a statement is hence:

$$C_1 \wedge C_2 \wedge ... \wedge C_n \wedge$$

$$NDG_1 \wedge NDG_2 \wedge ... \wedge NDG_n \wedge$$

$$d_1 \wedge ... \wedge d_m$$

$$p_1 \wedge ... \wedge p_m$$

$$\Rightarrow E_1 = E_2$$

where C_i are the propositions characterising the construction steps; NDG_i are the ndg-conditions associated to the construction steps; d_i are the conditions on denominators appearing in E_1 and E_2; and p_i are the conditions about parallelism: for each ratio of the form $\frac{\overline{AB}}{\overline{CD}}$ appearing in E_1 and E_2, there is the condition $AB \parallel CD$.

EXAMPLE 2.3. *The statement corresponding to the theorem given in Example 2.1 can be represented as follows:*

$AP \nparallel BC \wedge A \neq P \wedge B \neq C \wedge$
$BP \nparallel AC \wedge B \neq P \wedge A \neq C \wedge$
$CP \nparallel AB \wedge C \neq P \wedge A \neq B \wedge$
$F \neq B \wedge D \neq C \wedge E \neq A \wedge$
$AF \parallel FB \wedge BD \parallel DC \wedge CE \parallel EA \wedge$
$\Rightarrow \frac{\overline{AF}}{\overline{FB}} \frac{\overline{BD}}{\overline{DC}} \frac{\overline{CE}}{\overline{EA}} = 1$

2.4. Properties of Geometric Quantities & Elimination Lemmas

We present some definitions and the properties of geometric quantities, required by the area method. We follow the material from original descriptions of the method (Chou et al., 1993; Chou et al., 1994; Chou et al., 1996b; Zhang et al., 1995), but in a reorganised form. The rigorous traditional proofs (not formal) accompanying all the results presented in this section are available in (Quaresma and Janičić, 2009). The formal

(machine verifiable) proofs are available as a *Coq* contribution (Narboux, 2009).

Along the method application, in addition to the basic geometric quantities, some additional quantities ($\mathcal{S}_{ABCD}$ and $\mathcal{P}_{ABCD}$) may occur in the conjecture being proved. These quantities are defined in terms of the basic quantities, as follows.

DEFINITION 5. *The signed area of a quadrilateral $ABCD$ is defined as $\mathcal{S}_{ABCD} = \mathcal{S}_{ABC} + \mathcal{S}_{ACD}$.*

DEFINITION 6. *For four points A,B,C and D, $\mathcal{P}_{ABCD}$ is defined as follows:*

$$\mathcal{P}_{ABCD} = \mathcal{P}_{ABD} - \mathcal{P}_{CBD} = \overline{AB}^2 + \overline{CD}^2 - \overline{BC}^2 - \overline{DA}^2.$$

The following lemmas are implicitly universally quantified and it is assumed that it holds $A \neq B$ for any ratio of parallel directed segments of the form $\frac{\overline{XY}}{\overline{AB}}$.

LEMMA 1. $\frac{\overline{PQ}}{\overline{AB}} = -\frac{\overline{QP}}{\overline{AB}} = \frac{\overline{QP}}{\overline{BA}} = -\frac{\overline{PQ}}{\overline{BA}}$.

LEMMA 2. $\frac{\overline{PQ}}{\overline{AB}} = 0$ *iff* $P = Q$.

LEMMA 3. $\frac{\overline{PQ}}{\overline{AB}}\ \frac{\overline{AB}}{\overline{PQ}} = 1$.

LEMMA 4. $\mathcal{S}_{ABC} = \mathcal{S}_{CAB} = \mathcal{S}_{BCA} = -\mathcal{S}_{ACB} = -\mathcal{S}_{BAC} = -\mathcal{S}_{CBA}$.

LEMMA 5. $\mathcal{P}_{AAB} = 0$.

LEMMA 6. $\mathcal{P}_{ABC} = \mathcal{P}_{CBA}$.

LEMMA 7. $\mathcal{P}_{ABA} = 2\overline{AB}^2$.

2.4.1. *Elimination Lemmas*
An elimination lemma is a theorem that has the following properties:

- it states an equality between a geometric quantity involving a certain constructed point Y and an expression not involving Y;
- this last expression is composed using only geometric quantities;
- this expression is well defined: denominators are different from zero and ratios of distances are composed only using parallel segments.

It is required to describe elimination of points introduced by four construction steps (ECS2 to ECS5) from three kinds of geometric quantities.

Some elimination lemmas enable eliminating a point from expressions only at certain positions — usually the last position in the list of the arguments. That is why it is necessary first to transform relevant terms of the current goal into the form that can be dealt with by these elimination lemmas. Moreover, some elimination lemmas require that some points are assumed to be distinct. The first following lemma ensures that this assumptions can be met.

LEMMA 8. *If G is a geometric quantity involving Y, then either G is equal to zero or it can be transformed into one of the following forms (or their sum or difference), for some A, B, C, and D that are different from Y:*

$$\frac{\overline{AY}}{\overline{CD}}; \frac{\overline{AY}}{\overline{BY}}; -\frac{\overline{AY}}{\overline{BY}}; \frac{1}{\frac{\overline{AY}}{\overline{CD}}}; \mathcal{P}_{ABY}; \mathcal{P}_{AYB}; \mathcal{S}_{ABY};$$

Proof: If G is a geometric quantity of arity 4 ($\mathcal{S}_{ABCD}$ or $\mathcal{P}_{ABCD}$), the first step is to transform it into terms of arity 3 by one of the following two rules.

$$\begin{array}{lll} \mathcal{S}_{ABCD} & \rightarrow & \mathcal{S}_{ABC} + \mathcal{S}_{ACD} \quad \text{Definition 5} \\ \mathcal{P}_{ABCD} & \rightarrow & \mathcal{P}_{ABD} - \mathcal{P}_{CBD} \quad \text{Definition 6} \end{array}$$

Now, all remaining geometric quantities (involving Y) can be treated.

Signed ratios: G can have one of the following forms (for some A, B, and C different from Y):

- $\frac{\overline{YY}}{\overline{AY}} = 0$ (by Lemma 2)
- $\frac{\overline{YY}}{\overline{YA}} = 0$ (by Lemma 2)
- $\frac{\overline{YY}}{\overline{CD}} = 0$ (by Lemma 2)
- $\frac{\overline{AY}}{\overline{BY}}$
- $\frac{\overline{AY}}{\overline{YB}} = -\frac{\overline{AY}}{\overline{BY}}$ (by Lemma 1)
- $\frac{\overline{YA}}{\overline{BY}} = -\frac{\overline{AY}}{\overline{BY}}$ (by Lemma 1)
- $\frac{\overline{YA}}{\overline{YB}} = \frac{\overline{AY}}{\overline{BY}}$ (by Lemma 1)
- $\frac{\overline{AY}}{\overline{CD}}$
- $\frac{\overline{YA}}{\overline{CD}} = -\frac{\overline{AY}}{\overline{CD}}$ (by Lemma 1)

- $\frac{\overline{AB}}{\overline{CY}} = \frac{1}{\frac{\overline{CY}}{\overline{AB}}}$ (by lemmas 1 and 3)
- $\frac{\overline{AB}}{\overline{YC}} = \frac{1}{\frac{\overline{CY}}{\overline{BA}}}$ (by lemmas 1 and 3)

Signed area: G can have one of the following forms (for some A and B different from Y):

- $\mathcal{S}_{YYY} = 0$ (by Lemma 4)
- $\mathcal{S}_{AYY} = 0$ (by Lemma 4)
- $\mathcal{S}_{YAY} = 0$ (by Lemma 4)
- $\mathcal{S}_{YYA} = 0$ (by Lemma 4)
- $\mathcal{S}_{AYB} = \mathcal{S}_{BAY}$ (by Lemma 4)
- $\mathcal{S}_{YAB} = \mathcal{S}_{ABY}$ (by Lemma 4)
- $\mathcal{S}_{ABY}$

Pythagoras difference: G can have one of the following forms (for some A and B different from Y):

- $\mathcal{P}_{YYY} = 0$ (by Lemma 5)
- $\mathcal{P}_{AYY} = 0$ (by lemmas 6 and 5)
- $\mathcal{P}_{YAY} = \mathcal{P}_{AYA}$ (by Lemma 7)
- $\mathcal{P}_{YYA} = 0$ (by Lemma 5)
- $\mathcal{P}_{AYB}$
- $\mathcal{P}_{YAB} = \mathcal{P}_{BAY}$ (by Lemma 6)
- $\mathcal{P}_{ABY}$

Q.E.D.

If $G(Y)$ is one of the following geometric quantities: $\mathcal{S}_{ABY}$, $\mathcal{S}_{ABCY}$, $\mathcal{P}_{ABY}$, or $\mathcal{P}_{ABCY}$ for points A, B, C different from Y, then $G(Y)$ is called a *linear geometric quantity.*

The following lemmas are used for the elimination of Y from geometric quantities. Thanks to Lemma 8, it is sufficient to consider only geometric quantities with only one occurrence of Y and the case $\frac{\overline{AY}}{\overline{BY}}$. Therefore, it can be assumed that Y differs from A, B, C, and D in the following lemmas (although they are valid in a general case, unless stated otherwise). This ensures that Y does not occur on the right hand sides appearing in the elimination lemmas.

LEMMA 9. (**EL1**). *If Y is introduced by* (INTER Y (LINE U V) (LINE P Q)) *then it holds that:*[4]

$$\frac{\overline{AY}}{\overline{CY}} = \begin{cases} \frac{\mathcal{S}_{APQ}}{\mathcal{S}_{CPQ}} & \text{if } A \text{ is on } UV \\ \frac{\mathcal{S}_{AUV}}{\mathcal{S}_{CUV}} & \text{otherwise} \end{cases}$$

$$\frac{\overline{AY}}{\overline{CD}} = \begin{cases} \frac{\mathcal{S}_{APQ}}{\mathcal{S}_{CPDQ}} & \text{if } A \text{ is on } UV \\ \frac{\mathcal{S}_{AUV}}{\mathcal{S}_{CUDV}} & \text{otherwise} \end{cases}$$

LEMMA 10. (**EL2**). *If Y is introduced by* (FOOT Y P (LINE U V)) *then it holds that (we assume $D \neq U$; otherwise interchange U and V):*

$$\frac{\overline{AY}}{\overline{CY}} = \begin{cases} \frac{\mathcal{P}_{PUV}\mathcal{P}_{PCAV}+\mathcal{P}_{PVU}\mathcal{P}_{PCAU}}{\mathcal{P}_{PUV}\mathcal{P}_{CVC}+\mathcal{P}_{PVU}\mathcal{P}_{CUC}-\mathcal{P}_{PUV}\mathcal{P}_{PVU}} & \text{if } A \text{ is on } UV \\ \frac{\mathcal{S}_{AUV}}{\mathcal{S}_{CUV}} & \text{otherwise} \end{cases}$$

$$\frac{\overline{AY}}{\overline{CD}} = \begin{cases} \frac{\mathcal{P}_{PCAD}}{\mathcal{P}_{CDC}} & \text{if } A \text{ is on } UV \\ \frac{\mathcal{S}_{AUV}}{\mathcal{S}_{CUDV}} & \text{otherwise} \end{cases}$$

LEMMA 11. (**EL3**). *If Y is introduced by* (PRATIO Y R (LINE P Q) r) *then it holds that (we assume that $A \neq Y$):*

$$\frac{\overline{AY}}{\overline{CY}} = \begin{cases} \frac{\frac{\overline{AR}}{\overline{PQ}}+r}{\frac{\overline{CR}}{\overline{PQ}}+r} & \text{if } A \text{ is on } RY \\ \frac{\mathcal{S}_{APRQ}}{\mathcal{S}_{CPRQ}} & \text{otherwise} \end{cases}$$

$$\frac{\overline{AY}}{\overline{CD}} = \begin{cases} \frac{\frac{\overline{AR}}{\overline{PQ}}+r}{\frac{\overline{CD}}{\overline{PQ}}} & \text{if } A \text{ is on } RY \\ \frac{\mathcal{S}_{APRQ}}{\mathcal{S}_{CPDQ}} & \text{otherwise} \end{cases}$$

LEMMA 12. (**EL4**). *If Y is introduced by* (TRATIO Y (LINE P Q) r) *then it holds that:*

$$\frac{\overline{AY}}{\overline{CY}} = \begin{cases} \frac{\mathcal{S}_{APQ}-\frac{r}{4}\mathcal{P}_{PQP}}{\mathcal{S}_{CPQ}-\frac{r}{4}\mathcal{P}_{PQP}} & \text{if } A \text{ is on } PY \\ \frac{\mathcal{P}_{APQ}}{\mathcal{P}_{CPQ}} & \text{otherwise} \end{cases}$$

$$\frac{\overline{AY}}{\overline{CD}} = \begin{cases} \frac{\mathcal{S}_{APQ}-\frac{r}{4}\mathcal{P}_{PQP}}{\mathcal{S}_{CPDQ}} & \text{if } A \text{ is on } PY \\ \frac{\mathcal{P}_{APQ}}{\mathcal{P}_{CPDQ}} & \text{otherwise} \end{cases}$$

[4] Notice that in this and other lemmas, the condition A on UV is trivially met if A is one of the points U and V. This special case may be treated as a separate case for the sake of efficiency.

