A Graphical User Interface for Formal Proofs in Geometry.

Julien Narboux

A Graphical User Interface for Formal Proofs in Geometry

Julien Narboux (julien.narboux@inria.fr)
Project PCRI Pôle Commun de Recherche en Informatique du plateau de Saclay, CNRS, École Polytechnique, INRIA, Université Paris-Sud.

October 10, 2006

Abstract. We present in this paper the design of a graphical user interface to deal with proofs in geometry. The software developed combines three tools: a dynamic geometry software to explore, measure 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.

Keywords: geometry, theorem prover, proof assistant, interface, Coq, dynamic geometry, automated theorem proving

1. Introduction

Dynamic Geometry Software (DGS) and Computer Algebra Software (CAS) are the most widely used software for mathematics in the education. DGSs allow the user to create complex geometric constructions step by step using free objects such as free points and predefined atomic constructions depending on other objects (for instance the line passing through two points, the midpoint of a segment, *etc.*). The free objects can be dragged using the mouse and the figure is updated in real time. CAS allow symbolic manipulations of mathematical expressions.

The most widely used systems are the historical ones which appeared in the 90s, namely Geometer's sketchpad (Jackiw, 1990) and Cabri Geometer (Laborde and Bellemain, 1998). But there exists a large number of free and commercial software as well [1].

The education community has studied the impact of the use of these software on the *proving* activity (Yevdokimov, 2004; Furinghetti and Domingo, 2003). DGSs are mainly used for two activities:

– to make the student create geometric constructions;

[1] We can cite (the list is not intended to be exhaustive): CaR, Chypre Cinderella, Déclic, Defi, Dr. Geo, Euclid, Euklid DynaGeo, Eukleides, Gava, GeoExp, GeoFlash, GeoLabo, GeoLog, Geometria, Geometrix, Geometry Explorer, Geometry Tutor, GeoPlanW, GeoSpaceW, GEUP, GeoView, GEX, GRACE, KGeo, KIG, Mentoniezh, MM-Geometer, Non-Euclid, XCas, *etc.*

– to make the student explore the figure, invent conjectures and check facts.

We believe that these software systems should also be used to help the student in the proving activity itself. Work has been performed in this direction and several DGS with proof related features have been produced. These systems can be roughly classified into two categories:

1. the systems which permit to build proofs;
2. the systems which permit to check facts using an automated theorem prover.

The *Geometry Tutor* (Anderson et al., 1985), *Mentoniezh* (Py, 1990), *Defi* (Ag-Almouloud, 1992), *Chypre* (Bernat, 1993), *Cabri-Euclide* (Luengo, 1997), *Geometrix* (Gressier, 1998) and *Baghera* (Balacheff et al., 2002) systems belongs to the first category. Using these systems the student can produce proofs interactively using a set of known theorems. In most of these systems the student can not invent a proof very different from what the program had pre-computed using automated theorem proving methods. As far as we know, the exception is *Cabri-Euclide* which contains a small formal system and therefore gives more freedom to the student. *Baghera* includes also e-learning features, such as task management and network communication between teachers and their students.
MMP-Geometer(Gao, 2000),*Geometry Expert* (Gao and Lin, 2002), *Geometry Explorer*(Wilson and Fleuriot, 2005) and *Cinderella* (Kortenkamp, 1999; Kortenkamp and Richter-Gebert, 2004; Richter-Gebert and Kortenkamp, 1999; Schwartz, 1979) belongs to the second category. *Geometry Expert* and *MMP-Geometer* are DGS which are used as a graphical interface for an implementation of the main decision procedures in geometry. *Geometry Explorer* provides a diagrammatic visualization of proofs generated automatically by a prolog implementation of Chou's full angle method (Chou et al., 1996). *Cinderella* allows to export the description of the figure to computer algebra software to perform algebraic proofs.

The work closest to ours is (Bertot et al., 2003). The *GeoView* software provides a visualization tool for some formal geometric statements using an off-the-shelf DGS and the PCoq user interface for Coq (Bertot and Thery, 1998; Amerkad et al., 2001). It is intended to be used with the formalization of geometry for the French curriculum by Frédérique Guilhot (Guilhot, 2005) in the Coq proof assistant (Coq development team, The, 2004).

We present in this paper the design of a system whose aim is to combine automatic theorem proving, interactive theorem proving using

a formal proof system (the Coq proof assistant) and diagrammatic visualization. The difference between our approach and the other systems we have cited (except GeoView) is that we use of a general purpose proof assistant and combine interactive and automated theorem proving. The difference between our system and GeoView is that communication with Coq goes in the other direction.

Our approach is guided by the following motivations:

- It is very natural in geometry to illustrate a proof by a diagrammatic representation and even sometimes a diagram can be seen as a high level description of a proof (Barwise and Allwein, 1996; Jamnik, 2001; Miller, 2001; Wilson and Fleuriot, 2005; Winterstein, 2004a; Winterstein, 2004b). But sometimes a diagram can be misleading. That is why the verification of the proof by a formal proof system is crucial as it provides a very high level of confidence.

- Compared to an *adhoc* proof system specialized in geometry, the use of a general purpose proof assistant such as the Coq proof assistant provides a way to combine geometrical proofs with larger proofs. For example, it is possible to use the Coq system to prove facts about polygons by induction on the number of edges, or facts about transformations using complex numbers.

- There are facts that can not be visualized graphically and there are facts that are difficult to understand without a graphical representation. Hence, we need to combine both approaches.

- We should have both the ability to make arbitrarily complex proofs or to use a base of known lemmas, depending on the level of the user/student.

We will first give a short introduction of our prototype named GeoProof. Then we will focus on the proof related features of GeoProof: *automatic* theorem proving and interactive generation of Coq statements.

2. An overview of GeoProof

GeoProof is a free and open source Dynamic Geometry Software. It allows one to create and then manipulate geometric constructions. It is distributed under the term of the GPL Version 2 license. It has been implemented by starting from a project called DrGeoCaml initially developed by *Nicolas François*. GeoProof is written in the *Ocaml* programming language using only portable libraries in such a way that

it can be compiled for Linux, Windows and MacOSX.
In this section, we focus on the dynamic geometry features of GeoProof, the proof oriented functionality will be described in the next sections. Figure 1 gives a quick overview of the graphical user interface of GeoProof. The different tools can be sorted in four categories. The constructions tools are used to create new geometric objects. GeoProof supports the main geometric constructions and transformations involving points, circles, lines, segments and vectors.
The visualization tools allow to change the zoom factor and move the figure on the page. The manipulation tools allow to select, delete and move objects. The measures and tests tools are shortcuts to create special dynamic labels (those are described in the section 2.2). For instance the tool to test if two lines are parallel creates a textual label which tells if the two lines are parallel on the instance of the figure which is currently displayed. These test tools do not provide a proof, they should be used to quickly test the validity of a conjecture on several instances of the figure by manipulating the free points.
To simplify the creation of large figures, the user can organize the objects using layers and change the drawing style of the objects (hidden or not, dashed or not, color ...). A complete description of the features of GeoProof can be found in (Narboux, 2006d).

