Journal of Automated Reasoning manuscript No.
(will be inserted by the editor)

The Area Method

A Recapitulation

Predrag Janičić · Julien Narboux · Pedro Quaresma

Received: 2009/10/07 / Accepted:

Abstract The area method for Euclidean constructive geometry was proposed by Chou, Gao and Zhang in the early 1990's. The method can efficiently prove many non-trivial geometry theorems and is one of the most interesting and most successful methods for automated theorem proving in geometry. The method produces proofs that are often very concise and human-readable.

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 · formalisation

Mathematics Subject Classification (2000) 51A05 · 68T15

The first author is partially supported by a grant 144030 of the Ministry of Science of Serbia. The second author is partially supported by the ANR project Galapagos.

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

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 geometric problems to algebraic problems. The automation of the proving process along this line began with the quantifier elimination method of Tarski [59] and since then had many improvements [15]. The characteristic set method, also known as Wu's method [4,63], the elimination method [62], the Gröbner basis method [35,36], and the Clifford algebra approach [39] 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 [20], Nevis [48], Elcock [18], Greeno et al. [23], Coelho and Pereira [14], Chou, Gao, and Zhang [8].

In this paper we focus on the area method, an efficient coordinates-free method for a fragment of Euclidean geometry, developed by Chou, Gao, and Zhang [8,9,11] that is somewhere between the two above styles. This method enables one to implement provers capable of proving many complex geometry theorems. The method is sometimes credited (e.g., by its authors) to produce traditional, human-readable proofs. The generated proofs are indeed often concise, consisting of steps that are directly related to the geometrical contents involved and hence can be readable and easily understood by a mathematician. However, since the proofs are formulated in terms of arithmetic expressions, they can also significantly differ from traditional, Hilbert-style, synthetic proofs given in textbooks. Also, proofs may involve huge expressions, hardly readable, despite the fact their atomic expressions have clear and intuitive geometrical meaning.

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 goal equality of the conjecture collapses to an equality 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 conjecture is a theorem, otherwise it is not. 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 other implementations. These three implementations were made independently and in different contexts:

– within a tool for storing and exploring mathematical knowledge (Theorema [2]) — implemented by Judit Robu [58].
– within a generic proof assistant (Coq [61]) — implemented by Julien Narboux [43];
– within a dynamic geometry tool (GCLC [29]) — implemented by Predrag Janičić and Pedro Quaresma [33];

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 implementation 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 [30, 33, 43, 45, 52, 54].

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 [9]. 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 goals in the form of inequalities (of the form $L < R$ or $L \leq R$). In that case, the inequality 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 [10] and for hyperbolic plane geometry [64]. Substantially, the main idea of these variants is the same as in the basic method and this demonstrates 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:*

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

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 of 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 2.1).

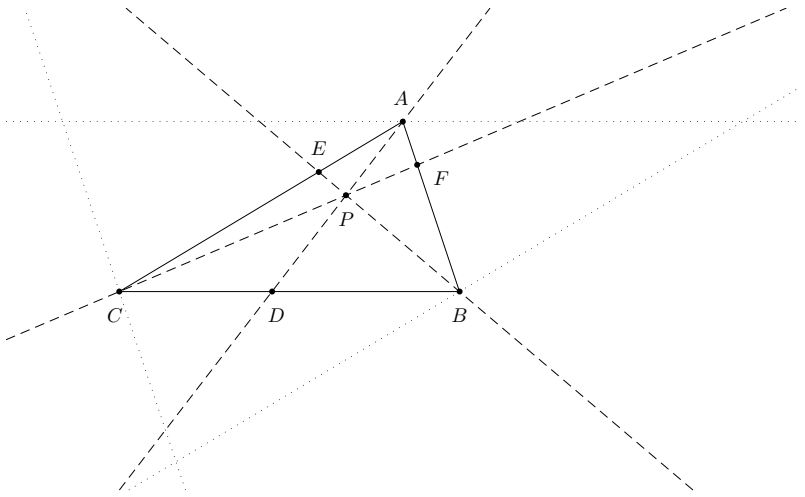


Fig. 2.1 Illustration for Ceva's theorem

*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 do they define? One can argue that the answer is one, but from a syntactic point of view, $\triangle ABC$ is not equal to $\triangle ACB$. 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 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 in the description of the area method [67].

For stating and proving conjectures, the area method uses a set of specific *geometric quantities* that enable treating arrangement relations. Some of them are:

- *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 $\mathcal{S}_{ABC}$ is the area of the triangle *ABC*, negated if *ABC* has the negative orientation.
- *Pythagoras difference*,[1] 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 purpose the properties of the geometric quantities, until an equality in only the free points is reached. If the equality is provable, then the original conjecture is a theorem as well. 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 [26], partly naive from today's point of view, was used for centuries. In the early twentieth century, Hilbert provided a more rigorous axiomatisation [27], one of the landmarks for modern mathematics, but still not up to modern standards [16, 42]. In the mid-twentieth century, Tarski presented a new axiomatisation for elementary geometry (with a limited support for

[1] 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 $\mathcal{P}_{ABC} = 0$).

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	$\mathcal{P}_{ABA} \neq 0 \wedge \mathcal{P}_{CDC} \neq 0 \wedge \mathcal{P}_{ACD} = \mathcal{P}_{BCD}$
AB is parallel to CD	$\mathcal{P}_{ABA} \neq 0 \wedge \mathcal{P}_{CDC} \neq 0 \wedge \mathcal{S}_{ACD} = \mathcal{S}_{BCD}$
O is the midpoint of AB	$\mathcal{S}_{ABO} = 0 \wedge \mathcal{P}_{ABA} \neq 0 \wedge \frac{\overline{AO}}{\overline{AB}} = \frac{1}{2}$
AB has the same length as CD	$\mathcal{P}_{ABA} = \mathcal{P}_{CDC}$
points A, B, C, D are harmonic	$\mathcal{S}_{ABC} = 0 \wedge \mathcal{S}_{ABD} = 0 \wedge \mathcal{P}_{BCB} \neq 0 \wedge \mathcal{P}_{BDB} \neq 0 \wedge \frac{\overline{AC}}{\overline{CB}} = \frac{\overline{DA}}{\overline{DB}}$
angle ABC has the same measure as DEF	$\mathcal{P}_{ABA} \neq 0 \wedge \mathcal{P}_{ACA} \neq 0 \wedge \mathcal{P}_{BCB} \neq 0 \wedge \mathcal{P}_{DED} \neq 0 \wedge \mathcal{P}_{DFD} \neq 0 \wedge$ $\mathcal{P}_{EFE} \neq 0 \wedge \mathcal{S}_{ABC} \cdot \mathcal{P}_{DEF} = \mathcal{S}_{DEF} \cdot \mathcal{P}_{ABC}$
A and B belong to the same circle arc CD	$\mathcal{S}_{ACD} \neq 0 \wedge \mathcal{S}_{BCD} \neq 0 \wedge \mathcal{S}_{CAD} \cdot \mathcal{P}_{CBD} = \mathcal{S}_{CBD} \cdot \mathcal{P}_{CAD}$

Table 2.1 Expressing geometry predicates in terms of the three geometric quantities.

continuity features), along with a decision procedure for that theory [60]. Although there are other variations of these systems [31,44], 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[2]). 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.

2.2.1 A Hilbert Style Axiomatisation

The geometric quantities used within the area method (mentioned in Section 2.1) 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).

Using geometric quantities, it is possible to express a range of geometry predicates as shown in Table 2.1.

The given correspondences can be proved as theorems of Hilbert's geometry. For instance, one direction of the property about angle congruence can be proved as follows. Since A, B, and C define an angle, they are different by definition (i.e., $\mathcal{P}_{ABA} \neq 0$, $\mathcal{P}_{ACA} \neq 0$, $\mathcal{P}_{BCB} \neq 0$), and the same holds for the points D, E, F. If the angle ABC is a right angle, then $\mathcal{P}_{ABC} = \mathcal{P}_{DEF} = 0$ and trivially $\mathcal{S}_{ABC} \cdot \mathcal{P}_{DEF} = \mathcal{S}_{DEF} \cdot \mathcal{P}_{ABC}$; otherwise, by the cosine rule, $\mathcal{S}_{ABC}/\mathcal{P}_{ABC} = (\frac{1}{2}\overline{AB} \cdot \overline{BC} \cdot \sin(ABC))/(\overline{AB}^2 + \overline{CB}^2 - (\overline{AB}^2 + \overline{CB}^2 - 2\overline{AB} \cdot \overline{BC}\cos(ABC))) = \sin(ABC)/(4\cos(ABC)) = \tan(ABC)/4$; hence, if the angle DEF is congruent to ABC, then $\mathcal{S}_{ABC}/\mathcal{P}_{ABC} = \tan(ABC)/4 = \mathcal{S}_{DEF}/\mathcal{P}_{DEF}$ and, further $\mathcal{S}_{ABC} \cdot \mathcal{P}_{DEF} = \mathcal{S}_{DEF} \cdot \mathcal{P}_{ABC}$.

