Integrating Dynamic Geometry Software, Deduction Systems, and Theorem Repositories

Pedro Quaresma[1⋆] and Predrag Janičić[2⋆⋆]

[1] Department of Mathematics, University of Coimbra
3001-454 Coimbra, Portugal, pedro@mat.uc.pt
[2] Faculty of Mathematics, University of Belgrade
Studentski trg 16, 11000 Belgrade, Serbia & Montenegro, janicic@matf.bg.ac.yu

Abstract. The axiomatic presentation of geometry fills the gap between formal logic and our spatial intuition. The study of geometry is, and will always be, very important for a mathematical practitioner. GCLCprover, an automatic theorem prover (ATP) integrated with dynamic geometry software (DGS) gives its user a tool to bridge his/her spatial intuition with formal, Euclidean geometry proofs. GeoThms, a system consisting of the mentioned programs and a database geoDB, provides a framework for exploring geometrical knowledge. A GeoThms user can browse through a list of available geometric problems, their statements, illustrations, and proofs. He/she can also interactively produce new geometrical constructions, theorems, and proofs and add new results to the existing ones. GeoThms framework provides an environment suitable for new ways of studying and teaching geometry at different levels. GeoThms also provides a system for storing mathematical knowledge (in a explicit, declarative form) — not only theorem statements, but also their (automatically generated) proofs and corresponding illustrations.

1 Introduction

The axiomatic presentation of geometry fills the gap between formal logic and our spatial intuition. The study of geometry is, and will always be, very important for a mathematical practitioner. Geometry and geometrical proofs always were, and still are, exemplary mathematical contents. In history they often served for guiding development of foundations of mathematics, and today they serve in mathematical education, aimed at acquiring mathematical rigour. Computer technologies give new ways for dealing with geometry: they are used for visualisation of geometrical objects, but also for exploring/testing geometrical conjectures and, finally, for automated proving of geometrical theorems. Integrating these ways of dealing with geometry brings new forms in communicating

⋆ This work was partially supported by programme POSC.
⋆⋆ This work was partially supported by the programme POSC, by the Centro International de Matemática (CIM), under the programme "Research in Pairs", while visiting Coimbra University under the Coimbra Group Hospitality Scheme. Also, partially supported by Serbian Ministry of Science and Technology grant 144030.

mathematical (geometrical, in this case) information — theorems, figures, and proofs. In this way, the deductive nature of geometrical conjectures and proofs is linked to the semantic nature of models of geometry and also, to human intuition and to geometrical visualisations. In order to explore such mathematical knowledge, a framework where one can browse through known results and seek for new ones is needed. In this paper, we present a tightly integrated framework that we developed, consisting of a repository of constructive geometry theorems (and proofs), a geometry theorem prover, and dynamic geometry software (as final applications). This complex framework provides an environment suitable for new ways of studying and teaching geometry at different levels and bridging spatial intuition with formal, axiomatic, Euclidean geometry proofs. The user can browse through a list of geometric problems, their statements, illustration, and proofs. He/she can also interactively use geometry software (GCLC, or Eukleides), to describe new geometric constructions (and corresponding figures), and GCLCprover to (try to) prove new conjectures, adding new results to the existing ones. In addition, this framework provides an environment for storing mathematical knowledge (in explicit, declarative way) — about geometrical constructions, proofs, and illustrations (in this context, geometrical illustrations are not stored as images, but as their formal, explicit descriptions; while mathematical illustrations may carry information, the original message cannot always be reproduced from the illustration itself; mathematical/geometrical images stored via formal descriptions are easy to maintain, understand, modify, and process in different ways — including for producing images.)

In this paper we present our framework consisting of dynamic geometry software, automated theorem provers, and the repository of constructive geometry conjectures. All constructions and conjectures are stored in formal, declarative representation that can be used as a description of a construction, a description of a figure, and also as a formal description of a conjecture that can be attempted to be proved by the developed theorem prover.

Paper overview. Section 2, briefly discusses geometric constructions, the domain of our integrated framework; Section 3 talks about parts of our framework, with §3.1 about dynamic geometry software, especially GCLC and Eukleides, §3.2 about automated theorem proving in geometry and especially the prover GCLCprover, based on the area method, and §3.3 about geoDB, a repository of constructive geometric theorems and proofs. Section 4 is about our integrated geometry framework and its features, and Section 5 presents the whole system through a step-by-step example. Section 7 discusses further work and the issue of standards for mathematical (geometrical, in this case) contents; Section 8 draws final conclusions.

2 Geometry Constructions

For hundreds, or even thousands, of years geometric construction problems have been one of the most attractive parts of geometry and mathematics. A geometric

construction is a sequence of specific, primitive construction steps. These primitive construction steps (also called *elementary constructions*) are based on using a *ruler* (or a *straightedge*[3]) and a *compass*, and they are:

- construction (with a *ruler*) of a line such that two given points belong to it;
- construction of a point which is an intersection of two lines (if such a point exists);
- construction (with a *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 segment connecting two points;
- construction of intersections between a given line and a given circle (if such points exist).