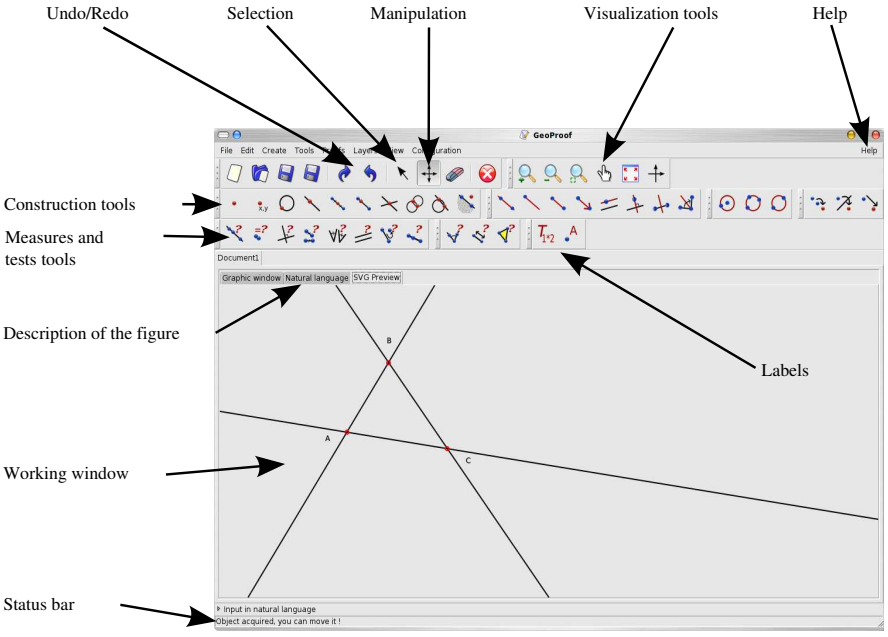


Figure 1. A screen-shot of the main window of GeoProof.

2.1. Input/Output

The documents can be saved using an open format based on the XML technology. It can export the figures using a bitmap format (PNG, BMP, JPEG), a vector graphic format (SVG) or a textual description in pseudo-natural language.
The description of the figure can also be exported to the input language of the Eukleides software to ease the insertion of figures in a LaTeX document[2]. The language used by Eukleides for the description of figures is high level. This means that after creating the figure using GeoProof, if the user wants to perform small changes it is not necessary to open it again using GeoProof, the description is readable enough to be edited directly within the LaTeX file. Figure 2 shows an example script.

```
frame(-10.00000,6.00000,12.48000,-3.90000,0.93416)
A = point(-3.22000,4.30000)
color(red)
thickness(2)
draw(A,dot)
color(black)
draw("A",A,0.28000,arg(circle(A,1),point(1.400,1.400)):)
...

...
Segment_3 = segment(C,A)
color(black)
thickness(2)
draw(Segment_3,full)
Line_1 = line(D,E)
color(blue)
thickness(2)
draw(Line_1,dashed)
```

Figure 2. Export to LaTeX using *Eukleides*.

2.2. Dynamic labels

A dynamic label is a text element enriched with the possibility to display the result of a computation defined using a small language (Narboux, 2006d). Textual labels which appear in a figure can contain *dynamic fields*. Dynamic fields contains expressions which are evaluated in real

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

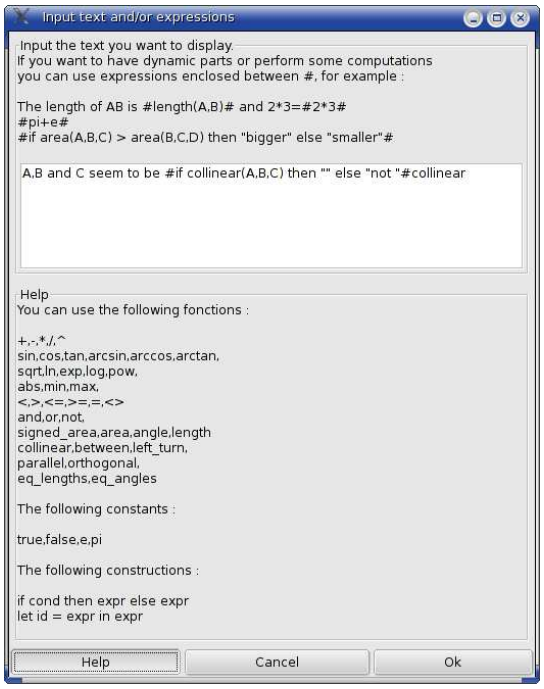


Figure 3. The definition of a dynamic label.

time when the figure is manipulated. Dynamic field are delimited by the sign `#`. As all the computations done by GeoProof, the evaluation of these expressions is performed using arbitrary precision. Thanks to a configuration file the user can choose at which precision the computations are performed. If the mathematical expressions contained in the text elements depend on other points of the figure, the text is updated in real time when the user changes the position of the free points. The dynamic part of the labels can contain measures and predicate tests using variables depending on other objects. For instance if the user wants to define a label to compare the size of two triangles he can define the following label: .

```
The triangle ABC is #if area(A,B,C)>area(D,E,F) then
"bigger" else "smaller"# than the triangle DEF.
```

Figure 3 shows an example of a dynamic label to test if three points are collinear. Using predefined dynamic labels the user can check easily for example if two lines are parallel (on the specific instance of the figure displayed).

3. Automatic proof

We present in this section how GeoProof can communicate with automatic theorem proving tools. We have implemented automatic theorem proving in GeoProof using two different systems: the first one takes advantage of an implementation of the Gröbner basis and Wu methods (Wu, 1978; Chou, 1988) written by John Harrison (Harrison, 2003), the second one consists of exporting to our own implementation of Chou's decision procedure for affine geometry (Chou et al., 1994) in the Coq proof assistant (Narboux, 2004). The implementation by John Harrison was designed to accompany a textbook on automated theorem proving and is hence not intended to be efficient. We have chosen this implementation because it is free and can be tightly integrated with GeoProof. We plan to add the possibility to use the other implementations provided by the CAS.

3.1. Using embedded automatic theorem prover

The formalization used by John Harrison is based on a theory with only points as basic objects whereas GeoProof uses points, lines and circles as the basic mathematical objects. We need to translate from one language to the other one. The input of the ATP is a first order formula with the following predicates: *collinear*, *parallel*, *perpendicular*, *eq_distance* (written as $AB = CD$) and *eq_angles*. These predicates are defined using an algebraic formula using the coordinates of the points.
Let x_P and y_P be the x and y coordinates of P.