Proofs generated by the area method use a set of specific lemmas (see Section 2.4). These lemmas can be proved within Hilbert's geometry (i.e., within its fragment for plane geometry), but the full, formal proofs would be very long and would involve complex notions like orientation and area of a triangle. That is why it is suitable to have an alternative,

[2] See von Plato's discussion about incidence in Hilbert's geometry [50].

higher-level axiomatisation, suitable for the area method. Chou, Gao and Zhang [8] 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 [8,9] is 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.

In contrast to Hilbert's system, in our axiom system there is just one primitive type of geometrical objects: points. Variables can also range over a field $(F,+,\cdot,0,1)$. F is any field of characteristic different from 2.[3] 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 2.2 are implicitly universally quantified. To improve readability (of the last three axioms), the following shorthands are used:

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

The following shorthands are also used within the method for better readability:

$$\begin{aligned} \mathcal{S}_{ABCD} &\equiv \mathcal{S}_{ABC} + \mathcal{S}_{ACD} \\ \mathcal{P}_{ABCD} &\equiv \mathcal{P}_{ABD} - \mathcal{P}_{CBD} \end{aligned}$$

Definition 2.1 (Geometry Quantities) Geometry quantities *are expressions of the form* $\frac{\overline{AB}}{\overline{CD}}$, $\mathcal{S}_{ABC}$, $\mathcal{S}_{ABCD}$, $\mathcal{P}_{ABC}$, $\mathcal{P}_{ABCD}$.

Relationship with the Hilbert style geometry. Note that in the Hilbert style approach, predicates $\overline{\cdot\cdot}$, $\mathcal{S}_{\cdot\cdot\cdot}$, and $\mathcal{P}_{\cdot\cdot\cdot}$ and are all defined (see Section 2.2.1), while in this approach, $\overline{\cdot\cdot}$, $\mathcal{S}_{\cdot\cdot\cdot}$ are primitive predicates and $\mathcal{P}_{\cdot\cdot\cdot}$ is a defined predicate. In both cases, ratio of parallel directed segments is defined using the notions of the theory of fields. Provable properties of Hilbert's geometry shown in Table 2.1, can be used as definitions (for notions of parallel lines, perpendicular lines, etc) in the area method theory. Thanks to all these definitions, all well-formed formulae of the theory of the area method are also well-formed formulae of the Hilbert style geometry. Moreover, all presented axioms of the area method can be proved in

[3] 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 $0 \neq 2$.

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 $A \neq B, \mathcal{S}_{ABP} = 0, \overline{AP} = r\overline{AB}, \mathcal{S}_{ABP'} = 0$ and $\overline{AP'} = r\overline{AB}$, then $P = P'$ (unicity)
9. If $PQ \parallel CD$ and $\frac{\overline{PQ}}{\overline{CD}} = 1$ then $DQ \parallel PC$ (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)

Table 2.2 The axiom system

the Hilbert style geometry as theorems.[4] Because of that, each conjecture that can be proved by the axioms for the area method, is also a theorem of Hilbert's geometry (assuming the same inference system).

Relationship with the axiom system of Chou, Gao, and Zhang. Our axiom system is an extended and modified version of the original system by Chou, Gao, and Zhang. While their axiom system deals with affine geometry only (and does not introduce the notion of Pythagoras difference), our system contains axioms about Pythagoras difference (axioms 11, 12, and 13) and, thanks to that, deals with Euclidean geometry. Compared to the original version, ours has also the advantage of being more precise and organised. The axiom system we propose differs from the axiom system of Chou, Gao and Zhang in the following ways too:

1. Our system does not use collinearity as a primitive notion and instead, collinearity is defined by the signed area. Chou, Gao and 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 $\mathcal{S}_{ABC} = 0$ [9].
2. 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).

Finally, using our axiom system — more suitable for that task — we *formally verified* (within the *Coq* proof assistant [61]) all the properties of the geometric quantities required

[4] We don't have formal proofs for these conjectures as they would involve formalisation of very complex notions like orientation and area of a triangle, which is still beyond reach for current formalisation of Hilbert's geometry.

by the area method, demonstrating the correctness of the system and eliminating all concerns about provability of the lemmas [47].

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 exists).
- construction of the intersections of two given circles (if such points exists).

The area method cannot deal with all geometry theorems involving the above constructions. It does not support construction of an arbitrary line, and it supports 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 construction steps supported by the area method are expressed in terms of the involved points.[5] Therefore, only lines and circles determined by specific points can be used (rather than arbitrarily chosen lines and circles) and the key construction steps are those introducing new points. For a construction step to be well-defined, certain conditions may be required. These conditions are called *non-degeneracy conditions* (ndg-conditions).

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

Given below is the list of elementary construction steps in the area method, along with the corresponding ndg-conditions. Free points are introduced only by ECS1 and, if r is a variable, by ECS4 and by ECS5.

[5] Elementary construction steps used by the area method do not use the concept of line and plane explicitly. This is convenient from the point of view of formalisation and automation. Indeed, in an axiom system based only on the concept of points (as in Tarski's axiom system [60]), the dimension implied can be easily changed by adding or removing some appropriate axioms (stated in the original 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 extend 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.

ECS1 construction of an arbitrary point U; this construction step is denoted by (POINT U).
ndg-condition: –

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 U V P Q).
ndg-condition: $UV \nparallel PQ$; $U \neq V$; $P \neq Q$.
A formula that corresponds to this construction step is: $U \neq V \wedge P \neq Q \wedge UV \nparallel PQ \wedge \mathcal{S}_{UVY} = 0 \wedge \mathcal{S}_{PQY} = 0$.

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 U V).
ndg-condition: $U \neq V$
A formula that corresponds to this construction step is: $U \neq V \wedge PY \perp UV \wedge \mathcal{S}_{UVY} = 0$.

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 U V r).
ndg-condition: $U \neq V$; if r is a rational expression in the geometric quantities, the denominator of r should not be zero.
A formula that corresponds to this construction step is: $U \neq V \wedge WY \parallel UV \wedge \frac{\overline{WY}}{\overline{UV}} = r$.

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 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.
A formula that corresponds to this construction step is: $U \neq V \wedge UY \perp UV \wedge \frac{4\mathcal{S}_{UVY}}{\mathcal{P}_{UVU}} = r$.

The above set of construction steps 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 U V), (PRATIO Y N 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 $O1$ $O2$), (PRATIO Y N 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); such line is determined by the points W and N, where N is obtained by (PRATIO N W U V 1).
- 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 such line is determined by the points W

and N, where N is obtained by (TRATIO N W U 1), otherwise, such line is determined by the points W and N, where N is obtained by (FOOT N W U V).
– construction of a perpendicular bisector of a segment with endpoints U and V; such line is determined by the points N and M, where these points are obtained by (PRATIO M U U V 1/2), (TRATIO N M U 1).

Also, it is possible to construct an arbitrary point Y on a line (LINE U V), by (PRATIO Y U U V r) where r is an indeterminate, or on a circle (CIRCLE O P), by (POINT Q), (FOOT N O P Q), (PRATIO Y N N P -1). There can be also used some additional construction steps (with corresponding elimination lemmas) that can help producing shorted proofs in some cases [8] but we will not describe them here.

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 hand, 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 A P B C)	*(ECS2)*
(INTER E B P A C)	*(ECS2)*
(INTER F C P A B)	*(ECS2)*

2.3.2 Constructive Geometry Statements

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

Definition 2.2 (Constructive Geometry Statement) *A* constructive geometry statement, *is a list $S = (C_1, C_2, \ldots, C_m, G)$ where C_i, for $1 \leq i \leq m$, 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 **C**.

Note that, in its basic form, the area method does not deal with conclusion statements in the form of inequalities (for another variants of the method see Section 2.5.8 and Section 3.3.2).

For a statement $S = (C_1, C_2, \ldots, C_m, (E_1 = E_2))$ from **C**, the ndg-condition is the set of the ndg-conditions of the steps C_i, plus the conditions d_i that the denominators appearing in E_1 and E_2 are not equal to zero, and the conditions p_i that lines appearing in ratios of segments in E_1 and E_2 are parallel: for each ratio of the form $\frac{\overline{AB}}{\overline{CD}}$ appearing in E_1 and E_2, there is a ndg-condition $AB \parallel CD$. The logical meaning of a statement is hence:

$$\begin{array}{c} c_1 \wedge c_2 \wedge \ldots \wedge c_m \wedge \\ d_1 \wedge \ldots \wedge d_m \wedge \\ p_1 \wedge \ldots \wedge p_m \wedge \\ \Rightarrow E_1 = E_2 \end{array}$$

where c_i are the formulae characterising the construction steps (including their ndg-conditions). The formula above is assumed to be universally quantified.