By using the set of primitive constructions, one can define more complex constructions (e.g., the construction of a right angle, a construction of the midpoint of a line segment, etc.).

Abstract (i.e., formal, axiomatic) nature of geometric objects have to be distinguished from their usual interpretations. A geometric construction is a procedure consisting of abstract steps and it is not a picture, but for each construction there is its counterpart in the standard Cartesian model.

Construction problems are often studied (in schools and universities) because they require rigour, but are in the same time intuitive (since they require effective procedures and since the level of abstraction is higher than the level of geometry axioms). The study of geometry and construction problems also represents a suitable field for interactive teaching supported by software tools.

3 Building Blocks

In this section, we present the building blocks of our geometry framework.

3.1 Dynamic Geometry Software, GCLC and Eukleides

Dynamic geometry software (e.g., *Cinderella*, *Geometer's Sketchpad*, *Cabri*[4]) visualise geometric objects and link formal, axiomatic nature of geometry (most often — Euclidean) with its standard models (e.g., Cartesian model) and corresponding illustrations. The common experience is that dynamic geometry software significantly help students to acquire knowledge about geometric objects.

GCLC [6, 8] is a tool for teaching and studying mathematics, especially geometry and geometric constructions, and also for storing descriptions of mathemat-

[3] The term "straightedge" is sometimes used instead of "ruler" in order to emphasise there are no markings which could be used to make measurements.

[4] See `http://www.cinderella.de`, `http://www.keypress.com/sketchpad/`, `http://www.cabri.com`

ical figures and producing digital illustrations of high quality.[5] GCLC provides support for a range of geometric constructions and isometric transformations. Although its primary initial goal is describing formal geometric constructions, GCLC also provides a support for some non-constructible objects too. In GCLC there is also support for symbolic expressions, second order curves, parametric curves, while-loops, etc. Thus, GCLC is more than a geometry tool.

GCLC is based on the idea that constructions are formal procedures, rather than drawings. Thus, in GCLC, producing mathematical illustrations is based on "describing figures" rather than of "drawing figures" (in a sense, this system is in spirit close to the LaTeX system [10], with its logical design of texts). All mathematical figures (not only geometric ones) are described in this spirit, in GC language. These descriptions directly reflect meaning of mathematical objects to be presented, and are easily understandable to mathematicians. In that sense, this language is more a high-level language than a script language.

WinGCLC is the Windows version of GCLC, with a rich graphical interface and provides a range of additional functionalities to GCLC. It supports interactive work, animations, traces, "watch window" for monitoring values of selected objects ("geometry calculator") etc. [8].

Eukleides[6] [14, 16] is an Euclidean geometry drawing language. Two programs are related to it. First, `eukleides`, a compiler for typesetting geometric figures within a (La)TeX document. It can also convert such figures to EPS format or to various other vector graphic formats. Second, `xeukleides`, a GUI front-end for creating interactive geometric figures. This program can also be used for editing and tuning Eukleides code. Eukleides, like GCLC has been designed to be close to the traditional language of elementary Euclidean geometry. In many cases, it is possible to completely avoid the use of Cartesian coordinates.

We have developed a tool `euktogclcprover`, that converts Eukleides files to GCLCprover files, enabling the prover to be used with geometric constructions described within Eukleides.

3.2 Automated Theorem Proving in Geometry and GCLCprover

Automated theorem proving in geometry has two major lines of research: synthetic proof style and algebraic proof style (see, for instance, [12] for a survey). Algebraic proof style methods are based on reducing geometry properties to algebraic properties expressed in terms of Cartesian coordinates. These methods

[5] GCLC package is freely available from `www.matf.bg.ac.yu/~janicic/gclc/`. The mirrored version is available from EMIS (The European Mathematical Information Service) `www.emis.de/misc/index.html`. There are command-line version and graphic interface versions of GCLC for Windows, while there is only a command-line version of GCLC for Linux.

[6] Eukleides is available from `http://www.eukleides.org`, There are versions for a number of languages. The first author of this paper is responsible for the Portuguese version of Eukleides: EukleidesPT is available from `http://gentzen.mat.uc.pt/~EukleidesPT/`

are usually very efficient, but the proofs they produce do not reflect the geometry nature of the problem and they give only a yes/no conclusion. Synthetic methods attempt to automate traditional geometry proof methods that produce human-readable proofs.

We have extended GCLC, with a theorem prover that allows formal deductive reasoning about constructions made in the (main) drawing module. The built-in prover, GCLCprover, is based on the area method [3, 4, 13]. It produces proofs that are human-readable, and with a clear justification for every proof step. The prover can be used in conjunction with other dynamic geometry software, which demonstrate the flexibility of the developed deduction module.