$collinear(A, B, C) \equiv$

$$(x_A - x_B)(y_B - y_C) - (x_B - x_C)(y_A - y_B) = 0$$

$parallel(A, B, C, D) \equiv$

$$(x_A - x_B)(y_C - y_D) - (x_C - x_D)(y_A - y_B) = 0$$

$perpendicular(A, B, C, D) \equiv$

$$(x_A - x_B)(x_C - x_D) + (y_A - y_B)(y_C - y_D) = 0$$

$eq\_distance(A, B, C, D) \equiv$

$$(x_A - x_B)^2 + (y_A - y_B)^2 - (x_C - x_D)^2 - (y_C - y_D)^2 = 0$$

$$eq\_angle(A,B,C,D,E,F) \equiv$$
$$\begin{aligned}&((y_B - y_A) * (x_B - x_C) - (y_B - y_C) * (x_B - x_A)) *\\ &((x_E - x_D) * (x_E - x_F) + (y_E - y_D) * (y_E - y_F))\\ =\\ &((y_E - y_D) * (x_E - x_F) - (y_E - y_F) * (x_E - x_D)) *\\ &((x_B - x_A) * (x_B - x_C) + (y_B - y_A) * (y_B - y_C))\end{aligned}$$

3.1.1. *Translating a construction into a statement for ATP.*
We need to translate from one language to the other one. The idea of the translation consist of maintaining the invariant that lines and circles are always defined by two points. Of course this is not true in GeoProof. For instance one can build a line as the parallel of another line passing through a point. In such a case we need to define a second defining point for the line. For that purpose we generate new points during the translation. We define the translation by case distinction on the construction. Table I gives the defining points for each line and circle depending on how these objects have been constructed. $P1_l$,$P2_l$ and O_c are fresh variables. For each line and circle we associate some fresh variables. These new variables which do not appear in the original figure are used to define lines and circles when we do not have two points on the object on the figure we translate from.
Lines are defined by two points $\mathcal{P}_1(l)$ and $\mathcal{P}_2(l)$. When we already know at least one of the defining points we use it instead of creating a new point because it simplifies the generated formulas.
Circles are defined by their center $\mathcal{O}(c)$ and a point $\mathcal{P}(c)$ on the circle. Table II provides the translation of GeoProof constructions[3] into the language accepted by the embedded theorem prover. Incidentally, it gives a subset of the constructions of the language of GeoProof. The non degeneracy conditions are inspired by those in (Chou and Gao, 1992). The predicate *isotropic* is defined by:

$$isotropic(A,B) \equiv perpendicular(A,B,A,B)$$

In Euclidean geometry it is equivalent to $A = B$ but not in metric geometry. We produce a statement which is interpreted in the metric geometry because Wu and Gröbner bases methods are complete only for metric geometry. For more information about this see (Chou and Gao, 1992; Chou, 1988). Moreover if I_1 and I_2 are the two intersections of a circle and of a line or a circle then we add the fact that $I_1 \neq I_2$ in the hypotheses. Note that different constructions of the same figure can lead to different degeneracy conditions and hence different formulas.

[3] To simplify the presentation we only provide the translation for the main GeoProof constructions.

Table I. Definition of the defining points of circles and lines

GeoProof Construction	**Defining points**
l passing through A and B	$\mathcal{P}_1(l) = A\ \mathcal{P}_2(l) = B$
l parallel line to m passing through A	$\mathcal{P}_1(l) = A\ \mathcal{P}_2(l) = P2_l$
l perpendicular line to m passing through A	$\mathcal{P}_1(l) = A\ \mathcal{P}_2(l) = P2_l$
l perpendicular bisector of A and B	$\mathcal{P}_1(l) = P1_l\ \mathcal{P}_2(l) = P2_l$
l bisector of the angle formed by A, B and C	$\mathcal{P}_1(l) = B\ \mathcal{P}_2(l) = P2_l$
c circle of center O passing through A	$\mathcal{O}(c) = O\ \mathcal{P}(c) = A$
c circle whose diameter is A B	$\mathcal{O}(c) = O_c\ \mathcal{P}(c) = A$

3.1.2. *Correctness of the translation*

To convince the reader that the translation we give is *correct* in the sense it corresponds to the intuition the user of *GeoProof* can have, we will prove that the translation we give is equivalent to a more intuitive semantic based on points, lines and circles. This semantic is given in Table III.

We assume that we have three types of objects: $Point$, $Line$ and $Circle$. We assume we have two relations of incidence[4]:

$$\_ \in \_ : Point \to Line \to Prop$$

and

$$\_ \in \_ : Point \to Circle \to Prop$$

We assume that we have the perpendicular and parallel predicates over lines:

$$\_ \parallel \_ : Line \to Line \to Prop$$

and

$$\_ \perp \_ : Line \to Line \to Prop$$

We assume that we have a predicate expressing the fact that a point is the center of a circle:

$$\_\ is\_center\ \_ : Point \to Circle \to Prop$$

We want to show that the formulas defined by the two semantics are equisatisfiable. We follow the definition of the translation and prove the property by case distinction, we only show a few cases:

Point P on line l We need to perform another case distinction on the way l has been constructed:

[4] Note that the notation $\in$ is overloaded here.

Table II. Predicate form for each type of construction

GeoProof Construction	**Predicate form**
Free point	$true$
Point P on line l	$collinear(P, \mathcal{P}_1(l), \mathcal{P}_2(l))$
Point P on circle c	$\mathcal{O}(c)\mathcal{P}(c) = P\mathcal{O}(c)$
I midpoint of A and B	$IA = IB \wedge collinear(I, A, B)$
I intersection of l_1 and l_2	$collinear(I, \mathcal{P}_1(l_1), \mathcal{P}_2(l_1)) \wedge$ $collinear(I, \mathcal{P}_1(l_2), \mathcal{P}_2(l_2)) \wedge$ $\neg parallel(\mathcal{P}_1(l_1), \mathcal{P}_2(l_1), \mathcal{P}_1(l_2), \mathcal{P}_2(l_2))$
I an intersection of c_1 and c_2	$I\mathcal{O}(c_1) = \mathcal{O}(c_1)\mathcal{P}(c_1) \wedge$ $I\mathcal{O}(c_2) = \mathcal{O}(c_2)\mathcal{P}(c_2) \wedge$ $\neg isotropic(\mathcal{O}(c_1), \mathcal{O}(c_2))$
I an intersection of c and l	$I\mathcal{O}(c) = \mathcal{O}(c)\mathcal{P}(c) \wedge$ $collinear(I, \mathcal{P}_1(l), \mathcal{P}_2(l)) \wedge$ $\neg isotropic(\mathcal{P}_1(l), \mathcal{P}_2(l))$
l passing through A and B	$A \neq B$
l parallel to m passing through A	$parallel(A, \mathcal{P}_2(l), \mathcal{P}_1(m), \mathcal{P}_2(m)) \wedge$ $A \neq \mathcal{P}_2(l)$
l perpendicular to m passing through A	$perpendicular(A, \mathcal{P}_2(l), \mathcal{P}_1(m), \mathcal{P}_2(m)) \wedge$ $A \neq \mathcal{P}_2(l)$
l perpendicular bisector of A and B	$\mathcal{P}_1(l)A = \mathcal{P}_1(l)B \wedge \mathcal{P}_2(l)A = \mathcal{P}_2(l)B \wedge$ $\mathcal{P}_1(l) \neq \mathcal{P}_2(l) \wedge A \neq B$
l bisector of the angle A,B,C	$eq\_angle(A, B, \mathcal{P}_2(l), \mathcal{P}_2(l), B, C) \wedge$ $B \neq \mathcal{P}_2(l) \wedge A \neq B \wedge B \neq C$
c circle of center O passing through A	$true$
c circle whose diameter is A B	$collinear(\mathcal{O}(c), A, B) \wedge$ $\mathcal{O}(c)A = \mathcal{O}(c)B$