The area method (as described in this paper) decides whether or not a conjecture of the above form is a theorem, i.e., whether it can be derived from the axiom system described in Section 2.2.2. If a conjecture is a theorem in the theory of the area method, then it is also a theorem of the Hilbert style geometry (as discussed in Section 2.2.2). Note that the area method is applied for statements of the form $H \Rightarrow E_1 = E_2$, while definitions of some geometry properties may involve inequalities as well, for instance, we say that *AB is parallel to CD* if $\mathcal{P}_{ABA} \neq 0 \wedge \mathcal{P}_{CDC} \neq 0 \wedge \mathcal{S}_{ACD} = \mathcal{S}_{BCD}$. Typically, when proving properties defined in Table 2.1, instead of proving $\mathcal{P}_{ABA} \neq 0 \wedge \mathcal{P}_{CDC} \neq 0 \wedge \mathcal{S}_{ACD} = \mathcal{S}_{BCD}$, the method is applied only for proving $\mathcal{S}_{ACD} = \mathcal{S}_{BCD}$, which gives a weaker conjecture (for the special cases of $A = B$ and $C = D$). Adding $A \neq B$ and $C \neq D$ to the set of ndg-conditions, would ensure that these two goals are equivalent.

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

$$\begin{array}{l} A \neq P \wedge B \neq C \wedge AP \nparallel BC \wedge \mathcal{S}_{APD} = 0 \wedge \mathcal{S}_{BCD} = 0 \wedge \\ B \neq P \wedge A \neq C \wedge BP \nparallel AC \wedge \mathcal{S}_{BPE} = 0 \wedge \mathcal{S}_{ACE} = 0 \wedge \\ C \neq P \wedge A \neq B \wedge CP \nparallel AB \wedge \mathcal{S}_{CPF} = 0 \wedge \mathcal{S}_{ABF} = 0 \wedge \\ F \neq B \wedge D \neq C \wedge E \neq A \wedge \\ AF \parallel FB \wedge BD \parallel DC \wedge CE \parallel EA \\ \Rightarrow \frac{\overline{AF}}{\overline{FB}} \frac{\overline{BD}}{\overline{DC}} \frac{\overline{CE}}{\overline{EA}} = 1 \end{array}$$

2.4 Properties of Geometric Quantities and 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 [8,9,11,67], but in a reorganised form. The rigorous traditional proofs (not formal) in the Hilbert's style geometry, accompanying all the results presented in this section are available [56]. The formal (machine verifiable) proofs are available as a *Coq* contribution [47].

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 2.1 $\frac{\overline{PQ}}{\overline{AB}} = -\frac{\overline{QP}}{\overline{AB}} = \frac{\overline{QP}}{\overline{BA}} = -\frac{\overline{PQ}}{\overline{BA}}$.

Lemma 2.2 $\frac{\overline{PQ}}{\overline{AB}} = 0$ *iff* $P = Q$.

Lemma 2.3 $\frac{\overline{PQ}}{\overline{AB}} \frac{\overline{AB}}{\overline{PQ}} = 1$.

Lemma 2.4 $\mathcal{S}_{ABC} = \mathcal{S}_{CAB} = \mathcal{S}_{BCA} = -\mathcal{S}_{ACB} = -\mathcal{S}_{BAC} = -\mathcal{S}_{CBA}$.

Lemma 2.5 $\mathcal{P}_{AAB} = 0$.

Lemma 2.6 $\mathcal{P}_{ABC} = \mathcal{P}_{CBA}$.

Lemma 2.7 $\mathcal{P}_{ABA} = 2\overline{AB}^2$.

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 segments 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 2.8 *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, using the shorthands defined in section 2.2.2: $\mathcal{S}_{ABCD} \equiv \mathcal{S}_{ABC} + \mathcal{S}_{ACD}, \mathcal{P}_{ABCD} \equiv \mathcal{P}_{ABD} - \mathcal{P}_{CBD}$.

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 2.2)
- $\frac{\overline{YY}}{\overline{YA}} = 0$ (by Lemma 2.2)
- $\frac{\overline{YY}}{\overline{CD}} = 0$ (by Lemma 2.2)
- $\frac{\overline{AY}}{\overline{BY}}$
- $\frac{\overline{AY}}{\overline{YB}} = -\frac{\overline{AY}}{\overline{BY}}$ (by Lemma 2.1)
- $\frac{\overline{YA}}{\overline{BY}} = -\frac{\overline{AY}}{\overline{BY}}$ (by Lemma 2.1)
- $\frac{\overline{YA}}{\overline{YB}} = \frac{\overline{AY}}{\overline{BY}}$ (by Lemma 2.1)
- $\frac{\overline{AY}}{\overline{CD}}$
- $\frac{\overline{YA}}{\overline{CD}} = -\frac{\overline{AY}}{\overline{CD}}$ (by Lemma 2.1)
- $\frac{\overline{AB}}{\overline{CY}} = \frac{1}{\frac{\overline{CY}}{\overline{AB}}}$ (by lemmas 2.1 and 2.3)
- $\frac{\overline{AB}}{\overline{YC}} = \frac{1}{\frac{\overline{CY}}{\overline{BA}}}$ (by lemmas 2.1 and 2.3)

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

- $\mathcal{S}_{YYY} = 0$ (by Lemma 2.4)
- $\mathcal{S}_{AYY} = 0$ (by Lemma 2.4)
- $\mathcal{S}_{YAY} = 0$ (by Lemma 2.4)
- $\mathcal{S}_{YYA} = 0$ (by Lemma 2.4)
- $\mathcal{S}_{AYB} = \mathcal{S}_{BAY}$ (by Lemma 2.4)
- $\mathcal{S}_{YAB} = \mathcal{S}_{ABY}$ (by Lemma 2.4)
- $\mathcal{S}_{ABY}$

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

- $\mathcal{P}_{YYY} = 0$ (by Lemma 2.5)
- $\mathcal{P}_{AYY} = 0$ (by lemmas 2.6 and 2.5)
- $\mathcal{P}_{YAY} = \mathcal{P}_{AYA}$ (by Lemma 2.7)
- $\mathcal{P}_{YYA} = 0$ (by Lemma 2.5)
- $\mathcal{P}_{AYB}$
- $\mathcal{P}_{YAB} = \mathcal{P}_{BAY}$ (by Lemma 2.6)
- $\mathcal{P}_{ABY}$

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 elimination of Y from geometric quantities. Thanks to Lemma 2.8, 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 provable 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 2.9 (EL1) *If Y is introduced by* (INTER Y U V P Q) *then (we assume that $A \neq Y$):*[6]

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

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

Lemma 2.10 (EL2) *If Y is introduced by* (FOOT Y P U V) *then (we assume that $A \neq Y$):*

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

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

[6] 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 2.11 (EL3) *If Y is introduced by* (PRATIO Y R P Q r) *then (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 2.12 (EL4) *If Y is introduced by* (TRATIO Y P Q r) *then (we assume that $A \neq Y$):*

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

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

Lemma 2.13 (EL5) *Let $G(Y)$ be a linear geometric quantity and Y is introduced by* (INTER Y U V P Q)*. Then:*

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

Lemma 2.14 (EL6) *Let $G(Y)$ be a linear geometric quantity and Y is introduced by* (FOOT Y P U V)*. Then:*

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

Lemma 2.15 (EL7) *Let $G(Y)$ be a linear geometric quantity and Y is introduced by* (PRATIO Y W U V r)*. Then:*

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

Lemma 2.16 (EL8) *If Y is introduced by* (TRATIO Y P Q r) *then:*

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

Lemma 2.17 (EL9) *If Y is introduced by* (TRATIO Y P Q r) *then:*

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

Lemma 2.18 (EL10) *Let $G(Y)$ be a linear geometric quantity and Y is introduced by* (INTER Y U V 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 2.19 (EL11) *Let $G(Y)$ be a linear geometric quantity and Y is introduced by* (FOOT Y P U V) *then:*

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

		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			

Table 2.3 Elimination Lemmas

Lemma 2.20 (EL12) *If Y is introduced by* (PRATIO Y W U V r) *then:*

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

Lemma 2.21 (EL13) *If Y is introduced by* (TRATIO Y P Q r) *then:*

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

The information on the elimination lemmas is summarised in Table 2.3.

On the basis 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 2.8, 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.

Namely, if the area method is applied to a conjecture with a goal of the form $E_1 \neq E_2$ and if it ends up with an inequality that is a trivial theorem (e.g., $0 \neq 1$), then the original statement is a theorem.

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 (see Section 2.5.8 for this variation of the method). 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 could be weaker than the original, given statement as the method may *introduce* additional ndg-conditions. Moreover, ndg-conditions that the method may introduce could be unnecessary, and the resulting statement could be less general than necessary.

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 introducing 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 cases, 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 names are in alphabetic order.

The uniformization procedure consists of applying exhaustively the following rules:

$$\overline{BA} \to -\overline{AB} \qquad \text{by Lemma 2.1}$$

$$\begin{array}{llll} \mathcal{S}_{BCA} \to \mathcal{S}_{ABC} & \mathcal{S}_{ACB} \to -\mathcal{S}_{ABC} & & \\ \mathcal{S}_{CAB} \to \mathcal{S}_{ABC} & \mathcal{S}_{BAC} \to -\mathcal{S}_{ABC} & \text{by Lemma 2.4} \\ \mathcal{S}_{CBA} \to -\mathcal{S}_{ABC} & & \end{array}$$

$$\begin{array}{ll} \mathcal{P}_{CBA} \to \mathcal{P}_{ABC} & \text{by Lemma 2.6} \\ \mathcal{P}_{BAB} \to \mathcal{P}_{ABA} & \text{by Lemma 2.7} \end{array}$$

2.5.3 Simplification

For simplification of the statement the following rewrite rules are applied.