The area method is a synthetic method providing traditional (not coordinate-based), human-readable proofs. The proofs are expressed in terms of higher-level geometric lemmas and expression simplifications. The main idea of the method is to express hypotheses of a theorem using a set of constructive statements, each of them introducing a new point, and to express a conclusion by an equality of expressions in some geometric quantities (e.g., signed area of a triangle), without referring to Cartesian coordinates. The proof is then based on eliminating (in reverse order) the points introduced before, using for that purpose a set of appropriate lemmas. After eliminating all introduced points, the current goal becomes an equality between two expressions in quantities over independent points. If it is trivially true, then the original conjecture was proved valid, if it is trivially false, then the conjecture was proved invalid, otherwise, the conjecture has been neither proved nor disproved. In all stages, different simplifications are applied to the current goal. The method does not have any branching, which makes it very efficient for many non-trivial geometry theorems. The method can transform a conjecture given as a geometry quantity of degree d, involving n constructed points, to a rational expression not involving constructed points, and with a degree at most $5d3^{5n}$ [3].

The area method is applicable to a wide range of constructions and a wide range of geometric conjectures. For this fragment of geometry, the area method gives a decision procedure: a terminating, sound, and complete procedure, i.e., a procedure that can prove any geometry theorem involving only points introduced by using supported constructions, and expressed in terms of geometric quantities. For details of the method, correctness proofs for all simplification steps, and for details about our implementation see [15].

GCLCprover is tightly integrated with geometry software. This means that one can use the prover to reason about a (say) GCLC construction (i.e., about objects introduced in it), without changing and adapting it for the deduction process — the user only needs to add the conclusion he/she wants to prove. The geometric constructions made within GCLC are internally transformed into primitive constructions of the area method, and in some cases, some auxiliary points are introduced.

GCLCprover was implemented in C++ (as GCLC) and is very efficient. The theorem prover produces proofs in LaTeX form and a report about the proving process: whether the conjecture was proved or disproved, CPU time spent, and

number of proof steps performed. For each proof step, there is a justification, and (optionally) its semantics counterpart (the semantic information is not used in the proof itself, but it can be used for testing conjectures). The prover can prove many complex geometric problems in milliseconds, producing readable proofs.

3.3 The geoDB database

The geoDB database gives support to the other programs, keeping the information, and allowing for its fast retrieving whenever necessary. The database is organised in the following form (see the entity-relationship diagram for details – Figure 1):

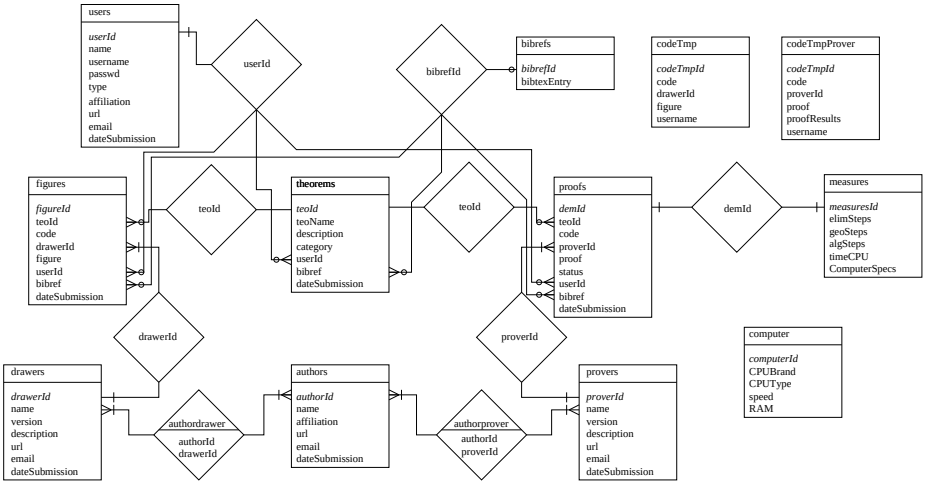

Fig. 1. geoDB — Entity-relationship diagram

theorems — statements of theorems, in natural-language form, formatted in LaTeX;

figures — descriptions of geometrical constructions, in DGS's code (GCLC, Eukleides, or other drawing software), they can be used for producing the corresponding figures;

proofs — geometrical constructions with conjectures in ATP's code (GCLCprover, or other provers), they are used for producing the corresponding proofs;

A geometric theorem can have different figures and/or proofs, made by different software, made by different users. This fact is expressed by the 1 to n relationships between the entities "theorems" and the other two entities (see Figure 1).

The database also has the following auxiliary entities:

bibrefs — bibliographic references, in BIBTEX format;
drawers & provers — information about the programs whose code is kept in the database, and with which the user can interact;
authors — information about the authors of the programs;
users — information about registered users.
computer — information about the computer used as the test bench.

The `codeTmp` and `codeTmpProver` tables are used to store temporary information, deleted after each session, for the interactive section of GeoThms.

The geoDB database is implemented in MySQL, with InnoDB transition safe type of tables, and with foreign key constraints.

4 The Framework

GeoThms[7], is a framework that links dynamic geometry software (GCLC, Eukleides), geometry theorem provers (GCLCprover), and a repository of geometry problems (geoDB) (see Figure 2).