Table III. Semantic of reference for GeoProof

GeoProof Construction	**Predicate form (second)**
Free point	$true$
Point P on line l	$P \in l$
Point P on circle c	$P \in c$
I midpoint of A and B	$IA = IB \wedge collinear(I, A, B)$
I intersection of l_1 and l_2	$I \in l_1 \wedge I \in l_2 \wedge l_1 \not\parallel l_2$
I an intersection of c_1 and c_2	$I \in c_1 \wedge I \in c_2$ $O_1\, is\_center\, c_1 \wedge O_2\, is\_center\, c_2 \wedge$ $O_1 \in m_{O_1O_2} \wedge O_2 \in m_{O_1O_2} \wedge$ $\neg isotropic(m_{O_1O_2})$
I an intersection of c and l	$I \in c \wedge I \in l \wedge \neg isotropic(l)$
l passing through A and B	$A \neq B \wedge A \in l \wedge B \in l$
l parallel to m passing through A	$l \parallel m \wedge A \in l$
l perpendicular to m passing through A	$l \perp m \wedge A \in l$
l perpendicular bisector of A and B	$IA = IB \wedge collinear(I, A, B) \wedge I \in l \wedge$ $l \perp m_{AB} \wedge A \in m_{AB} \wedge B \in m_{AB}$
l bisector of the angle A,B,C	$eq\_angle(A, B, \mathcal{P}_2(l), \mathcal{P}_2(l), B, C) \wedge$ $B \neq \mathcal{P}_2(l) \wedge A \neq B \wedge B \neq C$
c circle of center O passing through A	$A \in c \wedge O\, is\_center\, c$
c circle whose diameter is A B	$collinear(O_c, A, B) \wedge O_cA = O_cB \wedge$ $O_c\, is\_center\, c \wedge A \in c$

l passing through A and B The formula defined in Table I and II is the following:

$$collinear(P, A, B) \wedge A \neq B$$

The formula defined in Table III is the following:

$$P \in l \wedge A \neq B \wedge A \in l \wedge B \in l$$

It can be shown that:

$$collinear(P, A, B) \wedge A \neq B \iff$$

$$\exists l, P \in l \wedge A \neq B \wedge A \in l \wedge B \in l$$

Hence the result.

l **parallel to** *m* **passing through** *A* The formula defined in Table I and II is the following:

$$collinear(P, A, P_{2l}) \wedge parallel(A, P_{2l}, P_1(m), P_2(m)) \wedge A \neq P_{2l}$$

The formula defined in Table III is the following:

$$P \in l \wedge l \parallel m \wedge A \in l$$

From $A \neq P_{2l}$ we know that there is an l such that $A \in l$ and $P_{2l} \in l$. From $collinear(P, A, P_{2l})$ we know that $P \in l$ (note that here we need the hypothesis $A \neq P_{2l}$).
In the other direction, we first construct a point P_{2l} different from A on l. It follows that $collinear(P, A, P_{2l})$ and hence we have $parallel(A, P_{2l}, P_1(m), P_2(m))$.

... The other cases are similar.

Point *P* **on circle** *c* We need to perform another case distinction on the way *c* has been constructed:

c **circle of center** *O* **passing through** *A* This case is a consequence of the equivalence:

$$OA = PA \iff \exists c, P \in c \wedge A \in c \wedge O\, is\_center\, c$$

c **circle whose diameter is** *AB* This case is a consequence of the equivalence:

$$O_cA = PO_c \wedge collinear(O_c, A, B) \wedge O_cA = O_cB \iff$$

$$\exists c, P \in c \wedge collinear(O_c, A, B) \wedge O_cA = O_cB \wedge$$

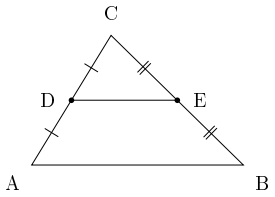
$$A \in c \wedge O_c\, is\_center\, c$$

I **midpoint of** *A* **and** *B* This case is trivial as the formulas for the midpoint are the same in both semantics.

... We do not detail here the other cases about intersection of lines and circles. They can be be shown by case distinction on the way the lines and the circles have been built.

3.1.3. *An example*
Let's take the midpoint theorem as an example, it states that:

Theorem 1. Let ABC be a triangle, and let D and E be the midpoints of AC and BC respectively. Then the line DE is parallel to the base AB.



The construction is translated into the following statement[5]:

```
(((((is_midpoint(D,C,A) /\ is_midpoint(E,C,B))/\
~C=A) /\ ~A=B) /\ ~B=C) /\ ~D=E) /\ ~A=B
```

The fact that $AB \parallel DE$ is then checked using the Gröbner basis method. During the proof process the user can work on his figure, if it takes too long the proof can be interrupted.

3.1.4. *Dealing with non-degeneracy conditions*
Non degeneracy conditions play a crucial role in formal geometry, this has been emphasized by most papers about formalization of geometry (Guilhot, 2005; Meikle and Fleuriot, 2003; Narboux, 2004). This translation is not an exception, we must be careful about the semantic of the generated statements. For this translation we have decided to consider GeoProof as a tool which permits to define a geometric *formula* and it does not build a *model* of this formula. The user can define "impossible" figures. For instance if we perform the following construction:
First, create two points A and B and then create the midpoint C of the segment $[AB]$ and the midpoint D of the segment $[BA]$. Finally, create the line passing through C and D. Then if we try to prove that $A = B$, GeoProof should answer "yes", as the hypotheses of the theorem are inconsistent (*ex falso quod libet*). This is consistent with logic but not with the user's intuition because the "impossible" objects are not displayed by GeoProof. This is why in fact GeoProof checks first if *false* can be proved, if this is the case it warns the user that its construction is impossible as shown on Figure 5.

[5] `~A=B` appears twice in this statement because both the line and the segment from A to B have been built.