Degenerated geometric quantities:

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

Ring simplifications:

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

$c_1+c_2\to c_3$ where c_1 and c_2 are constants (elements of F) and $c_1+c_2=c_3$
$c_1\cdot c_2\to c_3$, where c_1 and c_2 are constants (elements of F) and $c_1\cdot c_2=c_3$
Field simplifications (if $a\neq 0$):

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

2.5.4 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 ratios of directed segments, signed areas and Pythagoras differences in only *free points*, numerical constants and numerical variables. Most often, this simply leads to equalities that are trivially provable (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, by Axiom 6:

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

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 2.3 *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 an 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 $OU\perp OV$ and $d=\overline{OU}=\overline{OV}$, are introduced (for some d from F). 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 denote $\mathcal{S}_{OUP}$, let Y_P denote $\mathcal{S}_{OVP}$, and let $Col(A,B,C)$ denote the fact that A, B and C are collinear.

Lemma 2.22 *For any points A, B, C and D such that $C \neq D$ and $AB \parallel CD$:*

$$\frac{\overline{AB}}{\overline{CD}} = \begin{cases} \frac{X_C Y_A - X_C Y_B - Y_A X_B + Y_B X_A - Y_C X_A + Y_C X_B}{X_C Y_A - X_C Y_D - Y_A X_D - Y_C X_A + Y_C X_D + X_A Y_D} & \text{if not } Col(A,C,D) \\ \frac{X_B Y_A - X_A Y_B}{X_D Y_C - X_C Y_D} & \text{if } Col(A,C,D) \text{ and not } Col(O,A,C) \\ \frac{\mathcal{S}_{OUV}(X_B - X_A) + X_B Y_A - X_A Y_B}{\mathcal{S}_{OUV}(X_D - X_C) + X_D Y_C - X_C Y_D} & \text{if } Col(A,C,D) \text{ and } Col(O,A,C) \text{ and not } Col(U,A,C) \\ \frac{\mathcal{S}_{OUV}(Y_B - Y_A) + X_B Y_A - Y_B X_A}{\mathcal{S}_{OUV}(Y_D - Y_C) + X_D Y_C - Y_D X_C} & \text{otherwise} \end{cases}$$

Lemma 2.23 *For any points A, B and C:*
$\mathcal{S}_{ABC} = \frac{(Y_B - Y_C)X_A + (Y_C - Y_A)X_B + (Y_A - Y_B)X_C}{\mathcal{S}_{OUV}}$.

Lemma 2.24 *For any points A, B and C:*
$\mathcal{P}_{ABC} = 8(\frac{Y_A Y_C - Y_A Y_B + Y_B^2 - Y_B Y_C - X_A X_B + X_A X_C + X_B^2 - X_B X_C}{d^2})$.

Lemma 2.25 $S_{OUV} = \pm\frac{d^2}{2}$.

Using lemmas 2.22 to 2.25, 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.

2.5.5 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 such an equality (by transforming its two sides), the following (terminating) rewrite rules are used.

Reducing to a single fraction:

$$\begin{array}{lll} \frac{a}{b} + c \to \frac{a + c \cdot b}{b} & a \cdot \frac{b}{c} \to \frac{a \cdot b}{c} & \frac{a}{\frac{b}{c}} \to \frac{a \cdot c}{b} \\ c + \frac{a}{b} \to \frac{c \cdot b + a}{b} & \frac{a}{b} \cdot c \to \frac{a \cdot c}{b} & \frac{\frac{a}{b}}{c} \to \frac{a}{b \cdot c} \\ \frac{a}{b} + \frac{c}{b} \to \frac{a + c}{b} & \frac{a}{b} \cdot \frac{c}{d} \to \frac{a \cdot c}{b \cdot d} & \frac{\frac{a}{b}}{\frac{c}{d}} \to \frac{a \cdot d}{c \cdot b} \\ \frac{a}{b} + \frac{c}{d} \to \frac{a \cdot d + c \cdot b}{bd} & & \end{array}$$

Reducing to an equation without fractions:

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

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

$$a = c \to a - c = 0$$

Reducing left hand side to right associative form:

$$((a+b)+c) \to a+(b+c) \qquad a\cdot(b+c) \to a\cdot b+a\cdot c$$
$$((a\cdot b)\cdot c) \to a\cdot(b\cdot c) \qquad (b+c)\cdot a \to b\cdot a+c\cdot a$$

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

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

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

$E_1+\cdots+E_{i-1}+c_1\cdot C+E_{i+1}+\cdots+E_{j-1}+c_2\cdot C'+E_{j+1}+\cdots+E_n \to E_1+\cdots E_{i-1}+c_3\cdot C+E_{i+1}+\cdots+E_{j-1}+E_{j+1}+\cdots+E_n$, 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 provable 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 Section 3.2), the second in GCLCprover (see Section 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 equivalence preserving.

2.5.6 Non-degeneracy Conditions

Some construction steps 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 construction steps, 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, $\mathcal{S}_{PUV} \neq \mathcal{S}_{QUV}$.

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 goal equality are not equal to zero, and the conditions that $AB \parallel CD$ for every ratio $\frac{\overline{AB}}{\overline{CD}}$ that appear in the goal equality. As said in Section 2.3.2, it is proved that the goal equality follows from the construction specification and the ndg-conditions. Hence, if the negation of some ndg-condition of a geometry statement is met (i.e., if it is implied by the preceding construction steps), the left-hand side of the implication is inconsistent and the statement is trivially a theorem (so there is no need for activating the mechanism for transforming the goal equality). Negations of these ndg-conditions are checked during the proving process. As seen from the above forms, these negations can

be expressed as equalities in terms of geometric quantities and can be checked by the area method itself.

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 a theorem.

Additional ndg-conditions (additional with respect to the original statement) may be introduced during the proving process in the non-branching approach (see Section 2.5.1) to ensure that the elimination lemmas with side-conditions can be applied.

Ndg-conditions from definitions given in Table 2.1, are never a part of the assumptions of a statement, since the assumptions are built from the construction steps and the goal equality. They can be used only as goal equalities (or goal inequalities — see Section 2.5.8), when proving some of the properties defined as in Table 2.1, to ensure a full compliance with the Hilbert style geometry for degenerative cases.

2.5.7 The Algorithm

The area method checks whether a constructive geometry statement $(C_1, C_2, \ldots, C_m, E_1 = E_2)$ is a theorem 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 a theorem 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 2.1 in Section 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 a theorem; otherwise, this ndg-condition is one of the assumptions of the statement.
   (b) simplify the current goal (by using the simplification procedure, described in 2.5.3);
   (c) if the current construction step introduces a new point P, then eliminate (by using Lemma 2.8 and the elimination lemmas) all occurrences of P from the current goal;
3. uniformize the geometric quantities (using the uniformization rules, described in 2.5.2);
4. simplify the current goal (by using the simplification procedure, described in 2.5.3);
5. test if the obtained equality is provable (by using the procedure given in 2.5.5); if yes, then the conjecture $E_1 = E_2$ is provable, under the assumption that the ndg-conditions hold, otherwise:
   (a) eliminate the free points (using the area coordinates, as described in 2.5.4);

(b) simplify the current goal (by using the simplification procedure, described in 2.5.3);
(c) test if the obtained equality is provable (by using the procedure given in 2.5.5); if yes, then the conjecture $E_1 = E_2$ is proved, under the assumption that the ndg-conditions hold. Otherwise the conjecture is not a theorem.

Checking the 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 Area Method

*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 statement. Therefore, as the simplification procedure and the procedure for deciding equality over independent parameters terminate, the whole of the method terminates as well. The number of ngd-conditions is always finite, so it can be proved by a simple inductive argument that the method terminates also if it is used for checking ndg-conditions (since in each recursive call there is less ndg-conditions).

*Correctness.*The area method (as described here) is applied to geometry statements of the form $C \Rightarrow E_1 = E_2$. If some of ndg-conditions is inconsistent with the previously introduced ndg-conditions, the formula C is inconsistent, so the statement is trivially a theorem.[7] Otherwise, the method transforms the initial formula to a formula $C \Rightarrow E_1' = E_2'$ such that the equality $E_1' = E_2'$ involves only independent variables.[8] Thanks to the properties of the elimination lemmas and of the simplification procedure, the initial formula[9] is a theorem (i.e., is a consequence of the axioms) if and only if the final formula is a theorem. Hence, if $E_1' = E_2'$ is provable, then the original statement is a theorem. If $E_1' = E_2'$ is not provable, the original statement is not a theorem (since C is consistent). In summary, the original formula is a theorem if and only if C is inconsistent or $E_1' = E_2'$ is provable. Therefore, thanks to the properties of the simplification procedure, if E_1' is identical to E_2', the statement is a theorem. Otherwise, since all geometric quantities appearing in E_1' and E_2' are independent parameters, in the geometric construction considered they can take arbitrary values, so it is possible to choose concrete values that lead to a counterexample for the statement. Therefore, the method is terminating, sound, and complete: for each geometry statement (defined in Section 2.3.2), the method can decide whether or not it is a theorem, i.e., the method is a decision procedure for that fragment of the theory with the given axiom system.[10]

Each conjecture that can be proved by the axioms for the area method is also a theorem of Hilbert's geometry (as explained in Section 2.2.2).