LEMMA 13. (**EL5**). *Let $G(Y)$ be a linear geometric quantity and Y is introduced by* (INTER Y (LINE U V) (LINE P Q)). *Then it holds that:*

$$G(Y) = \frac{\mathcal{S}_{UPQ}G(V) - \mathcal{S}_{VPQ}G(U)}{\mathcal{S}_{UPVQ}}.$$

LEMMA 14. (**EL6**). *Let $G(Y)$ be a linear geometric quantity and Y is introduced by* (FOOT Y P (LINE U V)). *Then it holds that:*

$$G(Y) = \frac{\mathcal{P}_{PUV}G(V) + \mathcal{P}_{PVU}G(U)}{\mathcal{P}_{UVU}}.$$

LEMMA 15. (**EL7**). *Let $G(Y)$ be a linear geometric quantity and Y is introduced by* (PRATIO Y W (LINE U V) r). *Then it holds that:*

$$G(Y) = G(W) + r(G(V) - G(U)).$$

LEMMA 16. (**EL8**). *If Y is introduced by* (TRATIO Y (LINE P Q) r) *then it holds that:*

$$\mathcal{S}_{ABY} = \mathcal{S}_{ABP} - \frac{r}{4}\mathcal{P}_{PAQB}.$$

LEMMA 17. (**EL9**). *If Y is introduced by* (TRATIO Y (LINE P Q) r) *then it holds that:*

$$\mathcal{P}_{ABY} = \mathcal{P}_{ABP} - 4r\mathcal{S}_{PAQB}.$$

LEMMA 18. (**EL10**). *If Y is introduced by* (INTER Y (LINE U V) (LINE P Q)) *then it holds that:*

$$\mathcal{P}_{AYB} = \frac{\mathcal{S}_{UPQ}}{\mathcal{S}_{UPVQ}}G(V) + \frac{\mathcal{S}_{VPQ}}{\mathcal{S}_{UPVQ}}G(U) - \frac{\mathcal{S}_{UPQ}\cdot\mathcal{S}_{VPQ}\cdot\mathcal{P}_{UVU}}{\mathcal{S}^2_{UPVQ}}.$$

LEMMA 19. (**EL11**). *If Y is introduced by* (FOOT Y P (LINE U V)) *then it holds that:*

$$\mathcal{P}_{AYB} = \frac{\mathcal{P}_{PUV}}{\mathcal{P}_{UVU}}G(V) + \frac{\mathcal{P}_{PVU}}{\mathcal{P}_{UVU}}G(U) - \frac{\mathcal{P}_{PUV}\cdot\mathcal{P}_{PVU}}{\mathcal{P}_{UVU}}.$$

LEMMA 20. (**EL12**). *If Y is introduced by* (PRATIO Y W (LINE U V) r) *then it holds that:*

$$\mathcal{P}_{AYB} = \mathcal{P}_{AWB} + r(\mathcal{P}_{AVB} - \mathcal{P}_{AUB} + 2\cdot\mathcal{P}_{WUV}) - r(1-r)\mathcal{P}_{UVU}.$$

LEMMA 21. (**EL13**). *If Y is introduced by* (TRATIO Y (LINE P Q) r) *then it holds that:*

$$\mathcal{P}_{AYB} = \mathcal{P}_{APB} + r^2\mathcal{P}_{PQP} - 4r(\mathcal{S}_{APQ} + \mathcal{S}_{BPQ}).$$

Table III. Elimination Lemmas

		Geometric Quantities			
		$\frac{\overline{AY}}{\overline{CY}}$ $\frac{\overline{AY}}{\overline{CD}}$	$\mathcal{S}_{ABY}$ $\mathcal{S}_{ABCY}$	$\mathcal{P}_{ABY}$ $\mathcal{P}_{ABCY}$	$\mathcal{P}_{AYB}$
Constructive Steps	ECS2	EL1	EL5		EL10
	ECS3	EL2	EL6		EL11
	ECS4	EL3	EL7		EL12
	ECS5	EL4	EL8	EL9	EL13
		Elimination Lemmas			

The information on the elimination lemmas is summarized in Table III.

On the bases of the above lemmas, given a statement S, it is always possible to eliminate all constructed points (in reverse order) leaving only free points, numerical constants and numerical variables. Namely, by Lemma 8, all geometric quantities are transformed into one of the standard forms and then appropriate elimination lemmas (depending on the construction steps) are used to eliminate all constructed points.

2.5. The Algorithm and its Properties

In this section we present the area method's algorithm. As explained in section 2.1, the idea of the method is to eliminate all the constructed points and then to transform the statement being proved into an expression involving only independent geometric quantities.

2.5.1. *Dealing with Side Conditions in Elimination Lemmas*

Apart from ndg-conditions of the construction steps, there are also side conditions in some of the elimination lemmas. Namely, some elimination lemmas have two cases (side conditions) — positive (always of the form "A is on PQ") and negative (always of the form "A is not on PQ"). As in the case of ndg-conditions, the positive side conditions (those of the form "A is on PQ") can also be expressed in terms of geometric quantities (as $\mathcal{S}_{APQ} = 0$) and checked by the area method itself. Negative side conditions (expressed ad $\mathcal{S}_{APQ} \neq 0$) can also be proved in some situations. Basically, the area method can only prove conjectures of the form $E_1 = E_2$, but if, while trying to prove that it holds $E_1 \neq E_2$, one ends up with a trivial inequality ($a \neq b$ for two distinct constants a and b), then it yields $E_1 \neq E_2$ (since all the rules applied by the area method are equivalence preserving).

In one variant of the area method (implemented in GCLCprover, see 3.1), non-degeneracy conditions can be introduced not only at the

beginning (based on the hypotheses), but also during the proving process. If a side condition for the positive case of a branching elimination lemma (the one of the form $L = R$) can be proved (as a lemma), then that case is applied. Otherwise, if a side condition for the negative case (the one of the form $L \neq R$) can be proved (as a lemma), then that case is applied. Otherwise, the condition for the negative case is assumed and introduced as an additional non-degeneracy condition. Therefore, this approach includes proving subgoals (which initiate a new proving process on that new goal). However, there is no branching, so the proof is always sequential, possibly with lemmas integrated. Lemmas are being proved as separate conjectures, but, of course, sharing the construction and non-degeneracy conditions with the outer context. Note that in this variant of the method, the statement proved by the method is not exactly the one given by the user as the method *introduces* ndg-conditions.

In another variant of the method (implemented in Coq, see 3.2), if a condition for one case can be proved, then that case is applied, otherwise both cases are considered separately. Therefore, this variant may produce branching proofs (but does not generate additional ndg-conditions). Note that this variant does not change the initial statement and does not risk to introduce ndg-conditions which are not needed. Indeed, for example, in some contexts it could be the case that neither A always belongs to CD nor always it does not belong to CD, but the statement to be proved is still true in *both* cases. Using the first variant of the method, in such a case the condition $\mathcal{S}_{ACD} \neq 0$ would be added to the statement whereas the theorem could be proved without this assumption.

2.5.2. *Uniformization*

The main goal of the phase of eliminating constructed points is that all remaining geometric quantities are independent. However, this is not exactly the case, because two equal geometric quantities can be represented by syntactically different terms. For instance, $\mathcal{S}_{ABC}$ can also be represented by $\mathcal{S}_{CAB}$. To solve this issue, it is needed to uniformize the geometric quantities that appear in the statement. For this purpose, a set of conditional rewrite rules is used. To ensure termination, these rules are applied only when A, B and C stand for variables whose name are in alphabetic order.

The uniformization procedure consists of applying exhaustively the following rules:

$$
\begin{array}{llll}
\multicolumn{4}{c}{\overline{BA} \rightarrow -\overline{AB}} & \text{by Lemma 1} \\
\mathcal{S}_{BCA} \rightarrow \mathcal{S}_{ABC} & & \mathcal{S}_{ACB} \rightarrow -\mathcal{S}_{ABC} & \\
\mathcal{S}_{CAB} \rightarrow \mathcal{S}_{ABC} & & \mathcal{S}_{BAC} \rightarrow -\mathcal{S}_{ABC} & \text{by Lemma 4} \\
\mathcal{S}_{CBA} \rightarrow -\mathcal{S}_{ABC} & & & \\
\multicolumn{4}{c}{\mathcal{P}_{CBA} \rightarrow \mathcal{P}_{ABC}} & \text{by Lemma 6} \\
\multicolumn{4}{c}{\mathcal{P}_{BAB} \rightarrow \mathcal{P}_{ABA}} & \text{by Lemma 7}
\end{array}
$$

2.5.3. *Dealing with free points: area coordinates*
The elementary construction step ECS1 introduces arbitrary points. Such points are the *free points* on which all other objects are based. For a geometric statement $S = (C_1, C_2, \ldots, C_m, (E_1 = E_2))$, one can obtain two rational expressions E_1' and E_2' in ratio of directed segments, signed areas and Pythagoras differences in only *free points*, numerical constants and numerical variables. Most often this simply leads to equations that are trivially true (as in Ceva's example). However, the remaining geometric quantities can still be mutually dependent, e.g., for any four points A, B, C, and D it holds (by Axiom 6) that

$$\mathcal{S}_{ABC} = \mathcal{S}_{ABD} + \mathcal{S}_{ADC} + \mathcal{S}_{DBC}$$

In such cases, it is needed to reduce E_1' and E_2' to expressions in independent variables. For that purpose the *area coordinates* are used.

DEFINITION 7. *Let A, O, U, and V be four points such that O, U, and V are not collinear. The area coordinates of A with respect to OUV are*

$$x_A = \frac{\mathcal{S}_{OUA}}{\mathcal{S}_{OUV}}, \quad y_A = \frac{\mathcal{S}_{OAV}}{\mathcal{S}_{OUV}}, \quad z_A = \frac{\mathcal{S}_{AUV}}{\mathcal{S}_{OUV}}.$$

It is clear that $x_A + y_A + z_A = 1$.

It holds that the points in the plane are in a one to one correspondence with their area coordinates. To represent E_1 and E_2 as expressions in independent variables, first three new points O, U, and V, such that $UO \perp OV$ and $d = \overline{OU} = \overline{OV}$, are introduced. Expressions E_1 and E_2 can be transformed to expressions in the area coordinates of the free points with respect to OUV.

For any point P, let X_P denotes $\mathcal{S}_{OUP}$, let Y_P denotes $\mathcal{S}_{OVP}$, and let $Col(A, B, C)$ denotes the fact that A, B and C are collinear.

LEMMA 22. *For any points A, B, C and D such that $C \neq D$ and $AB \parallel CD$, it holds that:*

$$\frac{\overline{AB}}{\overline{CD}} = \begin{cases} \frac{X_CY_A - X_CY_B - Y_AX_B + Y_BX_A - Y_CX_A + Y_CX_B}{X_CY_A - X_CY_D - Y_AX_D - Y_CX_A + Y_CX_D + X_AY_D} & \text{if not } Col(A,C,D) \\ \frac{X_BY_A - X_AY_B}{X_DY_C - X_CY_D} & \text{if } Col(A,C,D) \text{ and not } Col(O,A,C) \\ \frac{\mathcal{S}_{OUV}(X_B - X_A) + X_BY_A - X_AY_B}{\mathcal{S}_{OUV}(X_D - X_C) + X_DY_C - X_CY_D} & \text{if } Col(A,C,D) \text{ and } Col(O,A,C) \text{ and not } Col(U,A,C) \\ \frac{\mathcal{S}_{OUV}(Y_B - Y_A) + X_BY_A - Y_BX_A}{\mathcal{S}_{OUV}(Y_D - Y_C) + X_DY_C - Y_DX_C} & \text{otherwise} \end{cases}$$

LEMMA 23. *For any points A, B and C it holds that:*
$\mathcal{S}_{ABC} = \frac{(Y_B - Y_C)X_A + (Y_C - Y_A)X_B + (Y_A - Y_B)X_C}{\mathcal{S}_{OUV}}$.