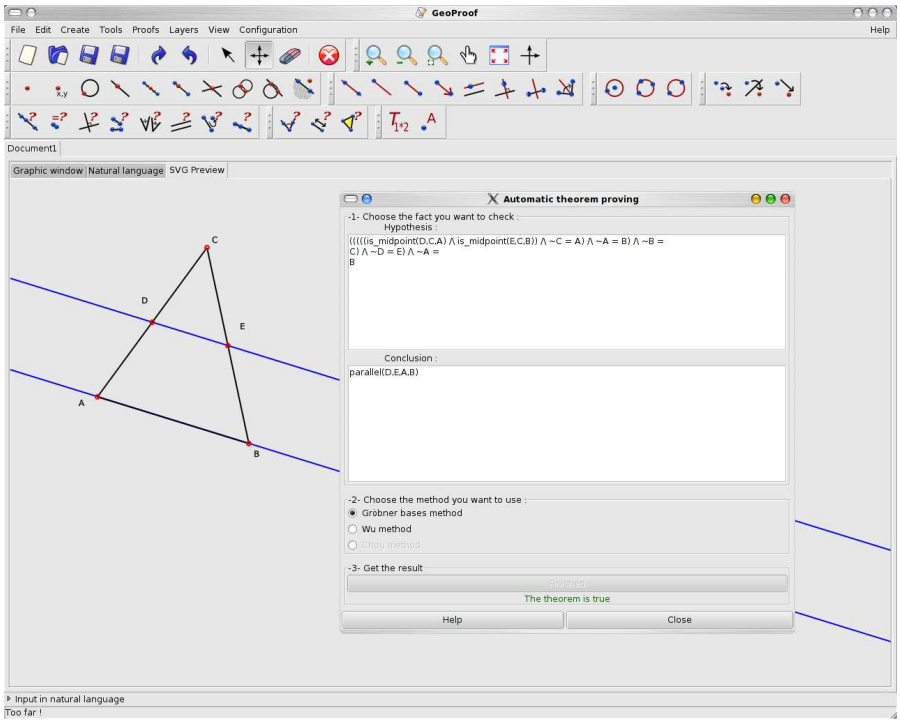


Figure 4. Checking the midpoint theorem using the embedded theorem prover.

3.2. Using Coq

In (Narboux, 2004) we have described the implementation of Chou, Gao and Zhang's decision procedure for affine geometry in the Coq proof assistant. This development provides a very high level of confidence as the proofs produced by our tactic are checked by the Coq kernel. This required the formal proofs of all the theorems needed to prove the correctness of the decision procedure. Our formalization has allowed to fix some non-degeneracy conditions in the statements of some lemmas. Moreover, as the logic behind Coq is intuitionist, this work has also permitted to clarify what are the classical reasoning steps which are used in the decision procedure. More information is also available in french in (Narboux, 2006a).

Here we want to export a construction built using GeoProof into a statement in the language of the Coq development. Our implementation of Chou, Gao and Zhang's decision procedure is restricted to affine plane geometry. Hence in GeoProof the tools which do not have any corresponding concept in the Coq implementation are grayed out. The Coq development is based on the axiom system shown on Table IV.

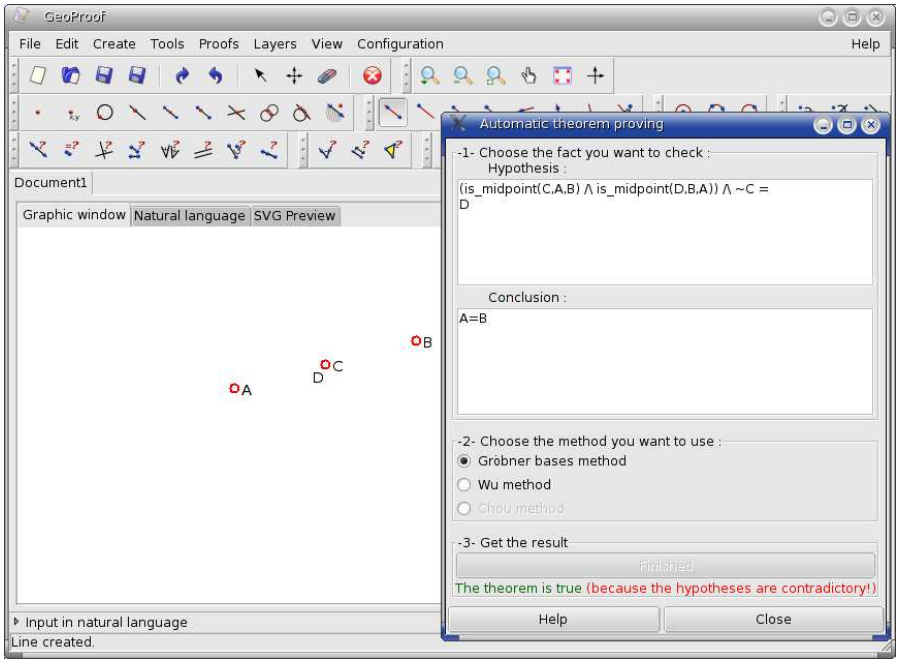


Figure 5. Trying to prove a property with contradictory hypotheses.

This axiom system is based on two geometric quantities. The signed area of a triangle ($\mathcal{S}_{ABC}$) and the ratio of two oriented distances ($\frac{\overline{AB}}{\overline{CD}}$). To ease the Coq formalization, this axiom system has been slightly modified compared to the axiom system found in (Chou et al., 1994). In the original axiom system the ratio of two oriented distances $\frac{\overline{AB}}{\overline{CD}}$ is defined only when AB is parallel to CD. Here we do not put this restriction at the axiom system level but only when we state theorems involving ratios. It is clear that this axiom system is based on points. Hence we have to perform a translation similar to those described in the last section. Table V gives the translation of some common geometric notions in the language of the axiom system. Figure 3.2 shows the translation of the statement corresponding to the midpoint theorem in the syntax of Coq.

Table IV. The Chou axiom system (slightly modified for the formalization in Coq).