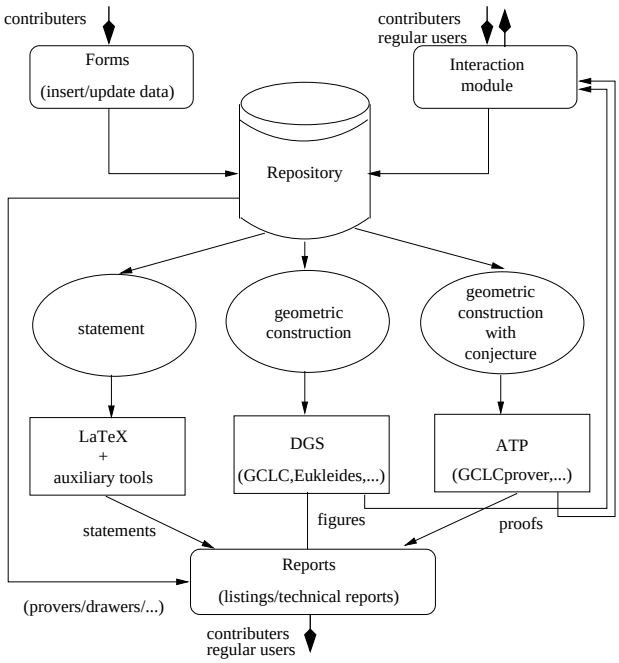

Fig. 2. The GeoThms framework

GeoThms provides a Web workbench in the field of constructive problems in Euclidean geometry. Its tight integration with dynamic geometry software

[7] GeoThms is accessible from `http://hilbert.mat.uc.pt/~geothms`

and automatic theorem provers (GCLC, Eukleides, and GCLCprover, for the moment) and its repository of theorems, figures and proofs, give the user the possibility to easily browse through the list of geometric problems, their statements, illustrations and proofs, and also to interactively use the drawing and proving programs (See Figure 3).

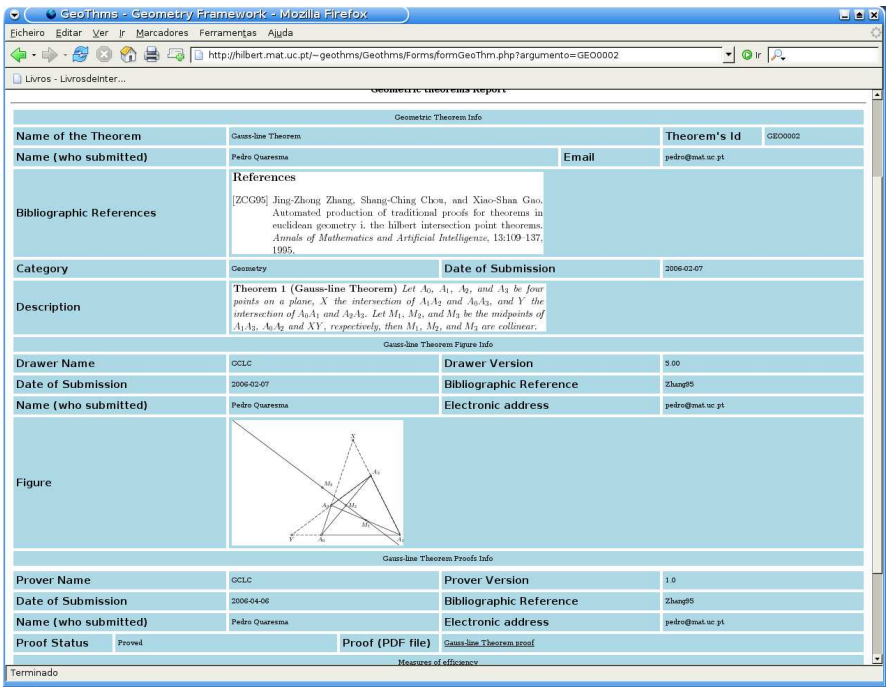

Fig. 3. GeoThms screenshot - Theorem Report

The structure of the web interface has two main levels of interaction (see Figure 4). The entry level, accessible to all web-users, has some basic information about GeoThms, including documents about the GeoThms Framework, and about the GCLCprover and the Area Method. This level offers the possibility of registration to anyone interested in using GeoThms, and it gives access to the other levels. A (registered) regular user has access to a second level where he/she can browse the data from the database (in a formatted, or in a plain textual form) and use the drawing/proof programs in an interactive way.

A regular user can apply to the status of *contributer* in which case he/she will have the possibility to insert new data, and/or to update the data he/she had inserted previously.

Constructions are described and stored in declarative languages of dynamic geometry software such as GCLC and Eukleides. Figures are generated directly on the basis of descriptions of constructions, by GCLC and Eukleides and stored

as JPEG files. Conjectures are described and stored in a a form that extends descriptions of constructions. Descriptions of conjectures is used (directly or via a converter) by GCLCprover. Proofs are generated by GCLCprover and stored as PDF files (after beeing processed by LaTeX, using a specific layout, `gclc_proof` style).

The framework can be simply augmented by other dynamic geometry software and other geometry theorem provers.