LEMMA 24. *For any points A, B and C it holds that:*
$\mathcal{P}_{ABC} = 8(\frac{Y_AY_C - Y_AY_B + Y_B^2 - Y_BY_C - X_AX_B + X_AX_C + X_B^2 - X_BX_C}{d^2})$.

LEMMA 25. $S_{OUV} = \pm\frac{d^2}{2}$.

Using lemmas 22 to 25, expressions E_1 and E_2 can be written as expressions in d^2, and in the geometric quantities of the form $\mathcal{S}_{OUP}$ or $\mathcal{S}_{OVP}$ where P is a free point (there is V such that $\mathcal{S}_{OUV} = \frac{d^2}{2}$).

After this transformation, the equality $E_1 = E_2$ is transformed into an equality over independent variables and numerical parameters.

2.5.4. *Simplification*
For simplification of the statement the following rewrite rules are applied.

Degenerated geometric quantities:

$$\frac{\overline{YY}}{\overline{AB}} \to 0 \qquad \begin{array}{ll} \mathcal{S}_{AAB} \to 0 & \mathcal{P}_{AAB} \to 0 \\ \mathcal{S}_{BAA} \to 0 & \mathcal{P}_{BAA} \to 0 \\ \mathcal{S}_{ABA} \to 0 & \end{array}$$

Ring simplifications:

$$\begin{array}{llllll} a \cdot 0 \to 0 & 0 + a \to a & -0 \to 0 & (-a) \cdot b \to -(a \cdot b) \\ 0 \cdot a \to 0 & a + 0 \to a & --a \to a & a \cdot (-b) \to -(a \cdot b) \\ 1 \cdot a \to a & a - 0 \to a & -a + a \to 0 & -a \cdot -b \to a \cdot b \\ a \cdot 1 \to a & 0 - a \to -a & a + (-b) \to a - b & \\ & a - a \to 0 & -b + a \to a - b & \end{array}$$

$c_1 + c_2 \rightarrow c_3$ where c_1 and c_2 are constants (elements of F) and $c_1 + c_2 = c_3$

$c_1 \cdot c_2 \rightarrow c_3$, where c_1 and c_2 are constants (elements of F) and $c_1 \cdot c_2 = c_3$

Field simplifications (if $a \neq 0$):

$$\begin{array}{llll} \frac{a}{a} \rightarrow 1 & \frac{0}{a} \rightarrow 0 & \frac{-b}{a} \rightarrow -\frac{b}{a} \\ \frac{a}{-a} \rightarrow -1 & \frac{a}{1} \rightarrow a & \frac{b}{-a} \rightarrow -\frac{b}{a} \\ \frac{-a}{a} \rightarrow -1 & a \cdot (\frac{1}{a}) \rightarrow 1 & \frac{a \cdot b}{a} \rightarrow b \\ \frac{-a}{-a} \rightarrow 1 & & \frac{b \cdot a}{a} \rightarrow b \end{array}$$

2.5.5. *Deciding equality of two rational expressions*
After the elimination of constructed points, uniformization of geometric quantities, treatment of the free points, and the simplification, an equality between two rational expressions involving only independent quantities is obtained. To decide validity of such an equality (by transforming its two sides), the following (terminating) rewrite rules are used.

Reducing to a single fraction:

$$\begin{array}{llll} \frac{a}{b} + c \rightarrow \frac{a+c \cdot b}{b} & a \cdot \frac{b}{c} \rightarrow \frac{a \cdot b}{c} & \frac{a}{\frac{b}{c}} \rightarrow \frac{a \cdot c}{b} \\ c + \frac{a}{b} \rightarrow \frac{c \cdot b + a}{b} & \frac{a}{b} \cdot c \rightarrow \frac{a \cdot c}{b} & \frac{\frac{a}{b}}{\frac{c}{1}} \rightarrow \frac{a}{b \cdot c} \\ \frac{a}{b} + \frac{c}{b} \rightarrow \frac{a+c}{b} & \frac{a}{b} \cdot \frac{c}{d} \rightarrow \frac{a \cdot c}{b \cdot d} & \frac{\frac{a}{b}}{\frac{c}{d}} \rightarrow \frac{a \cdot d}{c \cdot b} \\ \frac{a}{b} + \frac{c}{d} \rightarrow \frac{a \cdot d + c \cdot b}{bd} & & \end{array}$$

Reducing to an equation without fractions:

$$\begin{array}{ll} \frac{a}{b} = c \rightarrow a = c \cdot b & \frac{a}{b} = \frac{c}{b} \rightarrow a = c \\ c = \frac{a}{b} \rightarrow c \cdot b = a & \frac{a}{b} = \frac{c}{d} \rightarrow a \cdot d = c \cdot b \end{array}$$

Reducing to an equation where the right hand side is zero:

$$a = c \rightarrow a - c = 0$$

Reducing left hand side to right associative form:

$$\begin{array}{ll} ((a+b)+c) \rightarrow a + (b+c) & a \cdot (b+c) \rightarrow a \cdot b + a \cdot c \\ ((a \cdot b) \cdot c) \rightarrow a \cdot (b \cdot c) & (b+c) \cdot a \rightarrow b \cdot a + c \cdot a \end{array}$$

$a \cdot c \rightarrow c \cdot a$, where c is a constant (element of F) and a is not a constant.

$a \cdot (c \cdot b) \rightarrow c \cdot (a \cdot b)$ where c is a constant (element of F) and a is not a constant.

$c_1 \cdot (c_2 \cdot a) \rightarrow c_3 \cdot a$ where c_1 and c_2 are constants (elements of F) and $c_1 \cdot c_2 = c_3$.

$E_1 + \cdots + E_{i-1} + c_1 \cdot C + E_{i+1} + \cdots + E_{j-1} + c_2 \cdot C' + E_{j+1} + \cdots + E_n \rightarrow E_1 + \cdots E_{i-1} + c_3 \cdot C + E_{i+1} + \cdots + E_{j-1} + E_{j+1} + \cdots + E_n$, where c_1, c_2 and c_3 are constants (elements of F) such that $c_1 + c_2 = c_3$ and C and C' are equal products (with all multiplicands equal up to permutation).

The above rules are used in the "waterfall" manner: they are tried for applicability, and when one rule is (once) applied successfully, then the list of the rules is tried from the top. The ordering of the rules can impact the efficiency to some extent.

The original equality is valid if and only if it is transformed to $0 = 0$.

Note that all the rules involving ratios given above can be applied to ratios of directed segments (as, following the axiom system given in Section 2.2.2), ratios of directed segments *are* ratios over F. Since these rules are applied after the elimination process, there is no danger of leaving segment lengths involving constructed points (by breaking some ratios of segments). However, in this approach all ratios are handled only at the end of the proving process. To increase efficiency, it is possible to use these rules during the proving process. Namely, all the rules involving ratios can be used also in the simplification phase, but not applied to ratios of segments (they are treated as special case of ratios). The first approach is implemented in Coq (see 3.2), the second in GCLCprover (see 3.1).

The set of rules given above is not minimal, in a sense that some rules can be omitted and the procedure for deciding equality would still be complete. However, they are used for efficiency. Also, additional rules can be used, as long as they are terminating and validity and invalidity preserving.

2.5.6. *Non-degeneracy Conditions*

Some constructions are possible only if certain conditions are met. For instance, the construction of the intersection of lines a and b is possible only if the lines a and b are not parallel. For such constructions ndg-conditions are stored and considered during the proving process. Non-degeneracy conditions of the construction steps have one of the following two forms:

- $A \neq B$ or, equivalently, $\mathcal{P}_{ABA} \neq 0$;
- $PQ \nparallel UV$ or, equivalently, $\mathcal{S}_{PUV} \neq \mathcal{S}_{QUV}$;

Negations of these conditions have to be checked during the proving process. As seen from above, these negations can be expressed as equalities in terms of geometric quantities and in some cases can be proved by the area method itself.

A ndg-condition of a geometry statement is the conjunction of ndg-conditions of the corresponding construction steps, plus the conditions that the denominators of the ratios of parallel directed segments in the statement are not equal to zero, and the conditions that $AB \parallel CD$ for every ratio $\frac{\overline{AB}}{\overline{CD}}$ that appear in the statement. As said in Section 2.3.2, the statement is proved with the assumption that its ndg-conditions are satisfied. Hence, if the negation of a ndg-condition of a geometry statement is met, the statement is trivially valid.

As an example, consider a theorem about an *impossible construction*. Let A, B and C be three arbitrary points (obtained by ECS1). Let D be on the line parallel to AB passing through C (obtained by ECS4). Let I be the intersection of AB and CD (obtained by ECS2). Then, the assumptions of any statement G to be proved about these points are inconsistent since the construction of D implies $AB \parallel CD$ and the construction of I implies $AB \nparallel CD$. Therefore, G is trivially valid.

2.5.7. *Algorithm*

The area method checks whether a constructive geometry statement $(C_1, C_2, \ldots, C_m, E_1 = E_2)$ is valid or not, i.e., it checks whether $E_1 = E_2$ is a deductive consequence of the construction $(C_1, C_2, \ldots, C_m)$, along with its ndg-conditions. As said, the key part of the method is eliminating constructed points from geometric quantities. The point are introduced one by one, and are eliminated from the goal expression in the reverse order.

Algorithm: Area method

Input: $S = (C_1, C_2, \ldots, C_m, (E_1 = E_2))$ is a statement in **C**.

Output: The algorithm checks whether S is valid or not and produces a corresponding proof (consisting of all single steps performed).

1. initially, the current goal is the given conjecture; translate the goal in terms of geometric quantities using table 2.2.1 and generate all ndg-conditions for S;
2. process all the construction steps in reverse order:
   a) if the negation of the ndg-condition of the current construction step is met, then exit and report that the conjecture is trivially valid; otherwise, this ndg-condition is one of the assumptions of the statement.
   b) simplify the current goal (by using the simplification procedure);

   c) if the current construction step introduces a new point P, then eliminate (by using Lemma 8 and the elimination lemmas) all occurrences of P from the current goal;

3. uniformize the geometric quantities (using the uniformization rules);
4. simplify the current goal (by using the simplification procedure);
5. test if the obtained equality is valid (by using the procedure given in 2.5.5); if yes, then the conjecture $E_1 = E_2$ is valid, under the assumption that the ndg-conditions hold, otherwise:
   a) eliminate the free points (using the area coordinates, as described in 2.5.3);
   b) simplify the current goal (by using the simplification procedure);
   c) test if the obtained equality is valid (by using the procedure given in 2.5.5); if yes, then the conjecture $E_1 = E_2$ is valid, under the assumption that the ndg-conditions hold. Otherwise the conjecture is not valid.

Testing the validity of ndg-conditions within the main loop can also be performed by the area method itself (based on the construction steps that precede the current step).

2.5.8. *Properties of the Method*
The area method is terminating, sound, and complete: it can prove any geometry theorem expressed in terms of geometric quantities, and involving only objects introduced by using a specified set of constructions steps. Therefore, the procedure is a decision procedure for the described fragment of geometry.[5]

Termination. Since there is a finite number of constructed points, there is a finite number of occurrences of these points in the statement, and since in each application of the elimination lemmas there is at least one occurrence of a constructed points eliminated, it follows that all constructed points will be eventually eliminated from the statements. Therefore, if the simplification procedure and the procedure for deciding equality over independent parameters terminate, the whole of the method terminates as well.

[5] This fragment can also be defined as a quantifier-free theory with the set of axioms equal to the set of all introduced lemmas. It can be easily shown that this theory is a sub-theory of Euclidean geometry augmented by the theory of fields.

Correctness. All steps of the method transform the current goal. All steps are based on the (proved) lemmas, so these transformation preserve the validity of the goal: the goal is valid after one step if and only if it was valid before that step. Therefore, it remains to consider only the last step of the algorithm. If $E_1 = E_2$, then the original statement is obviously valid. Note that the ndg-conditions ensure that the denominators of all the expressions occurring in the proof are different from zero. Otherwise, if $E_1 \neq E_2$, since all geometric quantities occurring in E_1 and E_2 are free parameters, in the geometric construction considered they can take arbitrary values. So, it is possible to choose concrete values for these quantities leading to $E_1 \neq E_2$, and a counterexample for the statement. Hence, in this case, the statement is not valid. Therefore, the method is both sound and complete: it returns the positive answer (along with the proof) if and only if the given conjecture is valid.