Points	Point : Set
Field	F is a field $2 \neq 0$
Signed distance	$\overline{\cdot} : \text{Point} \to \text{Point} \to F$ $\overline{AB} = 0 \iff A = B$
Signed area	$\mathcal{S} : \text{Point} \to \text{Point} \to \text{Point} \to F$ $\mathcal{S}_{ABC} = \mathcal{S}_{CAB}$ $\mathcal{S}_{ABC} = -\mathcal{S}_{BAC}$
Chasles'axiom	$\mathcal{S}_{ABC} = 0 \to \overline{AB} + \overline{BC} = \overline{AC}$
Dimension	$\exists A, B, C : \text{Point}, \mathcal{S}_{ABC} \neq 0$ $\mathcal{S}_{ABC} = \mathcal{S}_{DBC} + \mathcal{S}_{ADC} + \mathcal{S}_{ABD}$
Construction	$\forall r : F\ \exists P : Point, \mathcal{S}_{ABP} = 0 \wedge \overline{AP} = r\overline{AB}$ $A \neq B \wedge \mathcal{S}_{ABP} = 0 \wedge \overline{AP} = r\overline{AB} \wedge \mathcal{S}_{ABP'} = 0 \wedge \overline{AP'} = r\overline{AB} \to P = P'$
Proportions	$A \neq C \to \mathcal{S}_{PAC} \neq 0 \to \mathcal{S}_{ABC} = 0 \to \frac{\overline{AB}}{\overline{AC}} = \frac{\mathcal{S}_{PAB}}{\mathcal{S}_{PAC}}$

Table V. Expressing some common geometric notions using $\mathcal{S}$ and ratios

Geometric notions	**Formalization**
A,B and C are collinear	$\mathcal{S}_{ABC} = 0$
$AB \parallel CD$	$\mathcal{S}_{ABC} = \mathcal{S}_{ABD}$
I is the midpoint of AB	$\frac{\overline{AB}}{\overline{AI}} = 2 \wedge \mathcal{S}_{ABI} = 0$

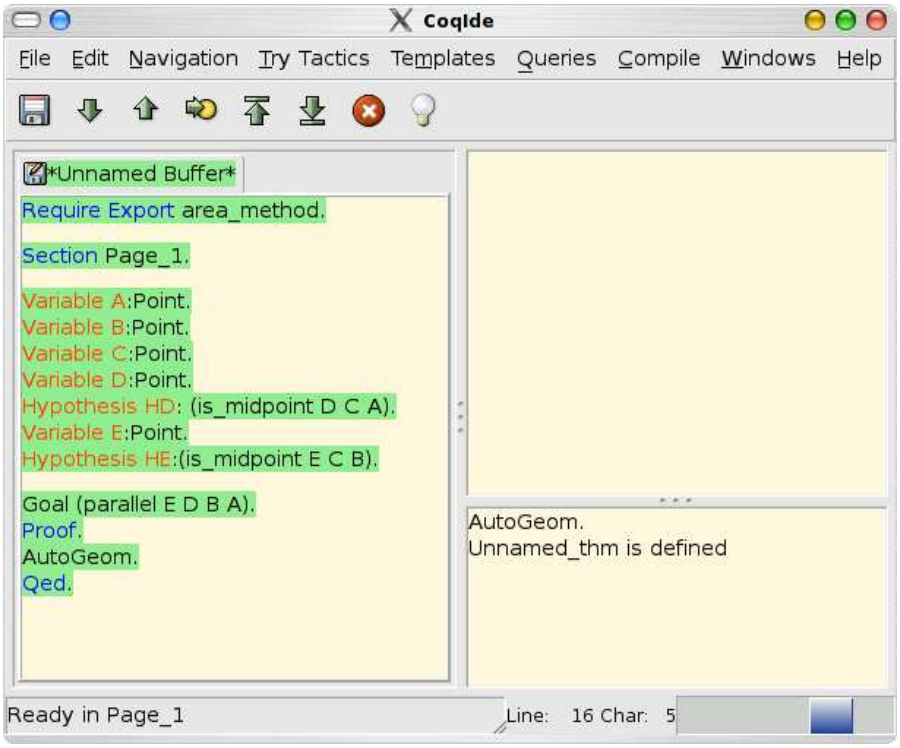
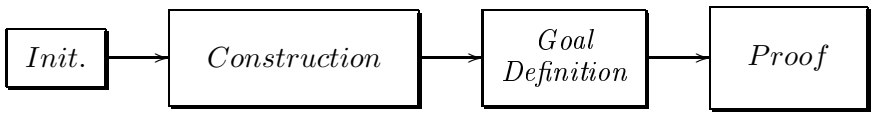


Figure 6. The midpoint theorem, expressed in the Coq language for Chou decision procedure.

4. Interactive input

In this section we describe the interactive proof mode of GeoProof. Thanks to the configuration menu, the user can choose between three interactive modes, the first one uses the language described in section 3.2 and the second one uses the language of the Coq development for high school geometry by Frédérique Guilhot (Guilhot, 2005) and the third one use the language of our formalization of the geometry of Tarski (Narboux, 2006c). In the first mode the user can deal with affine plane geometry and in the two other modes with Euclidean plane geometry. The interaction with Coq is performed through the CoqIDE user interface. GeoProof communicates with CoqIDE[6] via a private clipboard. We have started by implementing the translation from a GeoProof construction to a Coq statement. We perform the same translation as in (Bertot et al., 2003) except that it is in the reverse direction (here we translate *to* Coq)[7].

The interactive mode of GeoProof is decomposed into four steps:



In the initialization phase, the communication between CoqIDE and GeoProof is started. Depending on the used language some construction tools which can not be exported to Coq are grayed out in GeoProof. The Coq definitions corresponding to the used are language loaded using the Coq command `Require`. A new section is opened. If the user had already constructed some objects before starting the interactive proof mode, these objects are now exported to Coq. Objects which do not have any meaning in the selected language are ignored.

In the construction phase the objects created by the user are added in the Coq context with their corresponding assumptions. In the example shown[8] in Figure 9 this corresponds to the `Variable` and `Hypothesis` commands.

In the goal phase the user needs to define what he wants to prove. In the context of education this phase can be presented as an exercise consisting in finding an interesting conjecture about the figure. For that purpose GeoProof provides several features:

[6] This feature requires CoqIDE version 8.1 or later.

[7] In the future we should merge our developments to allow communication in both directions, this requires a more complex communication system as explained in the future work section.

[8] The predicates names are in French because this development is focused on the French high-school curriculum

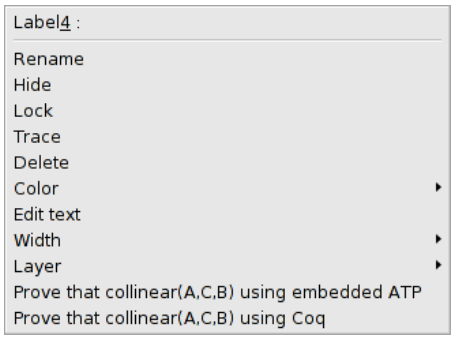


Figure 7. The contextual menu associated to a dynamic label.

1. The user can move the free points of the figure to guess the invariants.

2. When the user has guessed a conjecture, he can make a first experiment to check the conjecture by building a dynamic label to perform measures on the figure as described in section 2.2. Then if he wants to prove the fact represented by the label, he can right click on the label and choose the corresponding menu entry. Figure 7 shows the contextual menu of a dynamic label.

3. To invent a conjecture about the locus of a point i.e. the path traced out by a moving point under given geometrical conditions, the user can take advantage of the trace option. When this option is activated for an object, this object leaves a trace behind him. For instance the locus of a point, which is equidistant from two fixed points, is the perpendicular bisector of the straight line joining the two fixed points.