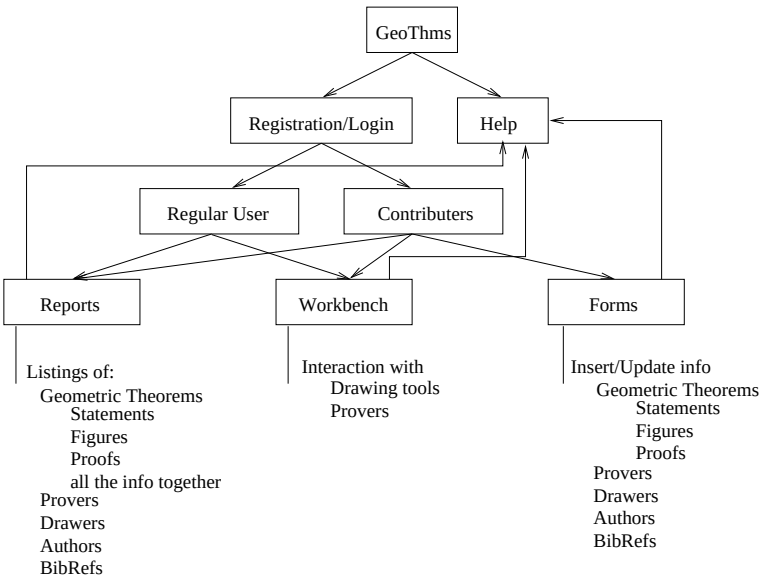

Fig. 4. GeoThms — Web Interface

GeoThms gives the user a complex framework suitable for new ways of communicating mathematical (geometrical, in this case) knowledge. It provides an open system where one can learn from the existing knowledge base and seek for new results. GeoThms also provides a system for storing mathematical knowledge (in a strict, declarative form) — not only theorem statements, but also their (automatically generated) proofs and corresponding figures, i.e., visualisations.

5 Geothms by Example

In this section we describe GeoThms framework through a step-by-step example.

The circumcircle of a triangle is the unique circle on which all its three vertices lie. Its center can be constructed as the intersection of any two out of the three perpendicular side bisectors. The crucial point is: do the three perpendicular side bisectors meet in a single point?

We can use GeoThms to answer this question, by describing the construction and proving the property. Using the interactive part of GeoThms, a user can begin by the construction, proceed attempting to prove the conjecture and, if all went as expected, insert all this information, along with the new result statement, in the database.

5.1 Describing the Construction

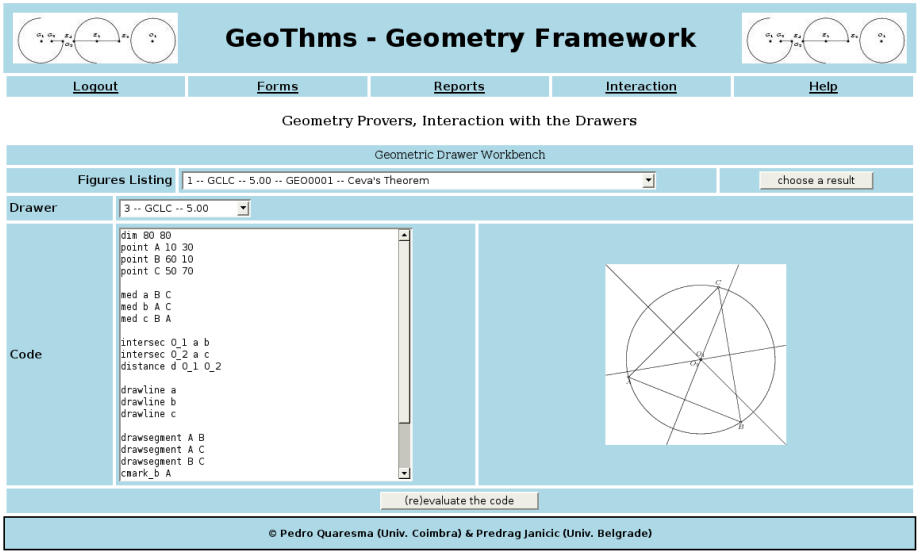

Fig. 5. Circumcircle of a triangle — Interaction with the DGS

The constructive specification of the figure has to define: three points A, B, C (the vertices of the triangle); three side bisectors; points O_1 and O_2 defined as the pairwise intersections of these lines. Apart from the construction steps, the figure description also provides the coordinates of the points A, B, and C, and all the "drawing" commands. Note that all these commands are irrelevant for the theorem prover, but are relevant for producing figures (see Figure 5).

The construction shown in Figure 5 was made using GCLC, but the user can also use Eukleides for describing the construction, by instructions very similar to the given ones.