[7] The number of ngd-conditions is always finite, so it can be proved by a simple inductive argument that the area method can be used for checking ndg-conditions.

[8] In the non-branching variant of the method (see Section 2.5.1), the formula C may be augmented by additional ndg-conditions along the proving process.

[9] In the non-branching variant of the method (see Section 2.5.1), the initial formula may be updated.

[10] 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 (e.g., built upon Hilbert's axioms) augmented by the theory of fields (where the theory of fields enable expressing measures and expressions).

The area method can also be used for proving (some) geometry statements of the form $C \Rightarrow E_1 \neq E_2$. If C is inconsistent, the statement is trivially a theorem. Otherwise, the method transforms the initial formula to a formula $C \Rightarrow E_1' \neq E_2'$. The initial formula is a theorem if and only if the final formula is a theorem. Hence, if $E_1' \neq E_2'$ is provable,[11] then the original statement is a theorem. If $E_1' \neq E_2'$ is not provable, the original statement is not a theorem (since C is consistent). In summary, the original formula is a theorem if and only if C is inconsistent or $E_1' \neq E_2'$ is provable.

*Complexity.*The core of the method does not have branching (unless the variant 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 [9]).

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 replaces 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 [9]. 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

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

3.1.1 GCLC

GCLC[12] [29,32] is a tool for the 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 more than that: GCLC can be used in mathematical education, for storing visual mathematical contents in textual form (as figure descriptions in the underlying language), and for studying automated reasoning methods for geometry. The basic idea behind GCLC is that constructions are abstract, formal procedures, rather than images. Thus, in GCLC, mathematical objects are described rather than

[11] Proving $E_1' \neq E_2'$ may not be trivial, for instance, in the example $x^2 + 1 \neq 0$.

[12] `http://www.matf.bg.ac.rs/~janicic/gclc`

drawn. A figure can be generated (in the Cartesian model of Euclidean plane) on the basis of an abstract description. The language of GCLC [32] consists of commands for basic definitions and constructions, transformations, symbolic calculations, flow control, drawing and printing (including commands for drawing parametric curves and surfaces, functions, graphs, and trees), automated theorem proving, etc. Libraries of GCLC procedures provide additional features, such as support for hyperbolic geometry. GCLC has been under constant development since 1996. It is implemented in *C++*, and consist of around 40000 lines of code (automated theorem provers take around half of it, while the area method takes around 8000 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 3.2).

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

```
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'
```

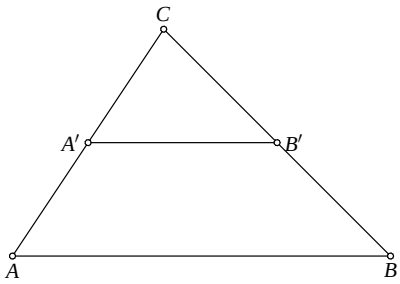


Fig. 3.1 A description of a triangle and midpoints of two of triangle's sides in GCLC language (left) and the corresponding illustration (right)

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, developed by Predrag Janičić and Pedro Quaresma [33], and algebraic theorem provers based on the Gröbner bases method and on Wu's method, developed by Goran Predović and Predrag Janičić [51]. Thanks to these theorem provers, GCLC links geometrical contents, visual information, and machine–generated proofs.

The provers are tightly integrated in GCLC — one can use the provers to reason about objects introduced in a GCLC construction without any adaptations other than the addition of the conjecture itself. GCLCprover transforms a construction command into a form required by the area method (and, for that purpose, may introduces some auxiliary points). A

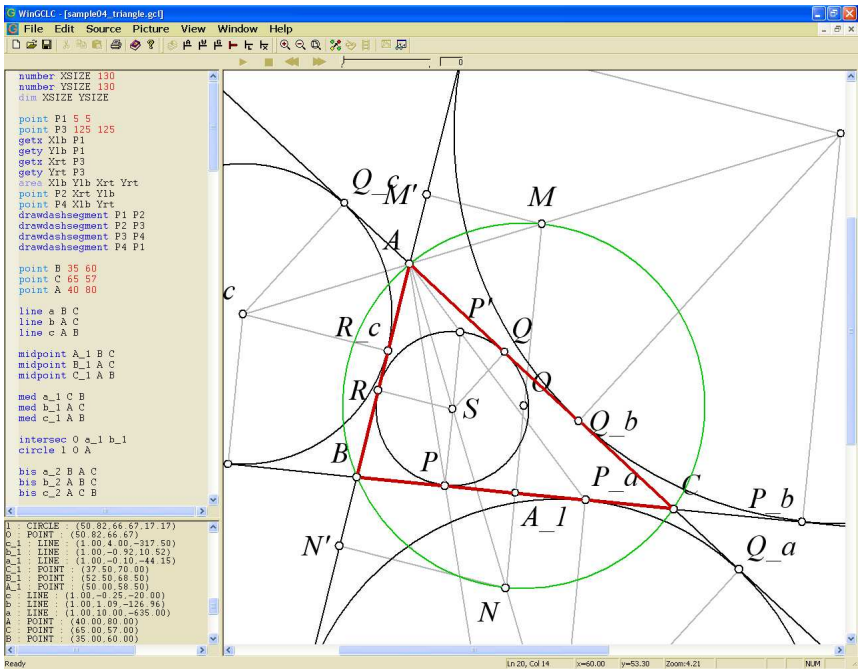


Fig. 3.2 WinGCLC Screenshot (the textual description on the left hand side and the visualisation on the right hand side depict the circumcircle, the inscribed circle, and the three escribed circles of the triangle *ABC*)

conjecture is given in the form $E_1 = E_2$, where E_1 and E_2 are expressions over geometric quantities. Alternatively, a conjecture can be given in the form of higher-level notions (given in Table 2.1). For instance, for the construction shown in Example 3.1, it holds that the lines AB and $A'B'$ are parallel and this conjecture can be given as an 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 and it considers only abstract specification of the construction (and not Cartesian coordinates of of the points involved, given by the user for visualisation purposes). There are GCLC commands for controlling a levels of detail for the output and for controlling the maximal number of proof steps or maximal time spent by the prover.

Thanks to the 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).

3.1.3 Specifics of the Implementation in GCLC

The algorithm implemented in GCLCprover is the one described in Section 2.5.7, with some specifics, used for increased efficiency and/or simpler implementation. With respect to the simplification procedure described in 2.5.3, there are the following specifics:

- The unary operator "$-$" is not used (and instead $-x$ is represented as $(-1)\cdot x$). Hence, the rules involving this operator are not used.
- The rules involving fractions given in 2.5.5 are not applied to ratios of segments. Instead, the following rules are used within the simplification procedure $\frac{\overline{AB}}{\overline{AB}} \to 1$, $\frac{\overline{AB}}{\overline{BA}} \to -1$.

- The following additional rules are used within the simplification phase:
  - $\frac{x}{c} \to (1/c) \cdot x$, where c is a constant (element of F) and $c \neq 1$.
  - $\frac{E_1 \cdot \ldots \cdot E_{i-1} \cdot C \cdot E_{i+1} \cdot \ldots \cdot E_n}{E'_1 \cdot \ldots \cdot E'_{j-1} \cdot C \cdot E'_{j+1} \cdot \ldots \cdot E'_m} \to \frac{E_1 \cdot \ldots \cdot E_{i-1} \cdot E_{i+1} \cdot \ldots \cdot E_n}{E'_1 \cdot \ldots \cdot E'_{j-1} \cdot E'_{j+1} \cdot \ldots \cdot E'_m}$
  - $E_1 + \cdots + E_{i-1} + c_1 \cdot C + E_{i+1} + \cdots + E_n = E'_1 + \cdots + E'_{j-1} + c_2 \cdot C' + E'_{j+1} + \cdots + E'_m \to E_1 + \cdots + E_{i-1} + c_3 \cdot C + E_{i+1} \cdots + E_n = E'_1 + \cdots + E'_{j-1} + E'_{j+1} + \cdots + E'_m$ 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, the rule $\mathcal{S}_{ABC} \to 0$ is applied for each three collinear points A, B, C.
- Reducing to area coordinates is not implemented. Instead, the following rules are applied at that stage:
  - $\overline{AA} \to 0$
  - $\mathcal{S}_{ABC} \to \mathcal{S}_{ABD} + \mathcal{S}_{ADC} + \mathcal{S}_{DBC}$, if there are terms $\mathcal{S}_{ABD}$, $\mathcal{S}_{ADC}$, $\mathcal{S}_{DBC}$ in the current goal.
  - $\mathcal{P}_{ABC} \to \overline{AB}^2 + \overline{CB}^2 + -1 \cdot \overline{AC}^2$

  Note that after these rules have been applied, the equality being proved may still involve dependent parameters. The simplification process is applied again and the equality is tested once more. Even without reducing to area coordinates, the above rules enable proving most conjectures from the area method scope.