In the proof phase the user proves his statement within CoqIDE. Hence, the current implementation of GeoProof requires to know how to use Coq. This will be improved in future versions by adding some features to allow the application of theorems within GeoProof.
If during the proof a new object needs to be created, the user can do it using GeoProof. Indeed when a new object is added in GeoProof a Coq tactic is pasted into CoqIDE. This tactic applies the theorem which proves the existence of the object which has just been created and introduce in the context the knowledge about this new object. In some cases this generates non-degeneracy conditions which need to be proved by the user. Figure 8 shows the command (defined in Ltac - the tactic language of Coq) which is used when the user creates a point I at the intersection of two lines AB and CD.

```
Ltac DecompEx H P := elim H;intro P;intro;clear H.

Ltac let_intersection I A B C D :=
let id1 := fresh in ((assert (id1:exists I,
I =  pt_intersection (line A B) (line C D));
[apply (existence_pt_intersection)|DecompEx id1 I])).
```

Figure 8. The tactic to prove the existence of the point of intersection.

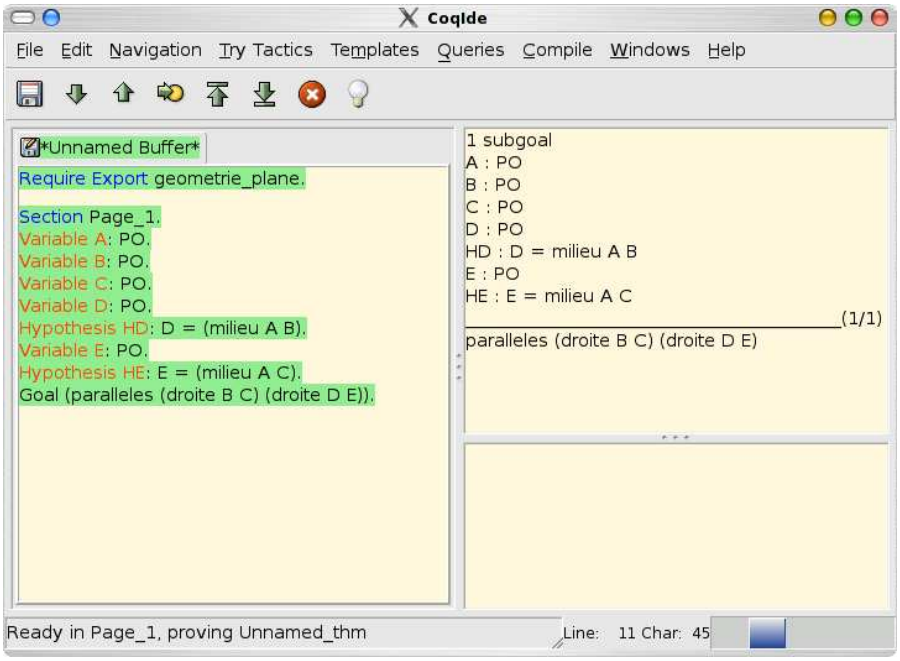


Figure 9. The midpoint theorem in the language used by Frédérique Guilhot's Coq development.

If the user deletes an object in GeoProof it is removed from the Coq context thanks to the `clear` command of Coq. If the user wants to delete some object without deleting it in Coq, he can hide the object in GeoProof.

5. Future Work

The current prototype of GeoProof uses a private clipboard[9] as a communication pipe between GeoProof and the Coq Interactive Develop-

[9] Technically, we use a feature provided by GTK: we create a clipboard identified by a name (here "`GeoProof`") which is different from the standard clipboard.

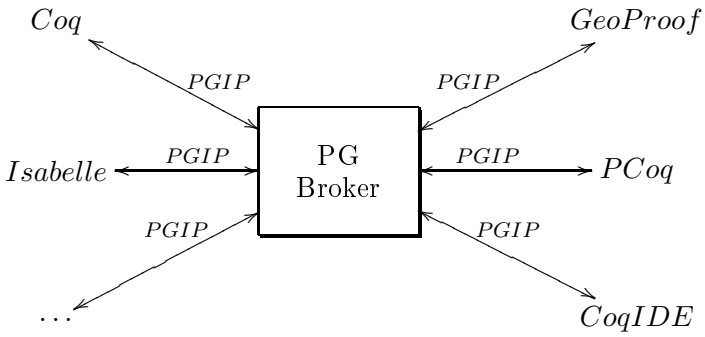


Figure 10. Integrating GeoProof in the proof general infrastructure

ment Environment. This approach has the advantage to be both easy to implement and easy to use. The user can start the interaction without any configuration step, he just needs to launch GeoProof and CoqIDE on the same computer. But this infrastructure has some limitations. First, the communication with Coq is done using the Coq syntax, which is easy to produce but hard to parse. Second, the synchronization between what is typed in CoqIDE and the input generated by GeoProof is not ensured. A better infrastructure for the communication between Coq and GeoProof would be to use the Proof General Interaction Protocol (PGIP) framework (Winterstein et al., 2004; Aspinall et al., 2004). This framework is based on XML and allows to have several interfaces interacting at the same time with one proof assistant. This is exactly what we need because as mentioned before, some proofs are easier to grasp diagrammatically and some are better presented the classic way (proofs using complex numbers for instance). In our example, GeoProof and CoqIDE would interact with the Coq proof assistant. But this could be generalized to other proof assistants and graphical user interfaces such as Isabelle, Eclipse/Proof General and PCoq as shown in Figure 10. This approach would require implementation of PGIP within Coq, CoqIDE and GeoProof.

The proving features of GeoProof in itself should also be extended. We need to add the possibility to apply a theorem graphically by drag and drop and to mark facts on the diagram to produce new assertions in Coq. We could also transform macro constructions into proof of existence of geometric objects verifying some properties.

Another planned extension of GeoProof is to adapt it to deal with diagrammatic proofs in abstract term rewriting (see the first chapter of (Baader and Nipkow, 1998)). We have formalized in (Narboux, 2006b) the kind of diagrams which are usually found in the rewriting literature. The next step is to implement this formalization in GeoProof to provide

a high level input language for proofs in abstract rewriting. The design presented in this paper can be adapted to abstract term rewriting.

We are also aiming at pseudo-diagrammatic proofs in euclidean geometry. Because of degenerated cases (impossible figures) we think that a fully diagrammatic and *intuitive* notation for euclidean geometry is hard to obtain. We believe that the solution consists in using a mixed approach which is diagrammatic or textual depending on the context.

6. Conclusion

Proving is a crucial aspect of mathematics and hence must have a prominent role in the education. The most widely used software in the teaching of mathematics are mainly used to explore, visualize, calculate, find counter examples, conjectures, or check facts, but most of them can not be used to build a proof in itself. We believe that proof assistants should be adapted to fulfill this need.