5.2 Testing the Conjecture

Having described the construction of the figure, now we have to add the conjecture. The property to be proved can be expressed in the following way: the

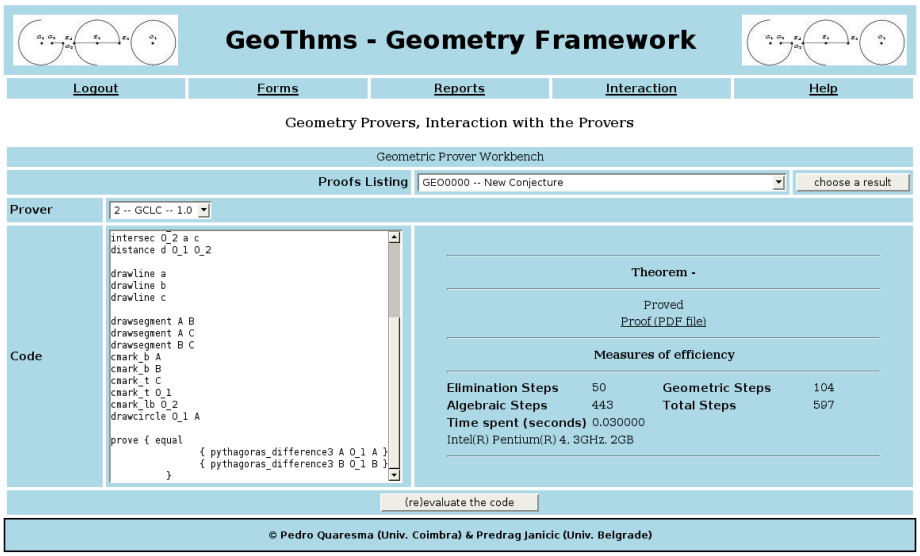

Fig. 6. Circumcircle of a triangle — Interaction with the ATP

points O_1 and O_2 are identical. The user must express this condition within the command `prove`, and using the geometrical quantities supported by the area method, in this case — via the *Pythagoras difference* geometric quantity (for more details, see [15]).

All the commands used in the construction of the figure are internally (within the prover) transformed into primitive constructions of the area method. The GCLC's code can be submitted to GCLCprover without modifications, the Eukleides' code needs to be converted with the `euktogclcprover` tool. As shown in Figure 6, the proof status and the measures of efficiency are accessible, the proof is given as a PDF file. Figure 7 shows the last steps of the proof made by GCLCprover. The proof was generated in 0.03 seconds.[8]

5.3 Inserting a Result in the Database

The user (with the status of contributer) can select the "Forms" section in order to insert a statement for the new result and the corresponding figure and proof (see Figure 8). The statement is kept in the database in LaTeX format and in declarative ATP's code[9], the figure description is kept in DGS's code and also in

[8] Many complex geometry theorems can be proved by the system in only milliseconds. For instance: theorems by Ceva (0.001s), Gauss (0.029s), Thales (0.001s), Menelaus (0.002s), Pappus' Hexagon (0.040s), midpoint theorem (0.002s), ratio of areas of parallelograms (0.190s), etc.

[9] ATP's code share most of DGS's code, the only difference is the conjecture itself, which does not appear in DGS's code

$$(113)\quad (0.062500 \cdot (P_{CBC} \cdot S_{BAC})) = \left(\frac{1}{4} \cdot \left(P_{CBM_a^0} \cdot S_{BAM_a^0}\right)\right) \quad , \text{ by algebraic simplifications}$$

$$(114)\quad (0.062500 \cdot (P_{CBC} \cdot S_{BAC})) = \left(\frac{1}{4} \cdot \left(\left(P_{CBB} + \left(\frac{1}{2} \cdot (P_{CBC} + (-1 \cdot P_{CBB}))\right)\right) \cdot S_{BAM_a^0}\right)\right) \quad , \text{ by Lemma 29 (point } M_a^0 \text{ eliminated)}$$

$$(115)\quad (0.062500 \cdot (P_{CBC} \cdot S_{BAC})) = \left(\frac{1}{4} \cdot \left(\left(0 + \left(\frac{1}{2} \cdot (P_{CBC} + (-1 \cdot 0))\right)\right) \cdot S_{BAM_a^0}\right)\right) \quad , \text{ by geometric simplifications}$$

$$(116)\quad (0.062500 \cdot S_{BAC}) = \left(\frac{1}{8} \cdot S_{BAM_a^0}\right) \quad , \text{ by algebraic simplifications}$$

$$(117)\quad (0.062500 \cdot S_{BAC}) = \left(\frac{1}{8} \cdot \left(S_{BAB} + \left(\frac{1}{2} \cdot (S_{BAC} + (-1 \cdot S_{BAB}))\right)\right)\right) \quad , \text{ by Lemma 29 (point } M_a^0 \text{ eliminated)}$$

$$(118)\quad (0.062500 \cdot S_{BAC}) = \left(\frac{1}{8} \cdot \left(0 + \left(\frac{1}{2} \cdot (S_{BAC} + (-1 \cdot 0))\right)\right)\right) \quad , \text{ by geometric simplifications}$$

$$(119)\quad 0 = 0 \quad , \text{ by algebraic simplifications}$$

Fig. 7. Last steps of the proof of the Circumcircle theorem

JPEG format, the proof is kept in PDF format. For these last two, this means that the DGSs and ATPs are called before the actual insertion is made, validating the code. The JPEG and PDF files are kept in order to avoid the re-evaluation of the code each time a user wants to consult the database.