Complexity The core of the method does not have branching (unless the variant with considering both cases in ndg-conditions is used, as explained in Section 2.5.6), which makes it very efficient for many non-trivial geometry theorems (still, the area method is less efficient than provers based on algebraic methods (Chou et al., 1994)).

The area method can transform a conjecture given as an equality between rational expressions involving constructed points, to an equality not involving constructed points. Each application of elimination lemmas eliminates one occurrence of a constructed point and replace a relevant geometric quantity by a rational expression with a degree less than or equal to two. Therefore, if the original conjecture has a degree d and involves n occurrences of constructed points, then the reduced conjecture (without constructed points) has a degree of at most 2^n (Chou et al., 1994). However, this degree is usually much less, especially if the simplification procedures are used along the elimination process. The above analysis does not take into account the complexity of the elimination of free points and the simplification process.

3. Implementations of the Area Method

In this section we describe specifics of our two (independent) implementations of the area method and briefly describe other two implementations. We also describe some applications of these implementations.

3.1. The Area Method in GCLC

A theorem prover GCLCprover, based on the area method, is a part of a dynamic geometry tool GCLC. This section begins with a brief description of GCLC.

3.1.1. *GCLC*

GCLC (Janičić, 2006; Janičić, 2009) is a tool for visualisation of objects and notions of geometry and other fields of mathematics. The primary focus of the first versions of the GCLC was producing digital illustrations of Euclidean constructions in LaTeX form (hence the name "**G**eometry **C**onstructions → LaTeX **C**onverter"), but now it is much more than that. For instance, there is support for symbolic expressions, for parametric curves and surfaces, for drawing functions, graphs, and trees, support for flow control, etc. Libraries of GCLC procedures provide additional features, such as support for hyperbolic geometry.

The basic idea behind GCLC is that constructions are abstract, formal procedures, rather than images. Thus, in GCLC, producing mathematical illustrations is based on "describing figures" rather than on "drawing figures". A figure can be generated (in the Cartesian model of the plane) on the basis of the abstract description.

The language of GCLC (Janičić, 2009) consists of the following groups of commands: basic definitions (e.g., `point` for introducing a point, `line` for a line determined by two point), basic constructions (e.g., `intersec` for constructing the intersection of two lines), transformations (e.g., `translate` for translation), commands for symbolic calculations, commands for flow control, drawing commands, labelling and printing commands, Cartesian commands, low level commands, commands for describing animations, and commands for automated theorem proving.

EXAMPLE 3.1. *The example GCLC code given in Figure 2 (left) describes a triangle and the midpoints of two of triangle's sides. From this GCLC code, Figure 2 (right) can be generated.*

Apart from producing digital mathematical illustrations (in different formats), GCLC can be used for teaching and studying geometry (and not only geometry), and for storing visual mathematical contents in textual form (as figure descriptions in the underlying language).

GCLC has been under constant development since 1996. It is implemented in C++, and consist of around 40.000 lines of code. WinGCLC is a version with a *MS-Windows* graphical interface that makes GCLC a dynamic geometry tool with a range of additional functionalities (Figure 3).

```
point A 20 10
point B 70 10
point C 35 40

midpoint B' B C
midpoint A' A C

drawsegment A B
drawsegment A C
drawsegment B C
drawsegment A' B'

cmark_b A
cmark_b B
cmark_t C
cmark_l A'
cmark_r B'
```

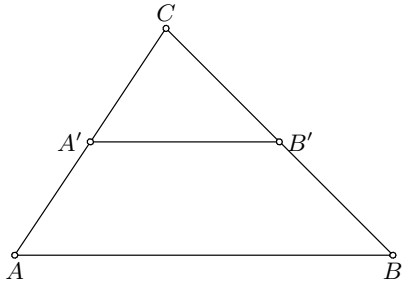


Figure 2. A description of a triangle and midpoints of two of triangle's sides in GCLC language (left) and the corresponding illustration (right)

3.1.2. *Integration of the Area Method*

GCLC has three geometry theorem provers for Euclidean constructive theorems built in: a theorem prover GCLCprover based on the area method[6] (Janičić and Quaresma, 2006) and algebraic theorem provers based on the Gröbner bases method and on the Wu's method[7] (Predović, 2008). Thanks to these theorem provers, GCLC links geometrical contents, visual information, and machine–generated proofs.

The provers are tightly integrated in GCLC. This means that one can use the prover to reason about a GCLC construction (i.e., about objects introduced in it) without any adaptations to the deduction process other then the addition of the conjecture itself. For this purpose, the provers use the standard GCLC construction commands. GCLCprover deals with the subset of GCLC construction commands (e.g., it does not deal with intersections of two circles). If needed, GCLCprover transforms a construction command into a form required by the area method and/or introduces some auxiliary points. For example, the GCLC command `med m A B` that introduces the segment bisector `m` of the segment with endpoints `A` and `B` is dealt with, in the following way: internally, two auxiliary points are introduced — a point M_m such that (PRATIO M_m A (LINE A B) 1/2) and a point T_m such that TRATIO T_m (LINE M_m A) 1); the line m is then, within the prover, determined by the points M_m and T_m. A conjecture to be proved may involve only points and

[6] This theorem prover was developed by Predrag Janičić and Pedro Quaresma.
[7] These theorem provers were developed by Goran Predović and Predrag Janičić.

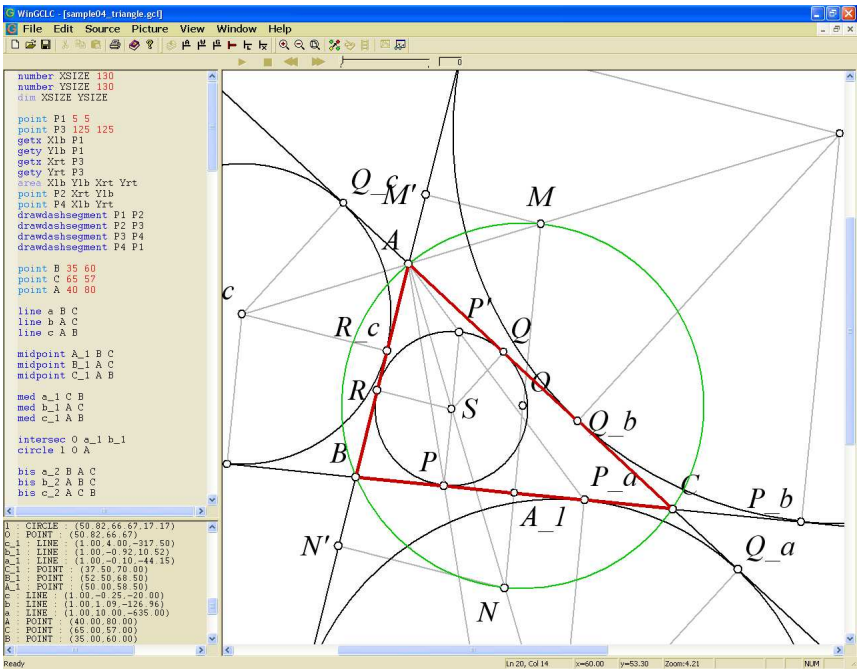


Figure 3. WinGCLC Screenshot

lines already introduced within the current construction. A conjecture is given as argument to the `prove` command. It has to be of the form $L = R$, where L and R are expressions over geometric quantities, which can be combined together into more complex terms by operators for addition, multiplication and division (written `sum`, `mult`, `ratio`). The conjecture and all its sub-terms, are written in prefix form, for instance, $\mathcal{S}_{A'B'A} = \mathcal{S}_{A'B'B}$ is given in the following way:

```
prove { equal { signed_area3 A' B' A }
              { signed_area3 A' B' B }
      }
```

Alternatively, a conjecture can be given in the form of some higher-level properties (e.g., `prove { parallel A B A' B' }` and in such cases it is internally transformed into a statement in terms of geometric quantities (following properties from Table 2.2.1). The prover considers only abstract specification of the conjecture and do not consider Cartesian values of the points involved (they are used only for visualisation).

For the construction shown in Example 3.1, it holds that the lines AB and $A'B'$ are parallel and this can be proved by the theorem prover. This property can be given as argument to the `prove` command: `prove`

`{ parallel A B A' B'}`, after the description of the construction. The prover is invoked at the end of processing of the GCLC file.

Support for the prover involves only five commands: `prove`, to state the conjecture, `prooflevel` used, optionally, for choosing one of the eight levels of detail for the output (see 3.1.4), `prooflimit` for controlling the maximal number of proof steps, `prover_timeout` for fixing a time limit to the prover, and `theorem_name` for setting the name of the theorem (later used in prover's output documents).

3.1.3. *Specifics of the Implementation in GCLC*
The algorithm implemented in GCLCprover is the one described in Section 2.5.7, with the following specifics, all introduced for increasing efficiency.

3.1.3.1. *Simplification procedure.* With respect to the simplification procedure described in 2.5.4, there are the following specifics in the variant implemented within GCLCprover:

- The unary operator $-$ is not used (and instead $-x$ is represented as $(-1)\cdot x$). Hence, the rules involving this operator are not used. This does not affect the correctness of the method, but simplifies the implementation.
- The rules given in 2.5.5 are used also within the simplification procedure, but the rules involving fractions are not applied to ratios of segments. Because of that, the following additional rules are used within the simplification procedure:
  - $\frac{\overline{AB}}{\overline{AB}} \rightarrow 1$
  - $\frac{\overline{AB}}{\overline{BA}} \rightarrow -1$
- The following rules are used within the simplification phase:
  - $\frac{x}{c} \rightarrow (1/c)\cdot x$, where c is a constant (element of F) and $c \neq 1$.
  - $\frac{E_1\cdot\ldots\cdot E_{i-1}\cdot C\cdot E_{i+1}\cdot\ldots\cdot E_n}{E'_1\cdot\ldots\cdot E'_{j-1}\cdot C\cdot E'_{j+1}\cdot\ldots\cdot E'_m} \rightarrow \frac{E_1\cdot\ldots\cdot E_{i-1}\cdot E_{i+1}\cdot\ldots\cdot E_n}{E'_1\cdot\ldots\cdot E'_{j-1}\cdot E'_{j+1}\cdot\ldots\cdot E'_m}$
  - $E_1+\cdots+E_{i-1}+c_1\cdot C+E_{i+1}+\cdots+E_n = E'_1+\cdots+E'_{j-1}+c_2\cdot C'+E'_{j+1}+\cdots+E'_m \rightarrow E_1+\cdots+E_{i-1}+c_3\cdot C+E_{i+1}\cdots+E_n = E'_1+\cdots+E'_{j-1}+E'_{j+1}+\cdots+E'_m$
    where c_1, c_2, and c_3 are constants (elements of F) such that $c_1 - c_2 = c_3$ and C and C' are equal products (with all multiplicands equal up to permutation).

- If the current goal is of the form $E_1 + \ldots + E_n = E'_1 + \ldots E'_m$ and if all summands E_i and E'_j have a common multiplication factor X, then try to prove that it holds $X = 0$:
  - if $X = 0$ has been proved, the current goal can be rewritten to $0 = 0$;
  - if $X = 0$ has been disproved (i.e., if $X \neq 0$ has been proved), then both sides in the current goal can be cancelled by X;
  - if neither $X = 0$ nor $X \neq 0$ can be proved, then assume $X \neq 0$ (and add to the list of non-degeneracy conditions) and cancel both sides in the current goal by X.

– The uniformization procedure (2.5.2) is used within the simplification procedure. In addition, if three points A, B, C are collinear, then the rule $\mathcal{S}_{ABC} \rightarrow 0$ is applied.

– Reducing to area coordinates is not implemented. Instead, the following rules are applied at that stage:

  - $\overline{AA} \rightarrow 0$
  - $\mathcal{S}_{ABC} \rightarrow \mathcal{S}_{ABD} + \mathcal{S}_{ADC} + \mathcal{S}_{DBC}$ (by Axiom 6), if there are terms $\mathcal{S}_{ABD}$, $\mathcal{S}_{ADC}$, $\mathcal{S}_{DBC}$ in the current goal.
  - $\mathcal{P}_{ABC} \rightarrow \overline{AB}^2 + \overline{CB}^2 + -1 \cdot \overline{AC}^2$ (by Definition 3)

  Note that after these rules applied, the equality being proved may still involve dependent parameters. Still, the simplification process is applied again and the equality is tested for validity for the last time. Even without reducing to area coordinates, the above rules enable proving most conjectures from the area method scope.