Concerning ndg-conditions, the prover records and reports about the ndg-conditions of construction steps, but there is no check of the ndg-conditions within the main loop by the area method itself (as described in Section 2.5.7). Instead, there is a semantic check, using floating numbers and Cartesian coordinates associated to the free points by the user. For each construction step, it is checked if it is possible (e.g., if two lines do intersect) and these tests corresponds to checking the ndg-conditions of the geometry statement. If all these checks pass successfully (i.e., if all construction steps are possible), all the ndg-conditions are true in the concrete model, and hence, the assumptions of the statement are consistent. [13] In that case, the construction is visualised and the conjecture is sent to the prover. Otherwise, if some of the checks fails, an error is reported, the construction is not visualised, and the conjecture is not sent to the prover.

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).

[13] Ensuring consistency is important for the case that the original goal transforms to an equality that is not valid. In that case, the original statement is not a theorem (see Section 2.5.8).

3.1.4 Prover Output

The proofs generated by GCLCprover[14] can be exported to LaTeX or to XML form using a special-purpose styles and with options for different formatting. At the beginning of an output document, the auxiliary points are defined. For each proof step (a single transformation of the goal being proved), there is an ordinal numbers, an explanation and, optionally, its semantic counterpart — as a check (based on floating-point numbers) whether a conjecture is true in the specific case determined by Cartesian coordinates associated (by the user, for the sake of visualisation) to the free points of the construction (this semantic information is useful for conjectures for which is not known whether or not they are theorems). Lemmas (about side conditions) are proved within the main proof (making nested proof levels). At the end of the proof, all non-degeneracy conditions are listed. In the following is a fragment of the output (generated in LaTeX) for the conjecture from Example 3.1:

$S_{AA'B'}$	$= S_{BA'B'}$ by the statement	(1)
$S_{B'AA'}$	$= S_{B'BA'}$ by geometrical simplifications	(2)
$\left(S_{B'AA} + \left(\frac{1}{2} \cdot \left(S_{B'AC} + \left(-1 \cdot S_{B'AA}\right)\right)\right)\right)$	$= S_{B'BA'}$ 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 [1,28,61]. It allows one to express mathematical assertions and to mechanically check proofs of these assertions.

3.2.1 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 one to formalise proofs in different domains. For instance, it has been used for the formalisation of the four colour theorem [22] and the fundamental theorem of algebra [21]. 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* [38].

There are several recent results in the formalisation of elementary geometry in proof assistants: Hilbert's *Grundlagen* [27] has been formalised in Isabelle/Isar [42] and in *Coq* [16]. Gilles Kahn has formalised Jan von Plato's constructive geometry in the *Coq* system [34,49]. Frédérique Guilhot has made a large development in *Coq* dealing with French high school geometry [24]. Julien Narboux has formalised Tarski's geometry using the *Coq* proof assistant [46]. Jean Duprat proposes the formalisation in *Coq* of an axiom system for

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

compass and ruler geometry [17]. Projective geometry has also been formalised in *Coq* [40, 41].

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. The *Coq* development[15] consists of about 7000 lines of specifications (this includes the statements of the lemmas and the tactics), and 6400 lines of proofs.

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 using the language $\mathcal{L}_{tac}$[16] which is integrated in the system *Coq*.

We did not prove formally the completeness of the method implementation (i.e., that the tactic always succeeds if the conjecture is a theorem). Our formal proofs guarantee only the soundness of the method implementation (i.e., the proofs generated by the tactic are always correct).

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 corresponds 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 3.2.5. 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 (of the goals or of an assumption), the context always contains a proof that it is not 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.

[15] `http://dpt-info.u-strasbg.fr/~narboux/area_method.htm`

[16] The $\mathcal{L}_{tac}$ language is a domain specific language which allows the user to write his/her own proof schemes.

*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 points 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 satisfied.

*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).

We give a detailed description of how the tactic works on the example 3.2 by decomposing the procedure into small steps[17].

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.
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[18]:

[17] 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).

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

```
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.
```

3.2.4 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. The files containing the proof terms have size about 50KB per example.