We have presented in the paper a prototype which aims at integrating dynamic geometry, automatic theorem proving and formal proof. This should be considered as a first step toward the use of a proof assistant in the classroom.

Availability

GeoProof is available at: `http://home.gna.org/geoproof/`

Acknowledgements

I want to thank Hugo Herbelin for his help during the elaboration of this work and Frédérique Guilhot for her comments and the formal proofs she has added to her development for GeoProof.

References

Ag-Almouloud: 1992, 'L'ordinateur, outil d'aide à l'apprentissage de la démonstration et de traitement de données didactiques'. Ph.D. thesis, Université de Rennes.

Amerkad, A., Y. Bertot, L. Pottier, and L. Rideau: 2001, 'Mathematics and Proof Presentation in Pcoq'. In: *Workshop Proof Transformation and Presentation and Proof Complexities in connection with IJCAR 2001.*

Anderson, J. R., C. F. Boyle, and G. Yost: 1985, 'The geometry Tutor'. In: *IJCAI Proceedings.* pp. 1–7.

Aspinall, D., C. Lüth, and D. Winterstein: 2004, 'A Framework for Interactive Proof'. Technical report.

Baader, F. and T. Nipkow: 1998, *Term rewriting and all that.* New York, USA: Cambridge University Press.

Balacheff, N., S. Pesty, M. Occello, R. Caffera, C. Webber, and N. Peltier: 1999-2002, 'Baghera'. `http://www-baghera.imag.fr`.

Barwise, J. and G. Allwein (eds.): 1996, *Logical Reasoning with Diagrams.* Oxford University Press.

Bernat, P.: 1993, 'CHYPRE: Un logiciel d'aide au raisonnement'. Technical Report 10, IREM.

Bertot, Y., F. Guilhot, and L. Pottier: 2003, 'Visualizing Geometrical Statements with GeoView'. *Electronic Notes in Theoretical Computer Science* **103**, 49–65.

Bertot, Y. and L. Thery: 1998, 'A generic approach to building user interfaces for theorem provers'. *The Journal of Symbolic Computation* **25**, 161–194.

Chou, S.-C.: 1988, *Mechanical Geometry Theorem Proving.* D. Reidel Publishing Company.

Chou, S.-C. and X.-S. Gao: 1992, 'A class of geometry statements of constructive type and geometry theorem proving'. In: *Proceeding of CADE 92.*

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

Chou, S.-C., X.-S. Gao, and J.-Z. Zhang: 1996, 'Automated Generation of readable proofs with geometric invariants, Theorem Proving with Full Angle'. *Journal of Automated Reasoning* **17**, 325–347.

Coq development team, The: 2004, 'The Coq proof assistant reference manual, Version 8.0'. LogiCal Project.

Furinghetti, F. and P. Domingo: 2003, 'To produce conjectures and to prove them within a dynamic geometry environment: a case study'. In: *Proceeding of Psychology of Mathematics 27th international Conference.* pp. 397–404.

Gao, X.-S.: 2000, 'Geometry Expert, Software Package'. `http://www.mmrc.iss.ac.cn/~xgao/gex.html`.

Gao, X.-S. and Q. Lin: 2002, 'MMP/Geometer - a software package for automated geometry reasoning'. In: F. Winkler (ed.): *Proceedings of ADG 2002.* pp. 44–46.

Gressier, J.: 1988-1998, 'Geometrix'. `http://perso.wanadoo.fr/jgressier/ENGLISH/english.html`.

Guilhot, F.: 2005, 'Formalisation en Coq et visualisation d'un cours de géométrie pour le lycée'. *Revue des Sciences et Technologies de l'Information, Technique et Science Informatiques, Langages applicatifs* **24**, 1113–1138. Lavoisier.

Harrison, J.: 2003, 'Automatic Theorem Proving Examples'. `http://www.cl.cam.ac.uk/users/jrh/atp/index.html`.

Jackiw, N.: 1990, 'The Geometer's Sketchpad'. `http://www.keypress.com/`.

Jamnik, M.: 2001, *Mathematical Reasoning with Diagrams: From Intuition to Automation*. CSLI Press.

Kortenkamp, U.: 1999, 'Foundations of Dynamic Geometry'. Ph.D. thesis, ETH Zürich.

Kortenkamp, U. and J. Richter-Gebert: 2004, 'Using automatic theorem proving to improve the usability of geometry software'. In: *Mathematical User Interface*.

Laborde, J.-M. and F. Bellemain: 1993-1998, 'Cabri-Geometry II'. `http://www.cabri.net`.

Luengo, V.: 1997, 'Cabri-Euclide: Un micromonde de Preuve intégrant la réfutation'. Ph.D. thesis, Université Joseph Fourier.

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

Miller, N.: 2001, 'A diagrammatic formal system for Euclidean geometry'. Ph.D. thesis, Cornell University.

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

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

Narboux, J.: 2006b, 'A formalization of diagrammatic proofs in abstract rewriting'.

Narboux, J.: 2006c, 'Mechanical Theorem Proving in Tarski's geometry'. In: *Proceedings of Automatic Deduction in Geometry 06*.

Narboux, J.: 2006d, 'The user manual of GeoProof'. `http://home.gna.org/geoproof/documentation.html`.

Py, D.: 1990, 'Reconnaissance de plan pour l'aide à la démonstration dans un tuteur intelligent de la géométrie'. Ph.D. thesis, Université de Rennes.

Richter-Gebert, J. and U. Kortenkamp: 1999, 'Die interaktive Geometrie software Cinderella Book and CD-ROM'. German school-edition of the Cinderella software. `http://cinderella.de`.

Schwartz, J. T.: 1979, 'Probabilistic algorithms for verification of polynomial identities'. In: *Symbolic and algebraic computation*, Vol. 72 of *Lecture Notes in Computer Science*. Marseille, pp. 200–215.

Wilson, S. and J. Fleuriot: 2005, 'Combining Dynamic Geometry, Automated Geometry Theorem Proving and Diagrammatic Proofs'. In: *ETAPS Satellite Workshop on User Interfaces for Theorem Provers (UITP)*. Edinburgh.

Winterstein, D.: 2004a, 'Dr.Doodle: A Diagrammatic Theorem Prover'. In: *Proceedings of IJCAR 2004*.

Winterstein, D.: 2004b, 'Using Diagrammatic Reasoning for Theorem Proving in Continous Domain'. Ph.D. thesis, The University of Edinburgh.

Winterstein, D., D. Aspinall, and C. Lüth: 2004, 'PG/Eclipse: A Generic Interface for Interactive Proof'. Technical report.

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

Yevdokimov, O.: 2004, 'About a constructivist approach for stimulating students' thinking to produce conjectures and their proving in active learning of geometry'. In: *Fourth Congress of the European Society for Research in Mathematics Education*.