3.1.3.2. *Dealing with ndg-conditions.* The prover records and reports about the ndg-conditions of the construction steps, but there is no check of ndg-conditions within the main loop. That check is not necessary in this context, i.e., within GCLC. Namely, when using GCLC, the user describes a construction and then provides a statement about the constructed objects to be proved. The construction is visualised for a set of free points with concrete Cartesian coordinates. For each construction step, it is checked if it is possible (e.g., if two lines do intersect) and the test corresponds to the ndg-condition of the construction step. If some of these checks fails, an error is reported, the construction is not visualised, and the conjecture is not sent to the prover. In that case, one of the ndg-conditions is false in the concrete model. Otherwise, all

the ndg-conditions are true in the concrete model, and hence, none of their negations can be valid, so the check of ndg-conditions (as given in section 2.5.7) is not needed.

3.1.3.3. *Dealing with side conditions.* If a side condition for one case of a branching elimination lemma can be proved, then that case is applied, otherwise, a condition for the negative case is assumed and introduced as an additional ndg-condition (as explained in Section 2.5.1). The same approach is used when applying the cancellation rule (see section 3.1.3).

Thanks to the powerful simplification procedure, efficient implementation in C++ and to the fact that there are no branching in the proofs, GCLCprover is very efficient and can prove many complex theorems in only milliseconds (for examples see the GeoThms web repository (described in Section 3.4.1).

3.1.4. *Prover Output*

The proofs generated by GCLCprover can be exported to LaTeX or to XML form using a special-purpose styles, with explanations for each proof step.[8]

At the beginning of the proof, the auxiliary points are defined, for instance:

Let M_a^0 be the midpoint of the segment BC. Let T_a^1 be the point on bisector of the segment BC (such that TRATIO T_a^1 M_a^0 B 1).

For each proof step (a single transformation of the goal being proved), there is an explanation and, optionally, its semantics counterpart — as a check whether a conjecture is valid in the specific case, determined by the given Cartesian points. This semantic information is calculated for concrete points used in the construction for visualisation purposes (these Cartesian coordinates are never used in the proof itself); it can serve as a semantic test, especially for conjectures for which is not known

[8] There are no object-level proofs verifiable by theorem proving assistants.

whether or not they are theorems. All proof steps are enumerated, for example:

$$\left(\left(\frac{\overline{AF}}{\overline{FB}} \cdot \frac{\overline{BD}}{\overline{DC}}\right) \cdot \frac{\overline{CE}}{\overline{EA}}\right) = 1 \quad \text{by the statement} \quad (1)$$
$$\left(\left(\left(-1 \cdot \frac{\overline{AF}}{\overline{BF}}\right) \cdot \frac{\overline{BD}}{\overline{DC}}\right) \cdot \frac{\overline{CE}}{\overline{EA}}\right) = 1 \quad \text{by geometric simplifications} \quad (2)$$

Lemmas (about side conditions) are proved within the main proof (making nested proof levels). After the proof steps, all non-degeneracy conditions are listed, for instance:

$S_{BPA} \neq S_{CPA}$ i.e., lines BC and PA are not parallel (construction based assumption)

At the end, the output document includes a short report, consisting of information on whether the conjecture was proved or disproved (or neither), data about CPU time spent, and the number of proof steps performed (in several categories).

The style for proofs formatted in LaTeX has options for different formatting. Proofs stored in XML are structured analogously as in LaTeX format. The proofs in XML format fulfil restrictions posed by a custom DTD file. For any XML file, it can be checked if it meets these restrictions (by a XML processor). A proof in XML format can be converted to a HTML form. A file with a proof in XML format can also be open directly by web browsers.

3.1.5. *Example*

In this section we give a fragment of the output for the conjecture from Example 3.2.

$S_{AA'B'}$ $= S_{BA'B'}$ by the statement (1)

$S_{B'AA'}$ $= S_{B'BA'}$ by geometrical simplifications (2)

$\left(S_{B'AA} + \left(\frac{1}{2} \cdot (S_{B'AC} + (-1 \cdot S_{B'AA}))\right)\right)$ $= S_{B'BA'}$ by Lemma 29 (point A' eliminated) (3)

...

0 $= \left(0 + \left(\frac{1}{2} \cdot (0 + (-1 \cdot 0))\right)\right)$ by geometrical simplifications (15)

0 $= 0$ by algebraic simplifications (16)

Q.E.D.

There are no ndg conditions.

Number of elimination proof steps: 5
Number of geometrical proof steps: 15
Number of algebraic proof steps: 25
Total number of proof steps: 45
Time spent by the prover: 0.001 seconds

3.2. The Area Method in *Coq*

This section describes the formalisation of the area method using the proof assistant *Coq*. *Coq* is a general purpose proof assistant (The Coq development team, 2009; Huet et al., 2004; Bertot and Castéran, 2004). It allows expressing mathematical assertions and to mechanically check proofs of these assertions.

3.2.1. *Coq*

We begin the description of the formalisation with a brief description of *Coq* and how decision procedures can be formalised in *Coq*. Although the *Coq* system has some automatic theorem proving features, it is *not* an automatic theorem prover. The proofs are mainly built by the user *interactively*. The system allows formalising proofs in different domains. For instance, it has been used for the formalisation of the four colour theorem (Gonthier and Werner, 2004) and the fundamental theorem of algebra (Geuvers and et.al., 2008). In computer science, it can be used to prove correctness of programs, like a C compiler that has been developed and proved correct using *Coq* (Leroy, 2006).

There are several recent results in the formalisation of elementary geometry in proof assistants: Hilbert's *Grundlagen* (Hilbert, 1977) has been formalised in Isabelle/Isar (Meikle and Fleuriot, 2003) and in *Coq* (Dehlinger et al., 2000). Gilles Kahn has formalised Jan von Plato's constructive geometry in the *Coq* system (Kahn, 1995; von Plato, 1995). Frédérique Guilhot has made a large development in *Coq* dealing with French high school geometry (Guilhot, 2004). Julien Narboux has formalised Tarski's geometry using the *Coq* proof assistant (Narboux, 2007b). Jean Duprat proposes the formalisation in *Coq* of an axiom system for compass and ruler geometry (Duprat, 2008). Projective geometry has also been formalised in *Coq* (Magaud et al., 2008; Magaud et al., 2009).

Implementing decision procedures in Coq There are three methods to add automation to the *Coq* system:

1. directly in the implementation language of *Coq* — Ocaml;
2. using the tactic[9] language of *Coq* — $\mathcal{L}_{tac}$;
3. by reflection using *Coq* as a programming language.

This third method, introduced by Samuel Boutin (Boutin, 1997), consists of formalising a subset of the language of *Coq* using an object of

[9] A tactic is a program which expresses the sequence of the basic logical steps needed to formally prove a theorem.

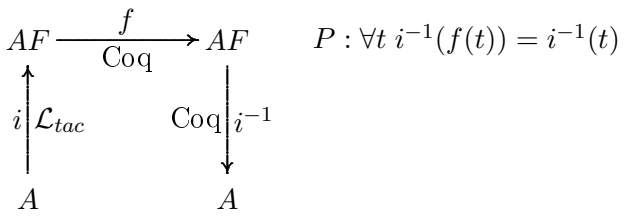


Figure 4. The Reflection mechanism.

Coq itself. The computations that can be done using the meta language (Ocaml or $\mathcal{L}_{tac}$) are performed using the *Coq* language itself. Figure 4 represents the reflection mechanism in the case of a tactic which applies a rewrite rule. A reflexive tactic is composed of four elements:

i**:** a piece of code written in $\mathcal{L}_{tac}$ (or in Ocaml) to translate a *Coq* term into an object of *Coq*;

f**:** a *Coq* function which perform the computations to solve the given problem;

i^{-1}**:** a *Coq* function which translates back from the universe of *Coq* objects to the universe of *Coq*. Note that it is necessary that $i^{-1}(i(t)) \rightarrow t$ holds, but this fact does not need to be proven *formally*;

P**:** the formal proof that the translation realised by f is correct.

This method has the advantage to produce tactics that are more efficient and that produce shorter proofs, since the application of the tactic is recorded in the proof just as a step of computations. For more information on the reflexive proof method, see, for instance, Chapter 16 of the book *Coq'Art* (Bertot and Castéran, 2004).

3.2.2. *Formalisation of the Area Method*

The goal of the formalisation of the area method (in *Coq*) is to bring the level of automation provided by the method to the *Coq* proof assistant. This is done by implementing the decision procedure as a *Coq* tactic and formalising all theorems needed by the method. We defined an axiom system, proved all the propositions needed by the tactics (we formally proved more than 700 lemmas) and wrote the tactics.

Conceptually, proving the propositions and writing the tactics that use them seem to be two separate tasks. But to ease the development, in our implementation we have intermixed the proofs of the propositions and the tactics. We bootstrap partially the construction of the whole decision procedure by using some automatic tactics for

the proof of the elimination lemmas. Our tactic is decomposed into sub-tactics performing the following tasks: initialisation; simplification; uniformization; elimination of constructed points; elimination of free points; conclusion.

The implementation of the prover is realized mainly using the language $\mathcal{L}_{tac}$ which is integrated in the system *Coq*. Still, some sub-tactics (for instance the simplification tactics) are implemented using the reflection mechanism. We chosed not to use the reflection tactic for the whole decision procedure for two reasons:

1. We believe that the efficiency of the method would not have been increased significantly. Indeed, the proof generated by our tactic consists mainly of a sequence of application of elimination lemmas.

2. Expressing the tactic as a *Coq* function and proving its correctness would have been a very difficult task, as we make heavy use of the high level primitives of the language $\mathcal{L}_{tac}$ such as pattern matching, deleting hypotheses, etc. To use the reflection method for the whole algorithm, the whole machinery and the proof of its correction should have been realized using *Coq*.

Consequently, we did not proved formally the completeness of the method implementation (i.e., that the tactic always succeeds if the theorem is valid). Our formal proofs guaranty only the soundness of the method implementation (i.e., the proofs generated by the tactic are always corrects).

3.2.3. *Specifics of the Implementation in Coq*

In this section, we describe the algorithm which is used in the *Coq*'s implementation of the area method.

As the method is implemented within a proof assistant, each step of the algorithm correspond to a proof step that is checked by the *Coq* system. At the end of the proof, it is checked another time by the *Coq* kernel as explained in section 3.2.6. The main difficulty is that *Coq* must be "convinced" at each step that the transformation we perform is correct. For this we have to maintain two invariants:

1. For each *syntactic* expression which occurs at the denominator of some fraction, the context always contains a proof that it is non zero.

2. For each *syntactic* expression which represents a ratio of directed segments $(\overline{AB}/\overline{CD})$, the context always contains a proof that AB is parallel to CD.

The algorithm implemented in *Coq* corresponds to the algorithm described in Section 2.5.7. We give details only for the phases with specific features.

Initialisation The initialisation phase performs the following tasks:

1. unfold definitions;
2. introduce hypotheses in the context;
3. encode constructions of half-free points (points that belong to a line or a circle) into constructions of fixed point with a parameter;
4. compose simple constructions into more complex constructions when it is possible;
5. transform hypotheses of the form $A \neq B$ into $\overline{AB} \neq 0$
6. split conjunctions in the goal *i.e.* decompose conjunctions in the goal into several goals;
7. check that the invariants are initially verified.

Dealing with Non-degeneracy Conditions and Case Splits in Lemmas As GCLC, the Coq implementation does not deal with ndg conditions, we assume that the statement is not contradictory.

Concerning case splits in elimination lemmas, new ndg-conditions are not generated (unlike in GCLCprover) and, instead, case distinction is performed (as explained in Section 2.5.1).

3.2.4. *Example*
We now give a detailed description of how the tactic works on the example 3.2 by decomposing the procedure into small steps[10].