These formal proofs are not readable, hence to have a readable proof we also output a human readable version of the proofs (using the print statement provided by $\mathcal{L}_{tac}$) 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...
```

3.2.5 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 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 [8]: 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 P Q r):

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

If $A = Y$, it may be the case that $CD \nparallel PQ$. This demonstrates that the lemma is provable 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.

3.2.6 Integration in GeoProof

Similarly to GCLC, the formalisation of the area method in *Coq* comes with a dynamic geometry tool [45]. The software developed, *GeoProof* combines three tools: a dynamic geometry tool 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-known and widely credited as one of the most efficient method for proving geometry theorems that produce readable proofs (at least in principle), 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, the Gröbner bases method adapted to geometry theorem proving, the full angle method [12], and the deductive database method [13]), counting versions employed within different systems. One of the 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. Another reason is that these methods still don't have many real-world applications (apart from applications in education). Having the area method fully formalised (as described in this paper) could help finding new applications, for instance, in formalisation projects such as Flyspeck [25].

In addition to the two implementations of the area method already described, we are aware of two other implementations: one used within a family of 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 a 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 [8]. It was implemented in *Common Lisp* and was accompanied by a list of 400 proved theorems.

Geometry Expert[19] (*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 [3]. *GEX* was implemented in 1998 by Xiao Shan Gao.

MMP/Geometer[20] is a new, Chinese, version of *GEX*. The tool has been developed in *Visual C* since 2002 by Xiao-Shan Gao and Qiang Lin. It automates geometry diagram generation, geometry theorem proving, and geometry theorem discovering [19]. 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.

Java Geometry Expert[21] (*JGEX*) is a new, Java version of *GEX* [65,66]. *JGEX* has been developed since 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 latest version (0.80, from May 2009), the area method and the *traditional 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[22] is a general mathematical tool with a uniform framework for computing, problem solving, and theorem proving [2]. *Theorema* is implemented in *Mathematica*. It has

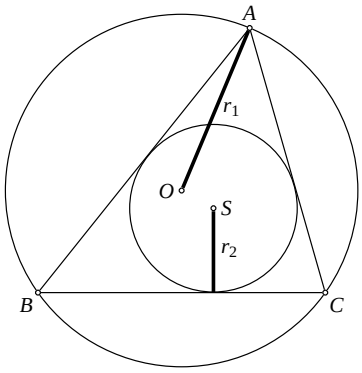
[19] `http://www.mmrc.iss.ac.cn/gex/`
[20] `http://www.mmrc.iss.ac.cn/mmsoft/`
[21] `http://www.jgex.net/`
[22] `http://www.theorema.org/`

been developed since 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 [58]. These provers were implemented by Judit Robu (the algebraic geometry theorem provers use implementations of algebraic methods from *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 a conjecture (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 $\leq$) is tested for validity by Collins' algorithm for quantifier elimination in real closed fields by cylindrical algebraic decomposition [15].

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*[23].



3.4 Applications

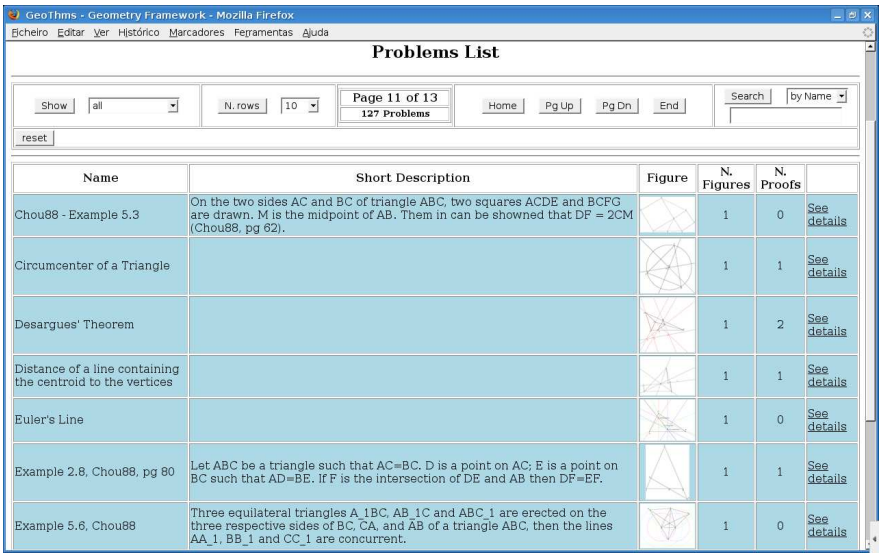
As other geometry theorem provers, the area method can have different applications in education, mathematical software, computer-aided design, computer graphics, computer vision, robotics, etc. [7], but also in formalisation projects such as Flyspeck which involves a lot of geometric reasoning [25]. In this section a few existing, rather straightforward applications, of the method are described.

[23] The statement can not be stated as $r_1 \geq 2r_2$ because, using the geometric quantities of the area method, only the square of an oriented distance can be expressed.

3.4.1 GeoThms

GeoThms[24] is a web-based framework for exploring geometrical knowledge that integrates dynamic geometry tools, automatic theorem provers, and a repository of geometric constructions, figures and proofs [53,55]. 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 tools currently used within GeoThms are GCLC [29] and Eukleides[25] [57], two widely used dynamic geometry tools. 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 [51].

GeoThms provides a web workbench that tightly integrates the mentioned tools into a single framework for constructive geometry.

The current collection (June 2010) of 176 problems was built using the examples in [47, 51], and also from [6,9,11,12]. From those problems, 111 are in the realm of the area method, 60 of them where coded in GCLC input format and the area method prover from GCLC was capable of proving successfully 56 of them within 600s of CPU time, in 4 other problems the prover was stopped before reaching its goal. The average CPU time was 3.5s, with a maximum of 69.98s and a minimum of less them 0.001s. The *Coq* based prover was capable of proving successfully 66 problems (coded in *Coq* format), under the time limit of 600s, in 6 other problems the prover was unable to complete the proof. The average CPU time was 18.23s, with a maximum of 213.71s and a minimum of 0.73s. On the set of problems in which both implementations where tested the GCLCprover was significantly more efficient than the *Coq* based prover.[26]

[24] `http://hilbert.mat.uc.pt/GeoThms`

[25] `http://www.eukleides.org/`

[26] All the CPU times where taken in a Pentium®4 CPU 3.00GHz, 2GB RAM, GNU/Linux

A more extensive set of problems should be built to have a better understanding of the capabilities of both implementations, this is being done within the projects GeoThms and TGTP.[27]

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. A formal argument that a construction step is always impossible 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 [30].

Example 3.4 *Example 85 from the book* Mechanical Geometry Theorem Proving *[5] will be used to illustrate the mechanism for automatic verification of regular constructions built into GCLC. Using GCLC, the illustration given in Figure 3.3 can be generated.*

If the code contains the intersection of lines AD and MN, GCLC will report that such intersection cannot be determined (using floating-point numbers and the concrete set, given by the user, of the free points in the Cartesian plane). Further, it will invoke the built-in theorem prover and prove the conjecture that the two lines AD and MN are parallel (hence, for any choice of free points, the intersection of lines AD and MN cannot be determined).

As far as we are aware of, the system for automated *deductive* testing whether a construction is illegal, an important feature that enhances the didactic nature of dynamic geometry tools, 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 illegal construction steps to be performed. However the justification is not based on deductive but on probabilistic reasoning [37].

[27] `http://hilbert.mat.uc.pt/TGTP/`

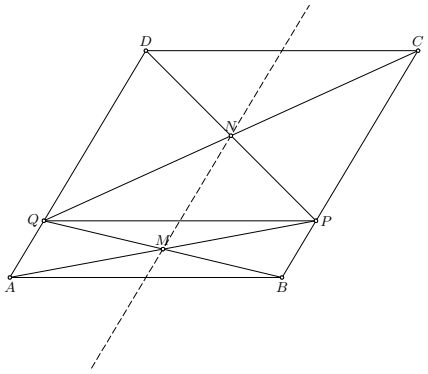
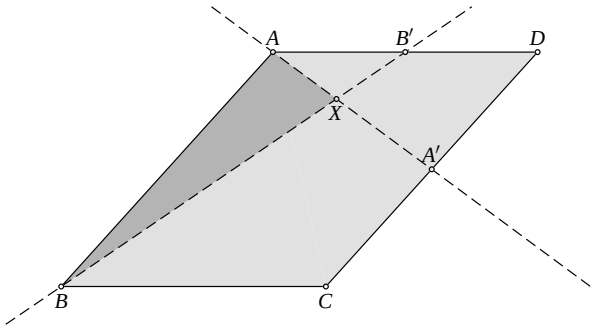


Fig. 3.3 Example 85 from the book *Mechanical Geometry Theorem Proving*

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 [58].

Example 3.5 *Let A, B and C be arbitrary points and let r be 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* [28].

[28] This extension of the method was originally described by the authors of the method [9].

3.4.4 Discovering Geometry Properties

Within *Theorema*, the area method machinery is used for exploring geometrical configurations and discovering geometry properties [58]. 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 [9] (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 correctness of the method not only for affine geometry (already described before [43]), but also for Euclidean geometry [47]. 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 [9], contained an error.
- We provided detailed traditional proofs in the Hilbert-style system (in a technical report accompanying this paper [56]) 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 formulae to be used for dealing with free points (Section 2.5.4).
- We made an explicit description of the simplification phase (Section 2.5.3).
- 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 3.2.5).

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 , Gao and Zhang

in 1993. The method produces proofs that are often concise and human-readable, 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.

Acknowledgements We thank the anonymous referees for the very helpful comments on the first version of this paper.

References

1. Bertot, Y., Castéran, P.: Interactive Theorem Proving and Program Development, Coq'Art: The Calculus of Inductive Constructions. Texts in Theoretical Computer Science. An EATCS Series. Springer (2004)
2. Buchberger, B., Craciun, A., Jebelean, T., Kovacs, L., Kutsia, T., Nakagawa, K., Piroi, F., Popov, N., Robu, J., Rosenkranz, M., Windsteiger, W.: Theorema: Towards computer-aided mathematical theory exploration. Journal of Applied Logic **4**, 470–504 (2006)
3. Chou, S., Gao, X., Zhang, J.: An introduction to geometry expert. In: M.A. McRobbie, J.K. Slaney (eds.) Proc. CADE-13, *Lecture Notes in Computer Science*, vol. 1104, pp. 235–239. Springer-Verlag (1996)
4. Chou, S.C.: Proving and discovering geometry theorems using wu's method. Ph.D. thesis, The University of Texas, Austin (1985)
5. Chou, S.C.: Mechanical Geometry Theorem Proving. D. Reidel Publishing Company, Dordrecht (1987)
6. Chou, S.C.: Mechanical Geometry Theorem Proving. D. Reidel Publishing Company (1988)
7. Chou, S.C., Gao, X.S.: Automated reasoning in geometry. In: J.A. Robinson, A. Voronkov (eds.) Handbook of Automated Reasoning, pp. 707–749. Elsevier and MIT Press (2001)