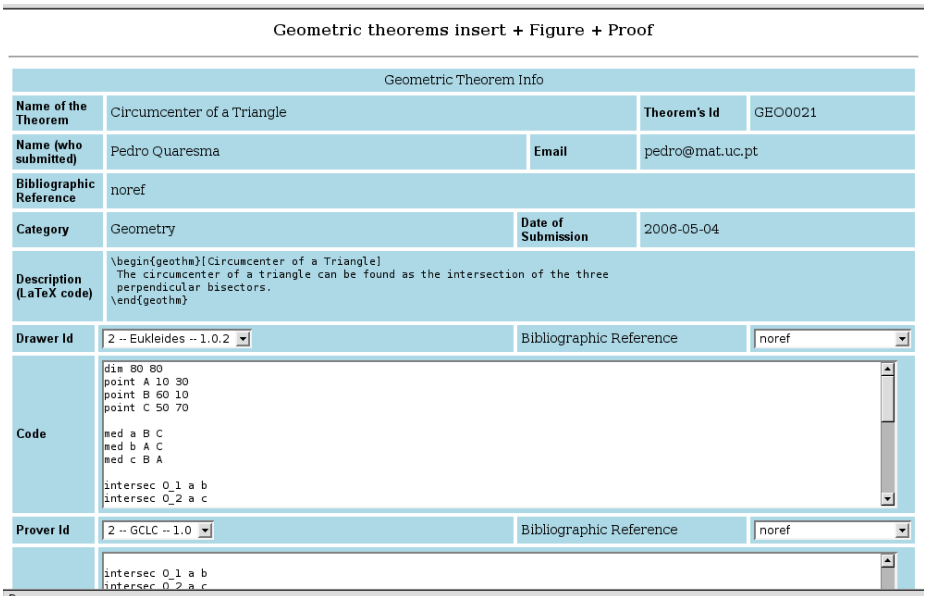

Fig. 8. Circumcircle of a triangle — Insertion Form

After inserting, this new result became available for all users, not only in the "Reports" section, but also in the "Interaction" section. In all cases the user

has access to the code allowing him/her to use it for inclusion in mathematical texts, for testing further results, etc. (see Figure 3).

6 Related Systems

There are, to our knowledge, the following systems, similar to the system presented in this paper: *Geometry Expert* (GEX)[10]; *Ludi Geometrici* (geometriagon)[11]; *Cinderella* [9]; *Discover* [2]; and *GeoView* [1]. The GEX program (new version currently under development) is a DGS with a web interface; it incorporates an ATP, but, unlike GCLCprover, the GEX prover implements an algebraic proof method, and the user can only select one from a limited number of conclusions (e.g., are three selected point collinear?). The GEX tool does not have an accessible database of problems, and does not provide a formatted output for images and proofs. The geometriagon has an already vast repository of problems in the area of classical constructive (ruler and compass only) Euclidean geometry, a registered user can access/edit all problems and solutions. It does not provide an ATP. The user can perform only valid steps in the construction, using only a limited set of tools, and in this way the system is capable to recognise whenever a user has reach a solution of a problem. The geometriagon does not provide any formatted output. Cinderella uses randomise theorem checking to analyse its users actions and to react properly; it does not provide a proof for a given construction in any form. Discover is a DGS that can communicate with Mathematica[12], using the symbolic capabilities of the latter to implement the Gröebner bases method, hence, it is necessary to translate the geometric construction to an algebraic form and back, from the conclusion in algebraic form to its geometric counterpart. No proof in any form is provided. The Geoview software combines the Coq[13] ATP and the GeoplanJ[14] DGS into a system where it is possible to edit statements of geometrical theorems, and to visualise the statement using the DGS. The proofs are not accessible. None of this last three systems have a database of problems easily accessible to its users.

7 Further Work

Automated theorem provers, applications, and repository of problems are often developed separately. In some cases, joint efforts of numbers of researchers led to standards such as DIMACS (for propositional logic) [5] and SMT (for satisfiability modulo theory) [17] and repositories of problems such as SAT-lib (for propositional logic) [7], TPTP (for predicate logic) [18], SMT-lib (for satisfiability modulo theory) [17] etc. Such efforts, standards, and libraries are fruitful for