The midpoint theorem is stated using our language in the syntax of *Coq* as follows:

EXAMPLE 3.2.

```
Theorem midpoint_A :
 forall A B C A' B' : Point, midpoint A' B C ->
   midpoint B' A C -> parallel A' B' A B.
geoInit.
```

[10] These steps are not exactly the same steps as those executed by our automatic procedure (the automatic procedure may treat the points in another order, and perform more simplification and unification steps).

```
1 subgoal
  A : Point
  B : Point
  C : Point
  A' : Point
  B' : Point
  H : on_line_d A' B C (1 / 2)
  H0 : on_line_d B' A C (1 / 2)
  ============================
   S A' A B' + S A' B' B = 0
```

on_line_d A' B C (1/2) states that A' is on line BC and $\frac{\overline{BA'}}{\overline{BC}} = \frac{1}{2}$.

At this step it would be enough to type `area_method` to solve the goal using our decision procedure, but for this presentation we mimic the behaviour of the decision procedure using our sub-tactics. We give the name of the sub-tactics on the left, and *Coq* output on the right[11]:

```
geoInit.
```
```
H : on_line_d A' B C (1 / 2)
H0 : on_line_d B' A C (1 / 2)
============================
 S A' A B' + S A' B' B = 0
```

```
eliminate B'.
```
```
H : on_line_d A' B C (1 / 2)
============================
 1 / 2 * S A' A C + (1 - 1 / 2) * S A' A A +
 (1 / 2 * S B A' C + (1 - 1 / 2) * S B A' A) = 0
```

```
basic_simpl.
```
```
H : on_line_d A' B C (1 / 2)
============================
 1 / 2 * S A' A C + (1 / 2 * S B A' C + 1 / 2 * S B A' A) = 0
```

```
eliminate A'.
```
```
 ============================
 1 / 2 * (1 / 2 * S A C C + (1 - 1 / 2) * S A C B) +
 (1 / 2 * (1 / 2 * S C B C + (1 - 1 / 2) * S C B B) +
  1 / 2 * (1 / 2 * S A B C + (1 - 1 / 2) * S A B B)) = 0
```

```
basic_simpl.
```
```
============================
 1 / 2 * (1 / 2 * S A C B) + 1 / 2 * (1 / 2 * S A B C) = 0
```

```
uniformize.
```
```
============================
 1 / 2 * (1 / 2 * S A C B) + 1 / 2 * (1 / 2 * - S A C B) = 0
```

```
field_and_conclude.
```
```
Proof completed.
```

[11] For this presentation the fact that A, B, C, A', and B' are of type `Point` has been removed from the context.

3.2.5. *Prover Output*
The main comparative feature of the implementation in *Coq* is that it produces formal proofs. It was built with that main motivation (unlike GCLCprover which aims at producing proofs efficiently).

The output of the formalisation in *Coq* is a formal proof. More precisely, it is a term of the calculus of inductive constructions which records all the details of the proof. These formal proofs are not readable, hence to have a readable proof we also output a human readable version of the proofs in a textual format in the console. For instance, for the example given above, the following output is generated:

```
Area method:
   initialisation...
   elimination...
   elimination of point : B'
   we need to show that:
(1 / 2 * S A' A C = 1 / 2 * S A' B C + 1 / 2 * S A' B A)
   elimination of point : A'
   we need to show that:
(1 / 2 * (1 / 2 * S A C B) = 1 / 2 * (1 / 2 * S B A C))
   uniformize areas...
   simplification...
   before field...
```

3.2.6. *Benefits of the Formalisation*
Formalising a decision procedure within a proof assistant, has not only the advantage of simplifying the tedious task of (rigorously) proving geometry theorems but also allows us to combine the geometry proofs provided by the tactic with arbitrary complicated proofs developed interactively using the full strength of the underlying logic of the theorem prover. For instance, theorems involving induction over the number of points can be formalised in *Coq*. This approach has also the advantage of providing a higher level of reliability than *ad hoc* theorem provers, because the proofs generated by tactics are double checked by the *Coq* internal proof-checker (the *Coq* system as a whole and its kernel). Namely, since it is possible that *Coq* itself contains a bug, the *Coq* system is, to reduce this risk, built using the de Bruijn's principle: only a small part of the system called the *kernel* is trusted. All the proofs generated are checked by the kernel. If there is a bug outside the kernel, the system can fail, but it guarantees the soundness (i.e., it does not allow proving an invalid statement).

During formalisation of the area method, we found two potential sources of incorrectness.

First, during proving, we discovered one mistake in the original descriptions (Chou et al., 1993): in lemma EL12 the factor 2 before $\mathcal{P}_{WUV}$ was missing.

Second, when proving the invariant that elimination lemmas transform always well defined geometric quantities into an expression involving only well defined geometric quantities, we noticed that some elimination lemmas require a non degeneracy condition. Let us consider Lemma EL3: if Y is introduced by (Pratio Y R (Line P Q) r), then it holds

$$\frac{\overline{AY}}{\overline{CD}} = \begin{cases} \frac{\frac{\overline{AR}}{\overline{PQ}}+r}{\frac{\overline{CD}}{\overline{PQ}}} & \text{if } A \text{ is on } RY \\ \frac{\mathcal{S}_{APRQ}}{\mathcal{S}_{CPDQ}} & \text{otherwise} \end{cases}$$

If $A = Y$, it can be the case that $CD \not\parallel PQ$. This demonstrates that the lemma is valid only if $A \neq Y$ and otherwise the ratio $\frac{\overline{CD}}{\overline{PQ}}$ is not well defined. Hence, during proofs it is necessary to distinguish the two cases ($A = Y$ and $A \neq Y$) as explained in Section 3.2.3 or to generate an additional ndg ($A \neq Y$) as explained in Section 3.1.3.3.

3.2.7. *Integration in GeoProof*
Similarly to GCLC, the formalisation of the area method in *Coq* comes with a dynamic geometry software (Narboux, 2007a). The software developed, *GeoProof* (Figure 5) combines three tools: a dynamic geometry software to explore and invent conjectures, an automatic theorem prover to check facts, and an interactive proof system (*Coq*) to mechanically check proofs built interactively by the user.

3.3. Other Implementations of the Area Method

Although it is very well-know and widely credited as the most efficient method for proving geometry theorems that produce readable proofs, there are just a very few implementations of the area method. Actually, the situation is similar with other proving methods for geometry—to our knowledge, there are only around a dozen implementations in total of other most efficient proving methods (Wu's method, Gröbner bases method adapted to geometry theorem proving, the full angle method, and the deductive database method), counting versions employed within different systems. One of the main reasons for this is probably the fact that these methods, while having simple basic ideas, are all still very complex and require many details to be filled when making a real implementation.

In addition to the two implementations of the area method already described, we are aware of the other two: one used within a family of

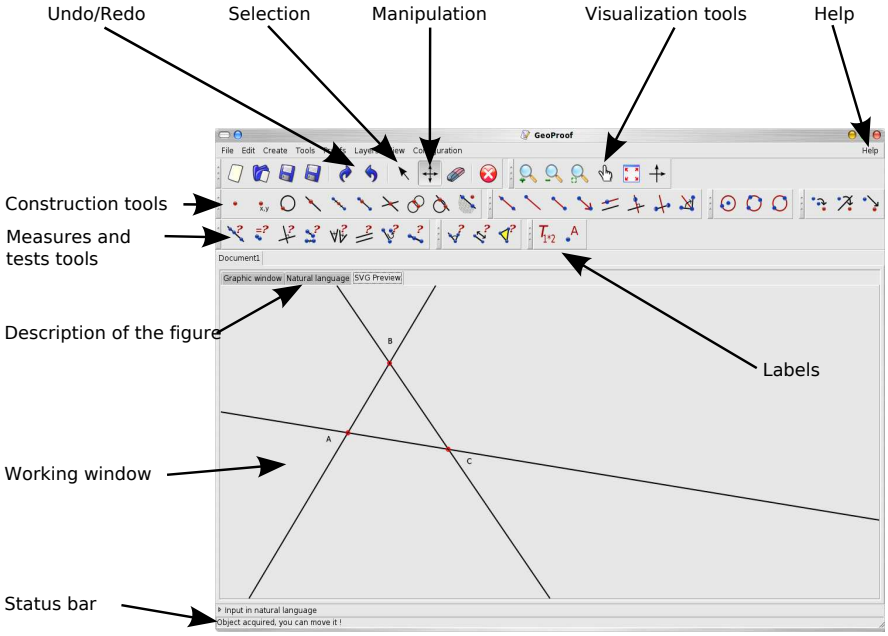


Figure 5. GeoProof

tools developed by the authors of the method and their collaborators, and one developed within the wider system *Theorema*.

3.3.1. *Euclid and Geometry Expert*

Euclid is theorem prover based on the area method, developed in 1993 by the authors of the method — Shang Ching Chou, Xiao Shan Gao, and Jing-Zhong Zhang (Chou et al., 1993). It was implemented in Common Lisp and was accompanied by a list of 400 proved theorems.

Geometry Expert[12] (*GEX*) is a dynamic geometry tool focused on automated theorem proving and it implements Wu's, Gröbner basis, vector, full-angle, and the area methods (Chou et al., 1996a). *GEX* was implemented in 1998 by Xiao Shan Gao.

MMP/Geometer[13] is a new, Chinese, version of *GEX*. The tool is being developed from 2002 by Xiao-Shan Gao and Qiang Lin. It automates geometry diagram generation, geometry theorem proving, and geometry theorem discovering (Gao and Lin, 2004). MMP/Geometer implements Wu's method, the area method, and the geometry deductive database method. Conjectures are given in a restricted pseudo-natural language or in a point-and-click manner.

[12] `http://www.mmrc.iss.ac.cn/gex/`
[13] `http://www.mmrc.iss.ac.cn/mmsoft/`

Java Geometry Expert[14] (*JGEX*) is a new, Java version of *GEX*. *JGEX* is being developed from 2004, by Shang Ching Chou, Xiao Shan Gao, and Zheng Ye. *JGEX* combines dynamic geometry, automated geometry theorem proving, and, as its most distinctive part, visual dynamic presentation of proofs. It provides a series of visual effects for presentation of proofs. The proofs can be visualised either manually or automatically. Within the program distribution, there are more than six hundred examples of proofs. *JGEX* implements the following methods for geometry theorem proving: Wu's method, the Groëbner basis method, the full-angle method, the deductive database method. In the version 0.80 (May 2009), the area method and the vector method are still under development.

The systems from the *GEX* family are publicly available, but they are not open-source and are not accompanied by technical reports with implementation details, so one cannot reconstruct how some parts of the proving methods are implemented. Available research papers describing these tools describe mainly only the high-level ideas and main required lemmas, but for instance, descriptions of the simplification phase and dealing with case splits are not available.

3.3.2. *Theorema*

Theorema[15] is a general mathematical tool with uniform framework for computing, problem solving, and theorem proving (Buchberger et al., 2006). *Theorema* is implemented in *Mathematica*. It has been developing from 1996 by Bruno Buchberger and a large team of his collaborators. *Theorema* has support for several methods for automated theorem proving, including methods for theorem proving in geometry. The geometry provers are designed for constructive geometry problems and there is support for Wu's method, Gröbner bases method, and the area method (Robu, 2002). These provers were implemented by Judit Robu (the algebraic methods rely on methods that were already available in *Mathematica* and *Theorema*).

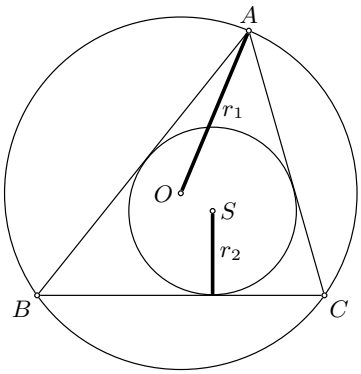
The geometry theorem provers are accompanied by visualisation tools typical for dynamic geometry. Numerical checks of the validity of geometry statements can also be performed for specific coordinates of the points.

In addition to the basic area method, there is also a modified version that can deal not only with conjectures in the form of equalities, but also with conjectures in the form of inequalities over geometric quantities. Within this method (*AreaCAD*), geometric expressions are transformed by the lemmas used in the basic area method and an con-

[14] `http://www.jgex.net/`
[15] `http://www.theorema.org/`

jecture (equivalent to the original one) only in terms of the free points of the construction is obtained. That new expression (with two sides linked by one of the relations $<$ or $>$) is tested for validity by Collins' algorithm for quantifier elimination in real closed fields by cylindrical algebraic decomposition (Collins, 1975).