8. Chou, S.C., Gao, X.S., Zhang, J.Z.: 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. IEEE Computer Society Press (1993)
9. Chou, S.C., Gao, X.S., Zhang, J.Z.: Machine Proofs in Geometry. World Scientific, Singapore (1994)
10. Chou, S.C., Gao, X.S., Zhang, J.Z.: Automated production of traditional proofs in solid geometry. Journal of Automated Reasoning **14**, 257–291 (1995)
11. Chou, S.C., Gao, X.S., Zhang, J.Z.: Automated generation of readable proofs with geometric invariants, I. multiple and shortest proof generation. Journal of Automated Reasoning **17**, 325–347 (1996)
12. Chou, S.C., Gao, X.S., Zhang, J.Z.: Automated generation of readable proofs with geometric invariants, II. theorem proving with full-angles. Journal of Automated Reasoning **17**, 349–370 (1996)
13. Chou, S.C., Gao, X.S., Zhang, J.Z.: A deductive database approach to automated geometry theorem proving and discovering. Journal of Automated Reasoning **25**, 219–246 (2000)
14. Coelho, H., Pereira, L.M.: Automated reasoning in geometry theorem proving with prolog. Journal of Automated Reasoning **2**(4), 329–390 (1986).
15. Collins, G.E.: Quantifier elimination for real closed fields by cylindrical algebraic decomposition. In: Automata Theory and Formal Languages 2nd GI Conference Kaiserslautern, May 2023, 1975, *Lecture Notes In Computer Science*, vol. 33, pp. 134–183. Springer (1975)
16. Dehlinger, C., Dufourd, J.F., Schreck, P.: Higher-order intuitionistic formalization and proofs in Hilbert's elementary geometry. In: D.W. Jrgen Richter-Gebert (ed.) Proceedings of Automated Deduction in Geometry (ADG00), *Lecture Notes in Computer Science*, vol. 2061, pp. 306–324 (2000)
17. Duprat, J.: The Euclid's Plane : Formalization and Implementation in Coq. In: Proceedings of ADG'10 (2010)
18. Elcock, E.W.: Representation of knowledge in geometry machine. Machine Intelligence **8**, 11–29 (1977)
19. Gao, X.S., Lin, Q.: MMP/Geometer - A Software Package for Automated Geometric Reasoning. In: F. Winkler (ed.) Proceedings of Automated Deduction in Geometry (ADG02), *Lecture Notes in Computer Science*, vol. 2930, pp. 44–66. Springer-Verlag (2004)
20. Gelernter, H.: Realization of a geometry-theorem proving machine. In: Computers & thought, pp. 134–152. MIT Press, Cambridge, MA, USA (1995)

21. Geuvers, H., et.al.: The “fundamental theorem of algebra” project. `http://www.cs.ru.nl/~freek/fta/` (2008)
22. Gonthier, G., Werner, B.: A computer checked proof of the four colour theorem. (2004)
23. Greeno, J., Magone, M.E., Chaiklin, S.: Theory of constructions and set in problem solving. Memory and Cognition **7**(6), 445–461 (1979).
24. Guilhot, F.: Formalisation en Coq d’un cours de géométrie pour le lycée. In: Journées Francophones des Langages Applicatifs (2004)
25. Hales, T.C.: Introduction to the flyspeck project. In: T. Coquand, H. Lombardi, M.F. Roy (eds.) Mathematics, Algorithms, Proofs, *Dagstuhl Seminar Proceedings*, vol. 05021. Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, Germany (2006)
26. Heath, T.L.: The Thirteen Books of Euclid’s Elements. Dover Publications, New-York (1956). 2nd ed.
27. Hilbert, D.: Foundations of Geometry. Open Court Publishing (1977). 10th Revised edition. Editor: Paul Barnays
28. Huet, G., Kahn, G., Paulin-Mohring, C.: The Coq Proof Assistant - A tutorial - Version 8.0 (2004). `http://coq.inria.fr`
29. Janičić, P.: GCLC – a tool for constructive Euclidean geometry and more than that. In: N. Takayama, A. Iglesias, J. Gutierrez (eds.) Proceedings of International Congress of Mathematical Software (ICMS 2006), *Lecture Notes in Artificial Intelligence*, vol. 4151. Springer-Verlag (2006)
30. Janičić, P., Quaresma, P.: Automatic verification of regular constructions in dynamic geometry systems. In: F. Botana, T. Recio (eds.) Proceedings of Automated Deduction in Geometry (ADG06), *Lecture Notes in Artificial Intelligence*, vol. 4869, pp. 39–51. Springer-Verlag, Pontevedra, Spain (2007)
31. Janičić, P.: One method for automathed theorem proving in geometry. Master’s thesis, Faculty of Mathematics, University of Belgrade (1996). In Serbian
32. Janičić, P.: Geometry Constructions Language. Journal of Automated Reasoning **44**(1-2), 3–24 (2010)
33. Janičić, P., Quaresma, P.: System Description: GCLCprover + GeoThms. In: F. Ulrich, S. Natarajan (eds.) Automated Reasoning, *Lecture Notes in Artificial Intelligence*, vol. 4130, pp. 145–150. Springer-Verlag (2006)
34. Kahn, G.: Constructive geometry according to Jan von Plato. Coq contribution (1995). Coq V5.10
35. Kapur, D.: Geometry Theorem Proving using Hilbert’s Nullstellensatz. In: SYMSAC ’86: Proceedings of the fifth ACM symposium on Symbolic and algebraic computation, pp. 202–208. ACM Press, New York, NY, USA (1986)
36. Kapur, D.: Using Gröbner bases to reason about geometry problems. Journal of Symbolic Computation **2**(4), 399–408 (1986).
37. Kortenkamp, U., Richter-Gebert, J.: Using automatic theorem proving to improve the usability of geometry software. In: Workshop on Mathematical User Interfaces (2004)
38. Leroy, X.: 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. ACM Press (2006)
39. Li, H.: Clifford algebra approaches to mechanical geometry theorem proving. In: X.S. Gao, D. Wang (eds.) Mathematics Mechanization and Applications, pp. 205–299. Academic Press, San Diego, CA (2000).
40. Magaud, N., Narboux, J., Schreck, P.: Formalizing Desargues’ Theorem in Coq using Ranks. In: S.Y. Shin, S. Ossowski (eds.) SAC, pp. 1110–1115. ACM (2009)
41. Magaud, N., Narboux, J., Schreck, P.: Formalizing Projective Plane Geometry in Coq. In: Post-Proceedings of ADG’08, Lecture Notes in Artifical Intelligence (2010). To appear
42. Meikle, L., Fleuriot, J.: Formalizing Hilbert’s Grundlagen in Isabelle/Isar. In: D.A. Basin, B. Wolff (eds.) Theorem Proving in Higher Order Logics, *Lecture Notes in Computer Science*, vol. 2758, pp. 319–334. Springer-Verlag (2003)
43. Narboux, J.: A decision procedure for geometry in Coq. In: S. Konrad, B. Annett, G. Ganesh (eds.) Proceedings of TPHOLs’2004, *Lecture Notes in Computer Science*, vol. 3223. Springer-Verlag (2004)
44. Narboux, J.: Formalisation et automatisation du raisonnement géométrique en Coq. Ph.D. thesis, Université Paris Sud (2006)
45. Narboux, J.: A graphical user interface for formal proofs in geometry. Journal of Automated Reasoning **39**(2), 161–180 (2007)
46. Narboux, J.: Mechanical theorem proving in Tarski’s geometry. In: Proceedings of Automatic Deduction in Geometry 06, *Lecture Notes in Artificial Intelligence*, vol. 4869, pp. 139–156. Springer-Verlag (2007)
47. Narboux, J.: Formalization of the area method. Coq user contribution (2009). `http://dpt-info.u-strasbg.fr/~narboux/area_method.html`
48. Nevis, A.: Plane geometry theorem proving using forward chaining. Artificial Intelligence **6**(1), 1–23 (1975).
49. von Plato, J.: The axioms of constructive geometry. In: Annals of Pure and Applied Logic, vol. 76, pp. 169–200 (1995)

50. von Plato, J.: Formalization of Hilbert's geometry of incidence and parallelism. In: Synthese, vol. 110, pp. 127–141. Springer (1997)
51. Predović, G.: Automated geometry theorem proving based on Wu's and Buchberger's methods. Master's thesis, Faculty of Mathematics, University of Belgrade (2008). Supervisor: Predrag Janičić (in Serbian)
52. Quaresma, P., Janičić, P.: Framework for constructive geometry (based on the area method). Tech. Rep. 2006/001, Centre for Informatics and Systems of the University of Coimbra (2006)
53. Quaresma, P., Janičić, P.: Geothms - a web system for euclidean constructive geometry. In: S. Autexier, C. Benzmüller (eds.) UITP 2006, vol. 174, pp. 21–33 (2006)
54. Quaresma, P., Janičić, P.: Geothms - geometry framework. Tech. Rep. 2006/002, Centre for Informatics and Systems of the University of Coimbra (2006)
55. Quaresma, P., Janičić, P.: Integrating dynamic geometry software, deduction systems, and theorem repositories. In: J. M. Borwein, W. M. Farmer (eds.) Mathematical Knowledge Management, *Lecture Notes in Artificial Intelligence*, vol. 4108, pp. 280–294. Springer (2006)
56. Quaresma, P., Janičić, P.: The area method - properties and their proofs. Tech. Rep. 2009/006, Centre for Informatics and Systems of the University of Coimbra (2009)
57. Quaresma, P., Pereira, A.: Visualização de construções geométricas. Gazeta de Matemática **151** (2006)
58. Robu, J.: Geometry theorem proving in the frame of the Theorema project. Ph.D. thesis, Johannes Kepler Universität, Linz (2002)
59. Tarski, A.: A decision method for elementary algebra and geometry. University of California Press (1951)
60. Tarski, A.: What is elementary geometry? In: P.S. L. Henkin, A. Tarski (eds.) The axiomatic Method, with special reference to Geometry and Physics, pp. 16–29. North-Holland, Amsterdam (1959)
61. The Coq development team: The Coq proof assistant reference manual, Version 8.2. TypiCal Project (2009). `http://coq.inria.fr`
62. Wang, D.: Reasoning about geometric problems using an elimination method. In: J. Pfalzgraf, D. Wang (eds.) Automated Practical Reasoning, pp. 147–185. Springer, New York (1995).
63. Wu, W.T.: Automated Theorem Proving: After 25 Years, vol. 29, chap. On the decision problem and the mechanization of theorem proving in elementary geometry, pp. 213–234. American Mathematical Society (1984)
64. Yang, L., Gao, X., Chou, S., Zhang, Z.: Automated proving and discovering of theorems in non-euclidean geometries. In: Proceedings of Automated Deduction in Geometry (ADG98), *Lecture Notes in Artificial Intelligence*, vol. 1360, pp. 171–188. Springer-Verlag, Berlin, Heidelberg (1998)
65. Ye, Z., Chou, S.C., Gao, X.S.: Visually Dynamic Presentation of Proofs in Plane Geometry. Part 1. Basic Features and the Manual Input Method. Journal of Automated Reasoning (2010)
66. Ye, Z., Chou, S.C., Gao, X.S.: Visually Dynamic Presentation of Proofs in Plane Geometry. Part 2. Automated Generation of Visually Dynamic Presentations with the Full-Angle Method and the Deductive Database Method. Journal of Automated Reasoning (2010)
67. Zhang, J.Z., Chou, S.C., Gao, X.S.: Automated production of traditional proofs for theorems in Euclidean geometry. Annals of Mathematics and Artificial Intelligence **13**, 109–137 (1995)