[10] GEX tool: `http://woody.cs.wichita.edu/gex/7-10/gex.html`
[11] geometriagon: http://www.polarprof.net/geometriagon/
[12] http://www.wolfram.com
[13] http://coq.inria.fr/
[14] http://erathostene.math.univ-montp2.fr/SPIP/De-Geoplan-Geospace-a-GeoplanJ

easier exchange of problems, ideas, and even program code. However, this is often very demanding and there are no many systems smoothly integrating libraries of problems, theorem provers, and real-world applications. In the previous sections, we presented a tightly integrated system consisting of a library of geometry construction problems, dynamic geometry software, and a geometry theorem prover. This system can serve as a good starting point for defining open repository of geometry problems. Currently, geometry conjectures are stored within the description of constructions, in GCLC or in Eukleides language (with additional, natural-language descriptions). This representation is formal, declarative and precise. The strict description of the notion of geometrical constructions and also our experience with GCLC, Eukleides and other similar programs show that different languages are very close to each other (primarily dealing with elementary constructions and isometric transformations, but also with dealing with scaling of figures, labelling components of figures, etc.). We believe that descriptions in all these languages can be normalised, i.e., transformed to a single description. We have already developed the converter from Eukleides to GCLC, but similar converters can be made for other pairs of languages. We propose defining such a normal, referent form, and making a repository usable by all geometry programs. Such language should have a XML version (in a similar way as for SMT-LIB [11]), closer to wide relevant mathematical initiatives such as MathML.[15] That way, it would be possible to store descriptions of constructions in a quality form that provides both formal mathematical contents and visual contents. Moreover, the generic XML validation mechanism could be used for verifying whether a given construction is legal.

8 Conclusions

In this paper we presented our framework GeoThms consisting of dynamic geometry software GCLC and Eukleides, automated theorem prover GCLCprover, and the repository of constructive geometry conjectures geoDB, all accessible through a web interface.

This complex framework provides an environment suitable for new ways of studying and teaching geometry at different levels. In addition, this framework provides an environment for storing mathematical knowledge (in explicit, declarative way) — about geometrical constructions, proofs, and illustrations. We hope that support from interested parties will make GeoThms growing and widely used repository.

We are planning to link additional geometry programs and additional theorem provers to our framework and to further develop the web interface. We are also considering developing a referent geometry language that can be linked to all geometry programs dealing with Euclidean constructions.

[15] MathML is the Mathematical Markup Language. It is an XML application for describing mathematical notation and capturing both its structure and content.

References

1. Yves Bertot, Frédéric Guilhot, and Locï Pottier. Visualizing geometrical statements with geoview, 2004. http://www-sop.inria.fr/lemme/geoview/geoview.ps.
2. Francisco Botana and José L. Valcarce. A dynamic-symbolic interface for geometric theorem discovery. *Computers and Education*, 38:21–35, 2002.
3. C. C. Chou, OU X. S. Gao, and J. Z. Zhang. Automated production of traditional proofs for constructive geometry theorems. In *Eighth Annual IEEE Symposium on Logic in Computer Science*, 1993.
4. Shang-Ching Chou, Xiao-Shan Gao, and Jing-Zhong Zhang. Automated generation of readable proofs with geometric invariants, I. multiple and shortest proof generation. *Journal of Automated Reasoning*, 17:325–347, 1996.
5. DIMACS. Satisfiability suggested format. ftp://dimacs.rutgers.edu/pub/challenge/satisfiability/doc/satformat.tex.
6. Mirjana Djorić and Predrag Janičić. Constructions, instructions, interactions. *Teaching Mathematics and its Applications*, 23(2):69–88, 2004.
7. Holger H. Hoos and Thomas Stützle. Satlib: An online resource for research on sat. In *Proceedings of SAT 2000*. IOS Press, 2000. SATLIB is available online at www.satlib.org.
8. Predrag Janičić and Ivan Trajković. WinGCLC — a Workbench for Formally Describing Figures. In *Proceedings of the 18th Spring Conference on Computer Graphics (SCCG 2003)*, pages 251–256, Budmerice, Slovakia, April, 24-26 2003. ACM Press, New York, USA.
9. Ulrich Kortenkamp and Jürgen Richter-Gebert. Using automatic theorem proving to improve the usability of geometry software. In *Procedings of the Mathematical User-Interfaces Workshop 2004*, 2004.
10. Leslie Lamport. *LaTeX: A Document Preparation System*. Addison-Wesley Publishing Company, 2nd edition, 1994.
11. Filip Marić and Predrag Janičić. SMT-LIB in XML clothes. In *Proceedings of Workshop Pragmatics of Decision Procedures in Automated Reasoning (PDPAR-2004)*, 2004.
12. Noboru Matsuda and Kurt Vanlehn. Gramy: A geometry theorem prover capable of construction. *Journal of Automated Reasoning*, 32:3–33, 2004.
13. Julien Narboux. A decision procedure for geometry in coq. In *Proceedings TPHOLS 2004*, volume 3223 of *Lecture Notes in Computer Science*. Springer, 2004.
14. Christian Obrecht. Eukleides. http://www.eukleides.org/.
15. Pedro Quaresma and Predrag Janičić. Framework for constructive geometry (based on the area method). Technical Report 2006/001, Centre for Informatics and Systems of the University of Coimbra, 2006.
16. Pedro Quaresma and Ana Pereira. Visualização de construções geométricas. *Gazeta de Matemática*. To appear.
17. Silvio Ranise and Cesare Tinelli. The SMT-LIB Format: An Initial Proposal. 2003. on-line at: http://goedel.cs.uiowa.edu/smt-lib/.
18. Geoff Sutcliffe. The tptp problem library. http://www.cs.miami.edu/~tptp/TPTP/TR/TPTPTR.shtml.