EXAMPLE 3.3. *Let r_1 be the radius of the circumcircle of a triangle ABC, and let r_2 be the radius of the inscribed circle of the triangle. Then it holds that $r_1^2 \geq 4r_2^2$ and this can be proved by AreaCAD.*



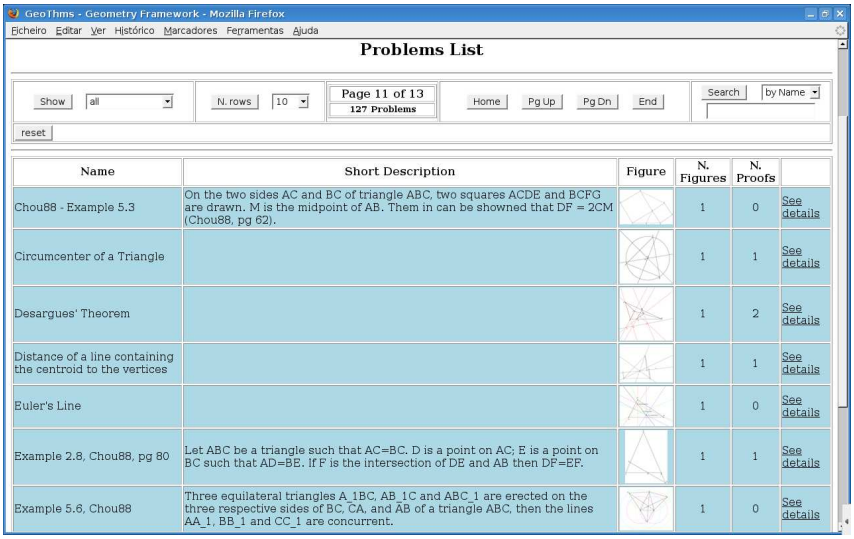
3.4. Applications

As other geometry theorem provers, the area method can have a range of different applications in education, mathematical software, computer-aided design, computer graphics, computer vision, robotics, etc. In this section a few existing, rather straightforward applications, of the method are described.

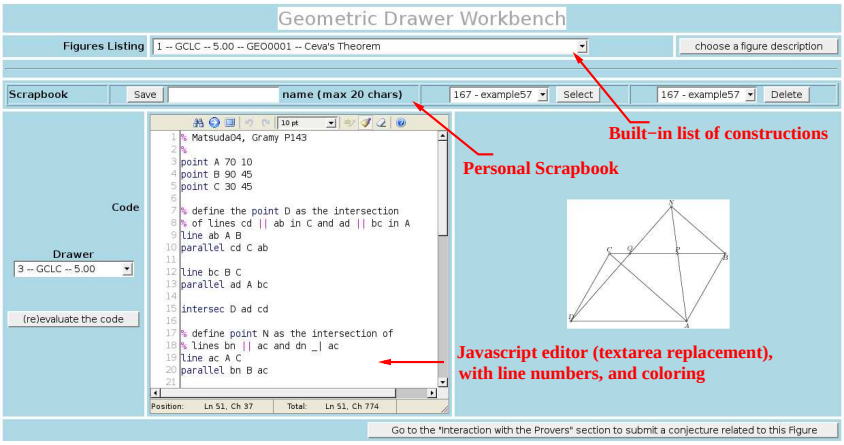
3.4.1. *GeoThms*

GeoThms is a web-based framework for exploring geometrical knowledge that integrates dynamic geometry software, automatic theorem provers, and a repository of geometric constructions, figures and proofs (Quaresma and Janičić, 2006b; Quaresma and Janičić, 2006a). The GeoThms users can easily use/browse through existing geometrical content and build new contents.

The main motivation is to build and maintain a publicly accessible and widely used Internet based framework for constructive geometry. It can be used for teaching and studying geometry, but also as a major Internet repository for geometrical knowledge.



The dynamic geometry software currently used within GeoThms are GCLC (Janičić, 2006) and Eukleides[16] (Quaresma and Pereira, 2006), two widely used dynamic geometry packages. The automated theorem provers used are the two theorem provers described in sections 3.1 and 3.2, both based on the area method, and two theorem provers based on algebraic methods (Predović, 2008). GeoThms provides a web workbench that tightly integrates the mentioned tools into a single framework for constructive geometry.



The Web interface is a PHP/MySQL server-side solution designed to enable GeoThms users easily browse through the list of geometry

[16] http://www.eukleides.org/

problems, their statements, illustrations and proofs, and also to interactively use the drawing and automatic proof tools. GeoThms is accessible at `http://hilbert.mat.uc.pt/GeoThms`.

3.4.2. *Automatic Verification of Regular Constructions*
Some geometry tools (e.g., *Eukleides*, GCLC) have a dual view of a given geometric construction — its description in a custom formal language and a visualised version, within the graphical interface. Other tools (e.g., *Geometer's Sketchpad*, *Cabri*) do not have, at least in an explicit form, a formal language for geometric constructions and instead the user does not describe a construction in abstract terms but "draws" it, using a pre-defined set of geometry operations. Generally, there are three types of construction errors:

- syntactic errors — only applicable for geometry tools with formal languages and this type of error is easily detected by the underlying processor and easily correctable by the user. For the other family of geometry tools this type of error doesn't occur due to a controlled environment where only syntactically correct actions are allowed.

- semantic errors —situations when, for a concrete set of geometrical objects (usually given in Cartesian plane), a construction step is not possible, for instance, two identical points do not determine a line. Such an error will be dealt by most (if not all) geometry tools for a given fixed set of points. However, that error is detected by an argument relevant only for the given instance of the construction and the question whether the construction step is always impossible or it is not possible only in the given special case is left open.

- deductive errors —when a construction step is geometrically unsound, e.g., there is never an intersection of two parallel lines in Euclidean geometry. The formal proof that a construction step is always inpossible can only be provided by geometry tools that incorporate geometry theorem provers.

GCLC has a built-in mechanism (using GCLCprover) for checking if a construction step is illegal, i.e., if it is always impossible (Janičić and Quaresma, 2007).

EXAMPLE 3.4. *Example 85 from the book* Mechanical Geometry Theorem Proving *(*Chou, 1987*) will be used to illustrate the mechanism for automatic verification of regular constructions built into GCLC. Using GCLC, the illustration given in Figure 6 can be generated.*

If the code for the intersection of lines AD and MN is added, e.g.,

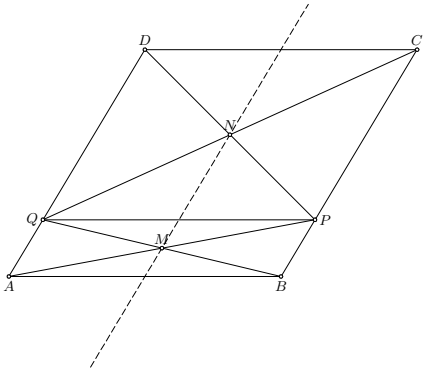


Figure 6. Example 85 from the book *Mechanical Geometry Theorem Proving*

```
line mn M N
intersec X mn ad
```

GCLC will not perform the last construction step and it will give the following error message:

```
Run-time error: Bad definition. Can not determine
intersection. (Line: 40, position: 10)
```

This is a semantic error only, detected for the concrete set of points in the Cartesian plane. However, if GCLC is called with an appropriate option, in the above situation (with a semantic error encountered), it will invoke the built-in theorem prover and provide the following information.

```
Deduction check invoked: the property that led to the error
will be tested for validity.

The conjecture successfully proved - the critical property
always holds. The prover output is written in the file
error-proof.tex.
```

Thus, the tool provides not only the statement that the construction is always illegal, but also a rigorous proof of it (in the area method style).

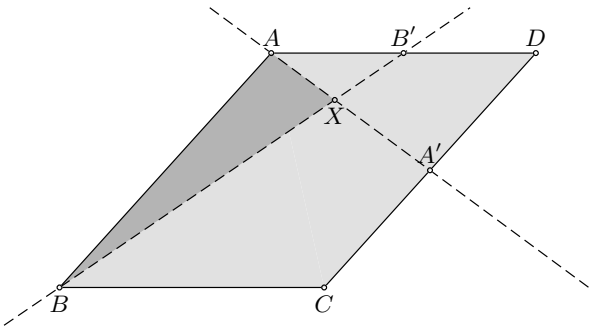
As far as we are aware of, the system for automated *deductive* testing whether a construction is illegal that is built into GCLC is the only such system. A similar mechanism is available in *JGEX*: when a user tries to perform an illegal construction step, the tool may report that it is not possible to perform the step, but it does not provide a proof for that argument. The geometry tool *Cinderella* does not allow performing illegal

construction steps, however the justification is not based on deductive but on probabilistic reasoning (Kortenkamp and Richter-Gebert, 2004).

The capability of performing deductive checks is important feature that enhances the didactic nature of dynamic geometry tools and provides an important link with automated theorem proving. That link connects the deductive nature of geometry conjectures with the semantic nature of models of geometry and, also, with human intuition and visualisations.

3.4.3. *Computing Geometric Expressions*
Within *Theorema*, the area method machinery is used for computing expressions involving geometric quantities relative to a given construction. For the given expression, all constructed points are eliminated and the expression is simplified, similarly as in the basic method (Robu, 2002).

EXAMPLE 3.5. *Let A, B and C be arbitrary points and let r is an arbitrary number. Let D be the intersection of the line through B that is parallel to AC and the line through C that is parallel to AB. Let A' be the point that divides CD in the ratio $1 : r(r-1)$ and let B' be the point that divides DA in the ratio $1 : r(r-1)$. Finally, let X be the intersection of the lines AA' and BB'. The goal is to find the ratio of the area of the triangle ABC and the quadrilateral $ABCD$.*



The tool implemented within Theorema, based on the area method can compute that the given ratio is equal to $\frac{1-r}{4-4r+2r^2}$.

Notice that the basic area method can prove *that the given ratio equals* $\frac{1-r}{4-4r+2r^2}$, *but* computing *the given ratio (without an expected result) requires some slight modifications of the method*[17].

[17] This extension of the method was originally described by the authors of the method (Chou et al., 1994).

3.4.4. *Discovering Geometry Properties*
Within *Theorema*, the area method machinery is used for exploring geometrical configurations and discovering geometry properties (Robu, 2002). The method is based on a systematic generation of all geometric expressions representing interesting properties relative to a construction (collinear points, congruent segments, parallel and perpendicular lines, triangles with the same area) and then analysing which of these properties might be unknown so far i.e., not present in an available knowledge base. Starting from a knowledge base that specifies some constructions and properties, a range of interesting theorems can be automatically obtained. These obtained theorems can be added to the knowledge base and the exploration may continue without recomputing the results already obtained. For testing generated properties, the area method is used, but other proving methods can be used as well.

4. Contributions

In this paper we gave a detailed account of the area method and described all existing implementation that we are aware of and their wider contexts. This account can serve as a basis for a straightforward implementation of the method. In addition to that, this paper brings the following original contributions:

- We gave an axiom system that serve as a basis for the method, an extension of the axiom system given by the authors of the method (Chou et al., 1994) (Section 2.2.2).
- We made formal proofs, within the proof assistant *Coq* (in a contribution accompanying this paper), of all the lemmas needed for the correction of the method not only for affine geometry (already described before (Narboux, 2004)), but also for Euclidean geometry (Narboux, 2009). Thanks to the formalisation, we ensured the correctness of all the lemmas required by the method, with an exception of one lemma that, as published in the original description (Chou et al., 1994), contained an error.
- We provided detailed traditional proofs in an Hilbert-style system (in a technical report accompanying this paper (Quaresma and Janičić, 2009)) of all the lemmas and filled-in some details missing in the original descriptions.
- We made explicit the elimination procedure for all cases including the special cases such as $\frac{\overline{AY}}{\overline{CY}}$ (Section 2.4.1).

- We made explicit dealing with the case split occurring in some of the lemmas (Section 2.5.1).
- We made explicit the uniformization phase which consists in finding normal forms for geometric quantities (Section 2.5.2).
- We made explicit the formulas to be used for dealing with free points (Section 2.5.3).
- We made an explicit description of the simplification phase (Section 2.5.4).
- We made explicit the algorithm for deciding equality between two rational expressions in independent parameters (Section 2.5.5).
- We highlighted the fact that a special case needs to be studied when eliminating Y in $\frac{\overline{AY}}{\overline{CD}}$ (Section 3.2.6).

5. Conclusions

In this paper we gave a detailed description of the area method, one of the most significant methods for automated theorem proving in geometry, introduced by Chou et al. in 1993. The method produces human-readable proofs and can efficiently prove many non-trivial theorems. The description of the method given here can serve as a detailed tutorial on the method (first of that kind), sufficient for understanding and implementing it in a straightforward manner.

Within this paper we also showed how the area method can be successfully integrated with other mathematical tools.

We, the authors of the paper, independently made two of these integrated implementations and in this paper we presented our combined results and experiences related to the method and its applications.

References

Bertot, Y. and P. Castéran: 2004, *Interactive Theorem Proving and Program Development, Coq'Art: The Calculus of Inductive Constructions*, Texts in Theoretical Computer Science. An EATCS Series. Springer.

Boutin, S.: 1997, 'Using reflection to build efficient and certified decision procedures.'. In: M. Abadi and T. Ito (eds.): *Proceedings of TACS'97*, Vol. 1281 of *Lecture Notes in Computer Science*.

Buchberger, B., A. Craciun, T. Jebelean, L. Kovacs, T. Kutsia, K. Nakagawa, F. Piroi, N. Popov, J. Robu, M. Rosenkranz, and W. Windsteiger: 2006, 'Theorema: Towards Computer-Aided Mathematical Theory Exploration'. *Journal of Applied Logic* **4**, 470–504.

Chou, S., X. Gao, and J. Zhang: 1996a, 'An Introduction to Geometry Expert'. In: *Proc. CADE-13*. pp. 235–239.

Chou, S.-C.: 1985, 'Proving and discovering geometry theorems using Wu's method'. Ph.D. thesis, The University of Texas, Austin.

Chou, S.-C.: 1987, *Mechanical Geometry Theorem Proving*. Dordrecht: D. Reidel Publishing Company.

Chou, S.-C., X.-S. Gao, and J.-Z. Zhang: 1993, 'Automated production of traditional proofs for constructive geometry theorems'. In: M. Vardi (ed.): *Proceedings of the Eighth Annual IEEE Symposium on Logic in Computer Science LICS*. pp. 48–56.

Chou, S.-C., X.-S. Gao, and J.-Z. Zhang: 1994, *Machine Proofs in Geometry*. Singapore: World Scientific.

Chou, S.-C., X.-S. Gao, and J.-Z. Zhang: 1995, 'Automated Production of Traditional Proofs in Solid Geometry'. *Journal of Automated Reasoning* **14**, 257–291.

Chou, S.-C., X.-S. Gao, and J.-Z. Zhang: 1996b, 'Automated Generation of Readable Proofs with Geometric Invariants, I. Multiple and Shortest Proof Generation'. *Journal of Automated Reasoning* **17**, 325–347.

Chou, S.-C., X.-S. Gao, and J.-Z. Zhang: 1996c, 'Automated Generation of Readable Proofs with Geometric Invariants, II. Theorem Proving With Full-Angles'. *Journal of Automated Reasoning* **17**, 349–370.

Coelho, H. and L. M. Pereira: 1986, 'Automated reasoning in geometry theorem proving with Prolog'. *Journal of Automated Reasoning* **2**(4), 329–390.

Collins, G. E.: 1975, 'Quantifier Elimination for Real Closed Fields by Cylindrical Algebraic Decomposition'. Vol. 33 of *Lecture Notes In Computer Science*. Springer-Verlag, pp. 134–183.

Dehlinger, C., J.-F. Dufourd, and P. Schreck: 2000, 'Higher-Order Intuitionistic Formalization and Proofs in Hilbert's Elementary Geometry'. In: *Proceedings of Automated Deduction in Geometry (ADG00)*. pp. 306–324.

Duprat, J.: 2008, 'Une axiomatique de la géométrie plane en Coq.'. In: *Actes des JFLA 2008*. pp. 123–136. In french.

Elcock, E. W.: 1977, 'Representation of knowledge in geometry machine'. *Machine Intelligence* **8**, 11–29.

Gao, X.-S. and Q. Lin: 2004, 'MMP/Geometer - A Software Package for Automated Geometric Reasoning'. In: F. Winkler (ed.): *Proceedings of Automated Deduction in Geometry (ADG02)*. pp. 44–66.

Gelernter, H.: 1959, 'Realization of a geometry theorem proving machine'. In: *Proceedings of the International Conference Information Processing*. Paris, pp. 273–282.

Geuvers, H. and et.al.: 2008, 'The "Fundamental Theorem of Algebra" Project'. `http://www.cs.ru.nl/~freek/fta/`.

Gonthier, G. and B. Werner: 2004, 'A computer checked proof of the four colour theorem.'.

Greeno, J., M. E. Magone, and S. Chaiklin: 1979, 'Theory of constructions and set in problem solving'. *Memory and Cognition* **7**(6), 445–461.

Guilhot, F.: 2004, 'Formalisation en Coq d'un cours de géométrie pour le lycée'. In: *Journées Francophones des Langages Applicatifs*.

Heath, T. L.: 1956, *The Thirteen Books of Euclid's Elements*. New-York: Dover Publications. 2nd ed.

Hilbert, D.: 1977, *Foundations of Geometry*. Open Court Publishing. 10th Revised edition. Editor: Paul Barnays.

Huet, G., G. Kahn, and C. Paulin-Mohring: 2004, 'The Coq Proof Assistant - A tutorial - Version 8.0'.

Janičić, P.: 2006, 'GCLC – A Tool for Constructive Euclidean Geometry and More than That'. In: N. Takayama, A. Iglesias, and J. Gutierrez (eds.): *Proceedings of International Congress of Mathematical Software (ICMS 2006)*.

Janičić, P. and P. Quaresma: 2007, 'Automatic Verification of Regular Constructions in Dynamic Geometry Systems'. In: F. Botana and T. Recio (eds.): *Proceedings of Automated Deduction in Geometry (ADG06)*, Vol. 4869 of *Lecture Notes in Artificial Intelligence*. Pontevedra, Spain, pp. 39–51.

Janičić, P.: 1996, 'One method for automathed theorem proving in geometry'. Master's thesis, Faculty of Mathematics, University of Belgrade. in Serbian.

Janičić, P.: 2009, 'Geometry Constructions Language'. *Journal of Automated Reasoning*. to appear.

Janičić, P. and P. Quaresma: 2006, 'System Description: GCLCprover + GeoThms'. In: F. Ulrich and S. Natarajan (eds.): *Automated Reasoning*, Vol. 4130 of *Lecture Notes in Artificial Intelligence*. pp. 145–150.

Kahn, G.: 1995, 'Constructive Geometry according to Jan von Plato'. Coq contribution. Coq V5.10.

Kapur, D.: 1986a, 'Geometry theorem proving using Hilbert's Nullstellensatz'. In: *SYMSAC '86: Proceedings of the fifth ACM symposium on Symbolic and algebraic computation*. New York, NY, USA, pp. 202–208.

Kapur, D.: 1986b, 'Using Gröbner bases to reason about geometry problems'. *Journal of Symbolic Computation* **2**(4), 399–408.

Kortenkamp, U. and J. Richter-Gebert: 2004, 'Using Automatic Theorem Proving To Improve The Usability Of Geometry Software'. In: *Workshop on Mathematical User Interfaces*.

Leroy, X.: 2006, 'Formal certification of a compiler back-end, or: programming a compiler with a proof assistant'. In: *33rd symposium Principles of Programming Languages*. pp. 42–54.

Li, H.: 2000, 'Clifford algebra approaches to mechanical geometry theorem proving'. In: X.-S. Gao and D. Wang (eds.): *Mathematics Mechanization and Applications*. San Diego, CA, pp. 205–299.

Magaud, N., J. Narboux, and P. Schreck: 2008, 'Formalizing Projective Plane Geometry in Coq'. In: *Proceedings of ADG'08*.

Magaud, N., J. Narboux, and P. Schreck: 2009, 'Formalizing Desargues' theorem in Coq using ranks'. In: S. Y. Shin and S. Ossowski (eds.): *SAC*. pp. 1110–1115.

Meikle, L. and J. Fleuriot: 2003, 'Formalizing Hilbert's Grundlagen in Isabelle/Isar'. In: *Theorem Proving in Higher Order Logics*. pp. 319–334.

Narboux, J.: 2004, 'A Decision Procedure for Geometry in Coq'. In: S. Konrad, B. Annett, and G. Ganesh (eds.): *Proceedings of TPHOLs'2004*, Vol. 3223 of *Lecture Notes in Computer Science*.

Narboux, J.: 2006, 'Formalisation et automatisation du raisonnement géométrique en Coq'. Ph.D. thesis, Université Paris Sud.

Narboux, J.: 2007a, 'A Graphical User Interface for Formal Proofs in Geometry'. *Journal of Automated Reasoning* **39**(2), 161–180.

Narboux, J.: 2007b, 'Mechanical Theorem Proving in Tarski's geometry'. In: *Proceedings of Automatic Deduction in Geometry 06*, Vol. 4869 of *Lecture Notes in Artificial Intelligence*. pp. 139–156.

Narboux, J.: 2009, 'Formalization of the Area Method'. Coq user contribution. `http://dpt-info.u-strasbg.fr/~narboux/area_method.html`.

Nevis, A.: 1975, 'Plane geometry theorem proving using forward chaining'. *Artificial Intelligence* **6**(1), 1–23.

Predović, G.: 2008, 'Automated geometry theorem proving based on Wu's and Buchberger's methods.'. Master's thesis, Faculty of Mathematics, University of Belgrade. supervisor: Predrag Janičić (in Serbian).

Quaresma, P. and P. Janičić: 2006a, 'Framework for Constructive Geometry (Based on the Area Method)'. Technical Report 2006/001, Centre for Informatics and Systems of the University of Coimbra.

Quaresma, P. and P. Janičić: 2006a, 'GeoThms - a Web System for Euclidean Constructive Geometry'. In: S. Autexier and C. Benzmüller (eds.): *UITP 2006*, Vol. 174. pp. 21–33.

Quaresma, P. and P. Janičić: 2006b, 'GeoThms - Geometry Framework'. Technical Report 2006/002, Centre for Informatics and Systems of the University of Coimbra.

Quaresma, P. and P. Janičić: 2006b, 'Integrating Dynamic Geometry Software, Deduction Systems, and Theorem Repositories'. In: J. M. Borwein and W. M. Farmer (eds.): *Mathematical Knowledge Management*, Vol. 4108 of *Lecture Notes in Artificial Intelligence*. pp. 280–294.

Quaresma, P. and P. Janičić: 2009, 'The Area Method - Properties and their Proofs'. Technical Report 2009/006, Centre for Informatics and Systems of the University of Coimbra.

Quaresma, P. and A. Pereira: 2006, 'Visualização de Construções Geométricas'. *Gazeta de Matemática* **151**.

Robu, J.: 2002, 'Geometry Theorem Proving in the Frame of the Theorema Project.'. Ph.D. thesis, Johannes Kepler Universität, Linz.

Tarski, A.: 1951, *A decision method for elementary algebra and geometry*. University of California Press.

Tarski, A.: 1959, 'What is Elementary Geometry?'. In: P. S. L. Henkin and A. Tarski (eds.): *The axiomatic Method, with special reference to Geometry and Physics*. Amsterdam, pp. 16–29.

The Coq development team: 2009, 'The Coq proof assistant reference manual, Version 8.2'. TypiCal Project.

von Plato, J.: 1995, 'The axioms of constructive geometry'. In: *Annals of Pure and Applied Logic*, Vol. 76. pp. 169–200.

von Plato, J.: 1997, 'Formalization of Hilbert's geometry of incidence and parallelism'. In: *Synthese*, Vol. 110. pp. 127–141.

Wang, D.: 1995, 'Reasoning about geometric problems using an elimination method'. In: J. Pfalzgraf and D. Wang (eds.): *Automated Practical Reasoning*. New York, pp. 147–185.

Wu, W.-T.: 1978, 'On the decision problem and the mechanization of theorem proving in elementary geometry'. Vol. 21. Scientia Sinica, pp. 157–179.

Yang, L., X. Gao, S. Chou, and Z. Zhang: 1998, 'Automated Proving and Discovering of Theorems in Non-Euclidean Geometries'. In: *Proceedings of Automated Deduction in Geometry (ADG98)*. Berlin, Heidelberg, pp. 171–188.

Zhang, J.-Z., S.-C. Chou, and X.-S. Gao: 1995, 'Automated production of traditional proofs for theorems in Euclidean geometry I. The Hilbert intersection point theorems'. *Annals of Mathematics and Artificial Intelligenze* **13**, 109–137.